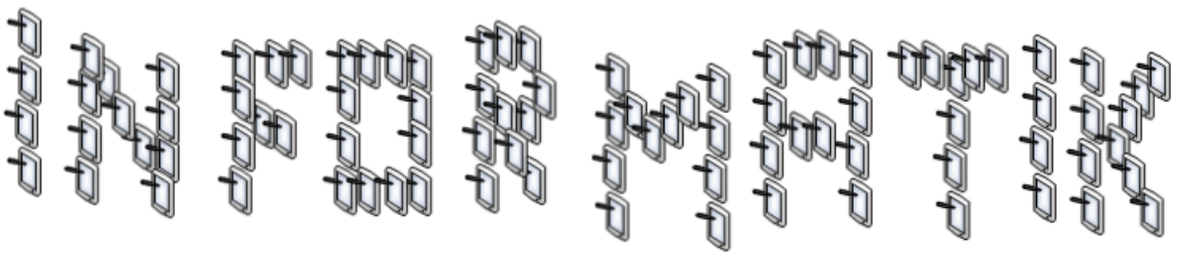


Informatik

Internet

Grundlagen der Programmierung

Alexandra Horn, September 2014

Mein Dank geht an alle Schüler, die vor dem Druck des Skriptes geholfen haben, Fehler im Skript zu finden.

Internet

Wie funktioniert das Internet?

Rechnernetze

Einteilung von Computernetzen

Computernetze lassen sich nach verschiedenen Kriterien wie Größe, Art der Übertragungswege oder Aufbau einteilen.

Einteilung nach Ausdehnung

Betrachtet man die geografische Ausdehnung von Computernetzen, so ist entscheidend, wie weit die Computer im Netzwerk auseinander sind. Prinzipiell kann man zwischen lokalen und nicht-lokalen Computernetzen unterscheiden.
Unter einem LAN, also einem Local Area Network, versteht man kleinere Rechnernetze mit geringer Ausdehnung. Ein LAN findet sich oft innerhalb einer Wohnung, einer Etage oder einem ganzen Gebäude.
Die Computer in einem nicht-lokalen Netz sind deutlich weiter voneinander entfernt. Wide Area Networks (WAN) haben eine Ausdehnung ab 10 km und nutzen sowohl private als auch öffentliche Datenübertragungseinrichtungen, um die Computer zu verbinden. Die größte Ausdehnung haben die so genannten Global Area Networks (GAN), welche verschiedene Kontinente verbinden. Der bekannteste Vertreter eines globalen Netzes ist das Internet.

Einteilung nach Netzarchitektur

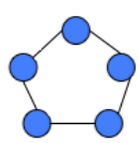
Unterscheidet man Computernetze dagegen hinsichtlich der Aufgabenteilung, dann gibt es zwei grundsätzliche Einteilungen: entweder man hat eine Client-Server-Architektur oder aber eine Peer-to-Peer-Architektur.
In einer Client-Server-Architektur stellt ein Rechner (der Server) die Ressourcen zur Verfügung, die dann von mehreren anderen Rechnern (den Clients) genutzt werden können. Das hat den Vorteil, dass nur der Server rechenintensive Vorgänge ausführen muss und alle dafür benötigten Informationen bereits bei sich gespeichert hat. Der Client stellt die angeforderten Daten nur dar oder übermittelt meistens geringe Datenmengen an den Server. Diese Architektur hat jedoch den Nachteil, dass beim Ausfall des Servers kein Client mehr an die Informationen kommt. Wenn außerdem viele Clients Informationen beim Server anfordern, kann es passieren, dass der Server überfordert ist und nicht alle Anfragen bearbeiten kann.
In Peer-to-Peer-Architekturen hingegen ist die Trennung zwischen Client und Server weitgehend aufgehoben. Ein Peer ist gleichzeitig das Eine und das Andere. Gleichzeitig zu einer eigenen Anfrage kann ein Peer im Netz also auch Anfragen anderer Peers bearbeiten. Dadurch haben Peer-to-Peer-Netzwerke den großen Vorteil sehr ausfallsicher zu sein und im Idealfall alle Rechner ungefähr gleich belastet werden. Zusätzlich gibt es noch Ansätze, die versuchen die Vorteile beider Ansätze zu verbinden und die jeweiligen Nachteile zu vermeiden. Diese Architekturen nennen sich hybride Architekturen.

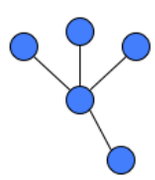
Einteilung nach physikalischer Struktur

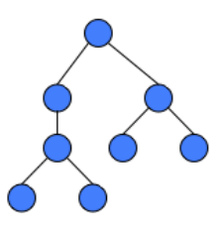
Damit die Kommunikationsteilnehmer in einem Computernetz überhaupt Informationen miteinander austauschen können, müssen sie physikalisch durch ein Medium miteinander verbunden sein. Die Verbindung von zwei Datenstationen mit einem Kommunikationsmedium nennt man dabei Übertragungsweg. Die kodierten Informationen werden auf dem Medium durch elektrische oder optische Signale oder durch Funk übertragen. Als Medium kommen im kabelgebundenen Fall Telefon-, Kupfer- oder Glasfaserkabel in Frage. Bei Funkverbindungen unterscheidet man zwischen erd-gebundenem (terrestrischem) und Satellitenfunk. Bei optischen Verbindungen unterscheidet man unter anderem zwischen Infrarot- oder Laserverbindungen.
Bei der Übertragung von Informationen über Telefonkabel werden elektrische Signale übertragen. Kabel lassen sich recht leicht verlegen, sind jedoch anfällig gegenüber Störungen. Außerdem sind sie nicht abhörsicher.
Bei der Kabelverbindung mit Glasfaserkabel (manchmal auch Lichtwellenleiter genannt) werden hingegen optische Signale versendet. Dazu müssen die elektrischen Impulse zunächst in Lichtsignale umgewandelt werden. Glasfaserkabel sind nicht durch elektromagnetische Störungen zu beeinflussen, abhörsicherer und erreichen extrem hohe Übertragungsleistungen.
In Mobilfunknetzen existieren viele Basisstationen, die den Funk aussenden und so die mobilen Kommunikationsteilnehmer erreichen. Die Basisstationen selbst sind dann jedoch wieder per Kabel an das Kommunikationsnetz angebunden.

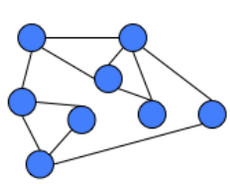
Einteilung nach Netztopologie


Linienstruktur


Ringstruktur


Sternstruktur


Hierarchische Struktur

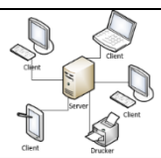
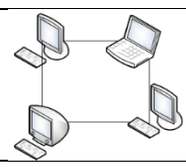

Dezentrale Struktur

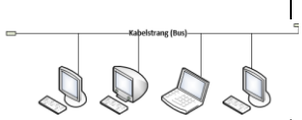
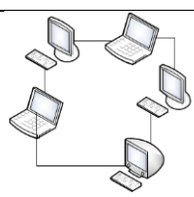
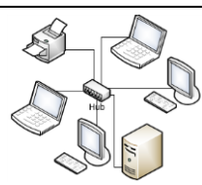
Die Kommunikationsteilnehmer in einem Netzwerk sind immer in einer bestimmten Struktur (einer Topologie, siehe Abbildung rechts) angeordnet. Die Ausfallsicherheit eines Netzes hängt stark von seiner Topologie ab – je mehr alternative Wege es zwischen zwei Stationen im Netzwerk gibt, umso ausfallsicherer ist es.
Ein Netzwerk mit Linienstruktur, auch Bus genannt, besteht aus mehreren Knoten, die linienförmig verbunden sind. Manchmal nennt man diese Topologie auch Busstruktur. Das ist die einfachste Art ein Netz aufzubauen, allerdings müssen Knoten im Inneren des Netzes mehr Daten verarbeiten als äußere Knoten. Außerdem ist die Kommunikation sofort unterbrochen, wenn ein innerer Knoten ausfällt.
Fügt man zwischen dem ersten und letzten Knoten in der Linie eine Verbindung hinzu, erhält man eine Ringstruktur.
In einem sternförmigen Netzwerk hat jeder Knoten eine Verbindung zu einem Zentralknoten. Wenn dieser ausfällt, bricht das gesamte Netzwerk zusammen.
Ein hierarchisches Netzwerk ist eine erweiterte Form eines sternförmigen Netzwerks und im Grunde eine baumartige Anordnung der Knoten. Jeder Teilbaum für sich genommen ist sternförmig aufgebaut und beim Ausfall eines Knotens auf einer höheren Ebene ist der komplette Teilbaum vom Netzwerk abgeschnitten.

In einem dezentralen Netzwerk sind die Knoten ohne spezielles Schema verbunden. Die verschiedenen Verbindungen erhöhen die Ausfallsicherheit durch eine Vielzahl von Wegen im Netzwerk.[1]

Aufgaben:

1. Erkläre die Begriffe LAN, WAN und GAN.
2. Übertrage die Tabellen in dein Heft und ergänze sie.

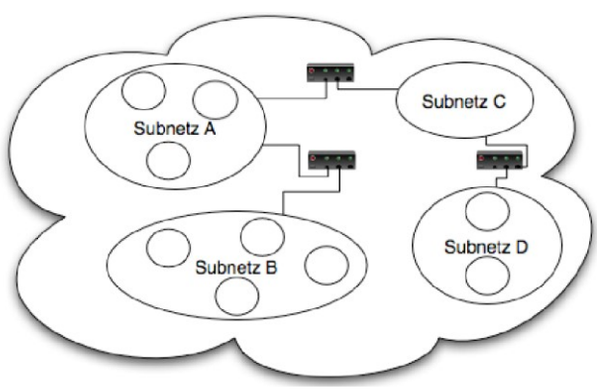
	Client-Server 	Peer to  Peer
Erklärung		
Vorteil		
Nachteil		

	Bus	Ring	Stern
Topologie (Aufbau)			
Vorteil			
Nachteil			

3. Was versteht man unter einer hybriden Architektur?

Struktur des Internet

Der allgemeine Aufbau des Internets



Das Internet ist ein globales Netz von Computernetzwerken. Über Computer mit einem Internetzugang kann weltweit elektronisch kommuniziert werden. Die physikalische Infrastruktur besteht aus mehreren verschiedenen Netzwerken (Subnetzen), die wiederum aus mehreren Subnetzen bestehen können. (siehe Abb. (Skrotzky, 2010, S. 9))
Damit man seinen eigenen Computer mit dem Internet verbinden kann, benötigt man einen Provider und spezielle Hardware. Heutzutage ist dies meist ein Kabelmodem oder ein DSL-Router. Möchte man mehrere Rechner an ein Kabelmodem anschließen, so benötigt man zusätzlich noch einen separaten Router. An

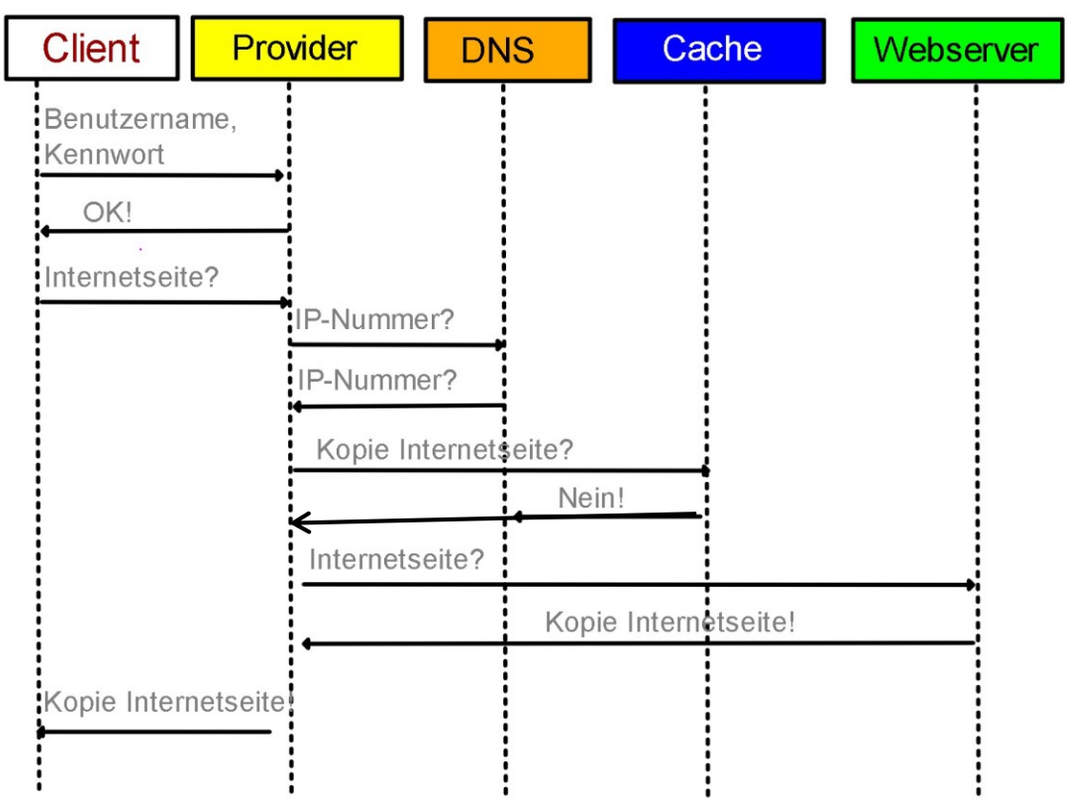
[1] Vgl. (Magenheim J.) Zusatzmaterial Kapitel Netze

einem DSL-Router können Computer über Kabel (Ethernet-Router) oder über Funk (WLAN-Router) angeschlossen werden. Entsprechend können lokale Netze, z. B. das Schulnetz oder das Netz einer Firma über einen Kommunikationsrechner oder Router mit dem Internet verbunden werden. Innerhalb des lokalen Netzes wird die Ethernet- und WLAN-Technologie zur Daten-Kommunikation verwendet. Leistungsfähigere Übertragungswege, beispielsweise zwischen universitären Einrichtungen oder nationalen Providern, basieren auf der Glasfasertechnik. Über die Einwahlknoten der nationalen Internetprovider sind die regional operierenden Internetdienstleister und Endkunden an das Internet angeschlossen. Die nationalen Internetprovider sind dann über Transkontinentalkabel und Satelliten mit den nationalen Internetprovidern der anderen Kontinente verbunden.
Dass trotz unterschiedlichster Hardware-Technologien die Kommunikation zwischen beliebigen Computern im Internet möglich ist, beruht letztlich auf der Vergabe global gültiger einheitlicher Internetadressen und der Verwendung von einheitlichen Kommunikationsprotokollen. (Röhner G. , Unterrichtsskript: Grundkurs Informatik E1, 2010/2011, S. 25)

Aufgabe:

4. Um Computer in einem lokalen Netzwerk miteinander zu verbinden, verwendet man meistens keine Router sondern Switchs. Was ist ein Switch? Welche Vor- und Nachteile ergeben sich, wenn man Computer mit Hilfe eines Switch verbindet?

Sequenzdiagramm zum Abruf einer Internetseite



Internet Protokol (IP) und Domain Name System (DNS)

Computer kommunizieren miteinander über das Internet, indem sie das Internet Protocol (IP) benutzen. Jeder Rechner, der am Datenaustausch im Internet teilnimmt, hat eine weltweit einmalige 32-Bit-Zahl als Adresse, die so genannte IP-Adresse. Diese 12stellige Zahl wird in durch Punkte getrennten Gruppen mit je 3 Ziffern geschrieben, damit man sie sich leichter merken kann (z. B. 10.12.70.199 anstatt von

010012070199). Über das Domain Name System (DNS) lässt sich für jede Domain die zugehörige IP-Adresse ermitteln.

Wie ist eine IP-Adresse aufgebaut?

Bit	1				8				16				24				32
Klasse A	0	Netz-ID							Host-ID								
Klasse B	1	0	Netz-ID										Host-ID				
Klasse C	1	1	0	Netz-ID												Host-ID	

(Röhner G. , Unterrichtsskript: Grundkurs Informatik E1, 2010/2011, S. 25)

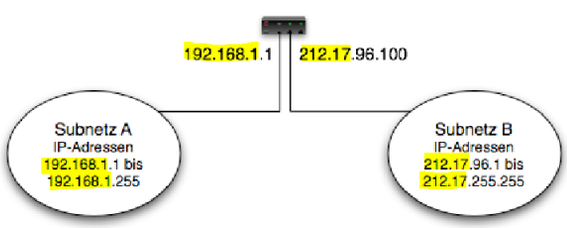
Das erste Bit gibt analog zum Telefonnetz die Adressklasse an, anhand der man erkennen kann, wie lang der Netz-ID-Bereich jeweils ist. Die Netz-ID entspricht der Vorwahl und die Host-ID der Rufnummer. So gibt die IP-Nummer nicht nur darüber Auskunft, wer der betreffende Netzteilnehmer ist, sondern auch wo er sich befindet.

Computer	IP	AND 255.255.255.0	im Netz
1	192.168.0.27	192.168.0.0	192.168.0
2	192.168.0.138	192.168.0.0	192.168.0
3	192.168.4.26	192.168.4.0	192.168.4

(Röhner G. , Unterrichtsskript: Grundkurs Informatik E1, 2010/2011, S. 25)

Wie kann man herausfinden, welche IP-Adresse man für seinen eigenen Computer verwenden darf?

Die Antwort ist sehr einfach: Darum muss man sich als Benutzer des Internets gar nicht kümmern! Die Provider verfügen über einen Vorrat an IP-Adressen, die sie deinen Kunden zuweisen. Ist der Computer über ein Kabelmodem direkt verbunden, so bekommt der Rechner eine IP-Adresse zugewiesen. Ist der Computer über einen Router mit dem Internet verbunden, so erhält der Router die IP-Adresse. Der Computer befindet sich dann im lokalen Netzwerk. Da ein Router dazu dient, zwei Subnetze miteinander zu verbinden, benötigt er auch zwei IP-Adressen. (siehe Abb. (Skrotzky, 2010, S. 10))



Für lokale Netzwerke sind drei Bereiche vorgesehen, so dass diese IP-Adressen nicht von einem Router ins Internet weitergegeben werden. Stattdessen verwendet der Router seine eigene IP-Adresse.
Diese drei Bereiche sind:

- Alle IP-Adressen, die mit 10 als erster Zahl beginnen. Diese IP-Adressen sind für große lokale Netzwerke gedacht.
- Alle IP-Adressen, deren erste Zahl 172 und deren zweite Zahl zwischen 16 und 31 liegt.
- Alle IP-Adressen, die mit 192.168 beginnen. Sie werden häufig bei privaten lokalen Netzwerken verwendet.

Die Loopback-Adresse 127.0.0.1

Diese Adresse kann man verwenden, wenn man einen Funktionstest bei der Programmierung von Internetanwendungen durchführen will, da die Datenpakete direkt wieder beim Sender landen, ohne jemals das Internet betreten zu haben.

Wie kann ich meine eigene IP-Adresse herausbekommen?

Unter Windows 7 gibt man im Startmenü **cmd** ein, dann öffnet sich dort ein Konsolenfenster. Hier gibt man nun **ipconfig** ein und erhält eine Auflistung der aktuellen IP-Konfiguration des Rechners.

Daten werden nun in IP-Pakete verpackt, auf denen die IP-Nummer des Empfängers und des Senders vermerkt sind. Diese Pakete kreisen nun so lange von Rechner zu Rechner, bis sie am Ziel ankommen.

Wie kann man testen, ob eine Verbindung zwischen zwei Rechnern existiert?

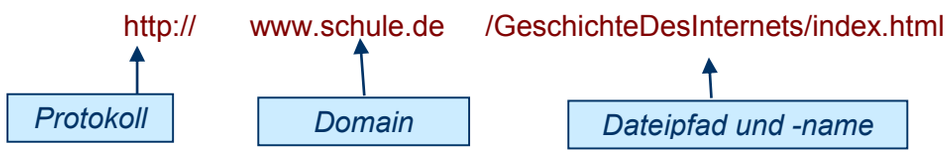
Im selben Konsolenfenster kann man den Befehl **ping** x.x.x.x eingeben, wobei x.x.x.x für die IP-Adresse des fremden Rechners steht, mit dem man eine Verbindung mit dem eigenen Rechner überprüft werden soll. Existiert eine Verbindung, so wird die Verbindungsanfrage (ping) vom zweiten Rechner mit einer Antwort (**pong**) bestätigt.

Aufgaben:

5. Wie ist dein häuslicher Rechner an das Internet angeschlossen? Lasse dir dazu mit **ipconfig/all** die Informationen ausgeben.
6. Welche IP-Adresse hat der Rechner, den du in der Schule verwendest? Hast du hierzu die nötigen Rechte?

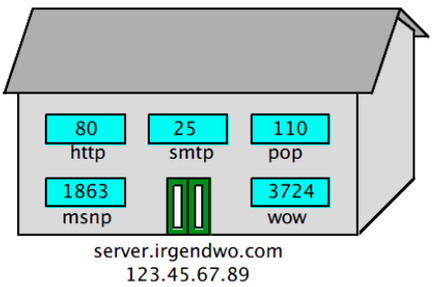
Uniform Resource Locator (Url)

Ein Uniform Resource Locator (die Url) ist eine weltweit eindeutige Ortsangabe für Seiten im Internet. Sie setzt sich aus drei Teilen zusammen: HTTP steht für das Hypertext Transfer Protocol – nun weiß der Computer, dass eine Webseite gesucht wird. Als zweiter Teil folgt die Domain, unter der der gesuchte Server zu finden ist und anschließend der Pfad und Name, unter dem die angeforderte Datei auf dem Server gespeichert ist:



Ports und Dienste

Wenn der Computer mit dem Internet verbunden ist, so möchte man das Internet natürlich auch nutzen. Genauer gesagt, möchte man einzelne Dienste davon nutzen, beispielsweise E-Mail, WWW, Chats, Online-Spiele etc. Hinter all diesen Diensten stehen Programme, die auf Computern laufen.
Sobald ein Computer einen Dienst anbietet, wird er als Server bezeichnet. Obwohl in der Regel spezielle Computer für Serverdienste verwendet werden, könnte grundsätzlich jeder Computer zum Server werden. Damit sich die einzelnen Dienste, die ein Server anbietet, nicht in die Quere kommen, benutzt jeder Dienst einen oder mehrere so genannte Ports. Ist nun auf einem Server ein Dienst installiert und aktiviert, so horcht dieser seinen Port ab, ob eine Anfrage vorliegt. Ist das der Fall, so reagiert er auf die Anfrage.



Man kann sich einen Server ungefähr wie ein Geschäftshaus vorstellen, in dem es verschiedene Bürozimmer (= Ports) gibt. Je nachdem, welches Anliegen ein Kunde hat, muss dieser sich an das entsprechende Büro wenden. Vor der URL steht häufig http://, beispielsweise http://google.ch. HTTP (HyperText Transfer Protocol) ist das Protokoll, das zur Übermittlung von Webseiten verwendet wird. In der oben abgebildeten Skizze sieht man, dass für dieses Protokoll der Port 80 verwendet wird.

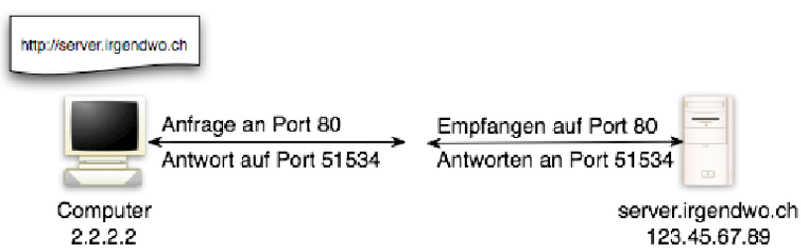
(Skrotzky, 2010, S. 13)

Die übrigen abgebildeten Ports werden für folgende Dienste gebraucht:

• Port 25	SMTP	Simple Mail Transfer Protocol	Verschicken von E-Mails
• Port 110	POP	Post Office Protocol	Empfangen von E-Mails
• Port 1863	MSNP	MSN Protocol	MSN
• Port 3724	WOW	World Of Warcraft	Online-Spiel

Port Nummern gehen von 0 bis 65535.

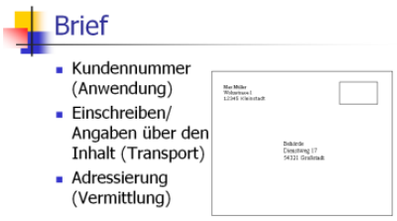
Möchte der Kunde eine Webseite vom Server abrufen, so muss er dazu ins Bürozimmer 80 (also, die Anfrage über Port 80 senden). Möchte er dagegen eine E-Mail verschicken, so muss er sich an das Büro 25 wenden (also den Port 25 benutzen). Die meisten Programme, die auf einen Dienst eines Servers zugreifen, sind so eingestellt, dass sie den jeweiligen Standardport eines Dienstes automatisch benutzen. Somit ist es selten nötig, dass der Anwender sich mit Porteinstellungen beschäftigen muss.
Wenn nun dein Computer eine Webseite auf dem Server server.irgendwo.ch aufrufen möchte, so richtet er an den Server eine entsprechende Anfrage auf dem Port 80. Um die Antwort des Servers zu empfangen, muss dein Computer ebenfalls einen Port öffnen. Da dein Computer nicht selbst ein Webserver ist, sondern nur ein Kunde eines Webservers, kann er für die Antwort nicht den Port 80 verwenden. Er wählt stattdessen zufällig einen nicht registrierten Port. Das sind diejenigen mit den Nummern größer als 49151.
(Abb. (Skrotzky, 2010, S. 14))



Aufgaben:

7. Was ist der wesentliche Unterschied zwischen IP-Adressen von lokalen Netzen, beispielsweise 192.168.1.5, und einer globalen IP-Adresse, beispielsweise 212.117.92.35?
8. Was ist die Aufgabe eines DNS Servers?
9. Was benötigest du, damit du dich mit deinem Computer von zuhause aus mit dem Internet verbinden kannst?
10. Kannst du auf einen Server, beispielsweise einen Webserver, auch zugreifen, ohne dass du dessen Domain Namen kennst?

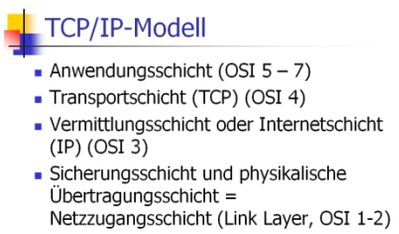
Protokolle und OSI-Modell



Wenn wir einen Brief mit der Post verschicken, werden gewisse Regeln eingehalten. Eine Adresse, die auf dem Umschlag links oben steht, wird als Absender interpretiert, diejenige rechts unten als Empfänger. In einer Adresse schreibt man zuerst den Namen, dann die Straße, dann die Stadt. Wenn der Brief zu einem bestimmten Arbeitsvorgang gehört, gibt es üblicherweise eine Bearbeitungsnummer, anhand derer er beim Empfänger gleich an den richtigen Sachbearbeiter weitergeleitet wird. Gehören zu einem Brief mehrere Anlagen, ist es üblich, deren Anzahl anzugeben.

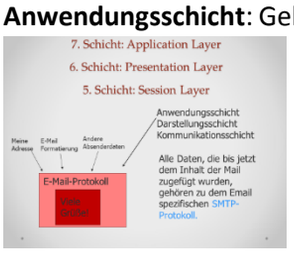
Für die Arbeit im Internet gibt es zwar keine festen Vorschriften, aber Absprachen, die in den RFCs festgehalten sind.
Zwischen zwei entfernten Rechnern werden zunächst nur Signale übertragen, also Strom an/Strom aus, was man auch als 0 und 1 aufschreibt. Wird nun eine Datenübertragung begonnen, steht am Anfang immer eine Folge von Nullen und Einsen, mit der nur mitgeteilt wird, nach welcher der alten Absprachen der Rest übersetzt wird. Man nennt diese Regeln Protokolle.
Die Protokolle beim Postversand kann man auch noch einmal unterteilen:

- Bei der Adressierung soll sichergestellt werden, dass der Brief den richtigen Empfänger findet (Vermittlung).
- Mit der Anzahl der Anlagen soll sichergestellt werden, dass der Brief vollständig ankommt (Transport).
- Wenn man die Bearbeitungsnummer angibt, will man sicherstellen, dass der Brief bei dem richtigen Arbeitsablauf verwandt wird (Anwendung).



Bei einer Nachricht, die im Internet verschickt wird, gibt es (üblicherweise) 4 Protokollschichten. In jeder dieser Schichten erhält die Nachricht neue Kopfdaten, was einem weiteren Umschlag mit Beschriftung bei dem Brief entsprechen würde.

Anwendungsschicht: Gehört die Nachricht zu einer E-Mail, dem Abruf einer WWW-Seite, einem Chat? Der



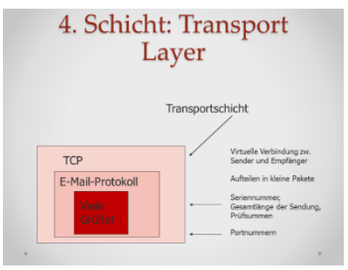
Anwender benötigt Programm, um die entsprechenden Daten sehen und ggf. verändern zu können. Wenn er eine Email schreiben möchte, benötigt er ein Emailprogramm. Soll eine Webseite geöffnet werden, muss ein Browser verwendet werden. Der Anwender möchte die Daten nun in korrekter Form angezeigt bekommen, egal von welchem Hersteller er das entsprechende Programm verwendet.

Das OSI-Modell unterteilt diese Anwendungsschicht in drei Schichten:

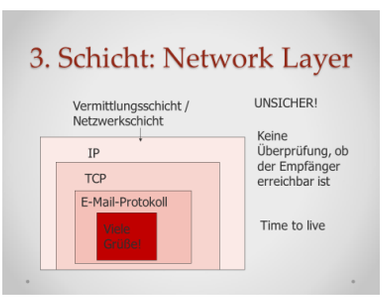
- Schicht 7: Anwendungsschicht (Application Layer) Hier ist die Schnittstelle zwischen dem Computer und der eigentlichen Anwendung. (Browser, Emailprogramm,..)
- Schicht 6: Darstellungsschicht (Presentation Layer) Die Daten der Anwendungsschicht werden in ein einheitliches plattformunabhängiges Dateiformat gebracht, zusätzlich kann diese Schicht die

Daten verschlüsseln oder komprimieren. Der HTML-Code von Webseiten wird beispielsweise in ASCII- Zeichencode codiert, so dass jedes Zeichen durch ein Byte codiert wird, damit alle Rechner diesen Zeichencode lesen können.

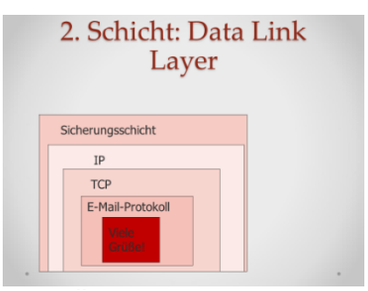
- Schicht 5: Kommunikations- oder Sitzungsschicht (Session Layer) Die Verbindung der an der Kommunikation beteiligten Rechner wird kontrolliert. Eventuelle Sitzungsabbrüche werden hier verwaltet. Benötigte Passwörter für die Verbindung werden in dieser Schicht kontrolliert.



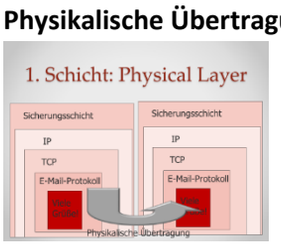
Transportschicht (Schicht 4 Transport Layer) (TCP = Transmission Control Protocol): Hier wird die Nachricht in Pakete zerlegt, die mit einer Prüfsumme und der Position in der Gesamtnachricht versehen werden, damit diese wieder in der richtigen Reihenfolge zusammengesetzt werden kann. Die Zerlegung in Teilpakete ist günstig, weil so bei Übertragungsfehlern nicht die ganze Nachricht wiederholt werden muss, und weil das Netz gleichzeitig vielen Benutzern zur Verfügung steht. Zusätzlich wird der Sende- und Empfangsport (siehe weiter oben) festgelegt. Die Transportschicht ist für die sichere, fehlerfreie und vollständige Übertragung der Pakete verantwortlich.



Vermittlungsschicht (Schicht 3 Network Layer) (IP = Internet Protocol): Hier wird anhand der IP-Adresse dafür gesorgt, dass alle Nachrichtenteile deinen Weg zum Ziel finden. Für die Weitervermittlung von Nachrichten haben die Knotenrechner sogenannte Routingtabellen, aus denen sie entnehmen, an welche anderen Rechner sie die Nachricht weiterschicken können. (Mit einem Programm wählen sie den günstigsten Weg aus.) Die Knotenrechner werden deshalb auch Router genannt. Damit Pakete, die Dein Ziel verfehlen, nicht ewig weitergereicht werden, haben sie eine Art Haltbarkeitsdatum.



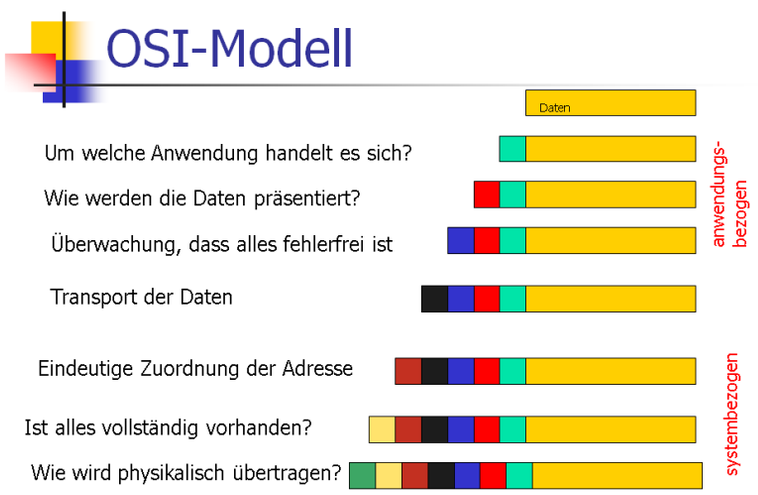
Sicherungsschicht (Schicht 2 Data Link Layer): Diese Schicht ist für die fehlerfreie Signalübertragung zuständig. Sie hängt von der Arbeitsweise des Teilnetzes (Ethernet, Token Ring, ...) ab, in dem sich das Paket gerade befindet. An Knotenpunkten wird eine Kopie zurückbehalten bis eine Übertragungsbestätigung eintrifft, so dass beschädigte oder verlorene Pakete neu versendet werden können.



Physikalische Übertragung (Schicht 1 Physical Layer) Hier werden die verwendeten Übertragungsmedien (diverse Kabel, Infrarot- oder Funkübertragung) sowie die Schnittstellen mit Spannungspegeln, Steckverbindern und Datenübertragungsraten beschrieben. Das Übertragungsmedium an sich ist kein Bestandteil dieser Schicht. Hier geht es nur um die notwendigen Definition bzgl. Strom- und Spannungswerten etc. Repeater arbeiten auf dieser Schicht.

Im Zielrechner wird die Nachricht wieder rückwärts entpackt. Jede Schicht liest dabei die im Header verpackte Information, verarbeitet diese Information und entfernt dann den Header. Auf diese Weise kommuniziert jede Schicht des Senders mit der entsprechenden Schicht des Empfängers. Keine einzige Schicht darf dabei übersprungen werden. Die Kommunikation zwischen den einzelnen Schichten funktioniert nur über definierte Schnittstellen zwischen direkt aufeinander folgende Schichten (Service Access Point). Auf diese Art können Rechner unterschiedlichster Bauart miteinander kommunizieren.

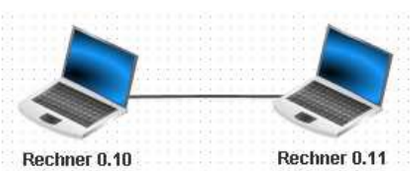
Anwendung	File Transfer	Electronic Mail	WWW	Domain Name Service
Darstellung	File Transfer Protocol (FTP)	Simple Mail Transfer Protocol (SMTP)	Hyper Text Transfer Protocol (http)	Domain Name System (DNS)
Kommunikation				
Transport	Transmission Control Protocol (TCP)			User Datagram Protocol
Vermittlung	Address Resolution Protocol (ARP)		Internet Protocol (IP)	Internet Control Message Protocol (ICMP)
Sicherung	Ethernet, Token Ring,…			
Bitübertragung	Übertragungsmedium (Kabel, Funk,…)			



Netzwerke mit FILIUS simulieren

FILIUS ist eine interaktive Lernumgebung, die an der Universität in Siegen entwickelt worden ist. Mit Hilfe dieser Lernsoftware können Netzwerke aufgebaut, konfiguriert und getestet werden. (Garmann, 2011)[2]

Einstiegsaufgaben:

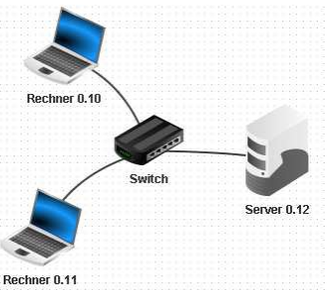


1. **Vernetzung zweier PCs**:
   Erstelle ein Netzwerk mit zwei vernetzten Computern, welche beide eine Client-Funktion haben. Die Computer sollen die abgebildeten Namen sowie die IPs 192.168.0.10 und 192.168.0.11 besitzen. Durch die Subnetzmaske 255.255.255.0 kannst du sicherstellen, dass beide Computer im selben Netzwerk liegen.
2. **Überprüfung der Verbindung mit dem ping-Befehl:**
   Installiere auf dem Rechner 0.10 das Programm Befehlszeile. Starte Befehlszeile und teste die Verbindung zum Rechner 0.11 mit dem Befehl „ping 192.168.0.11“. Beobachte die Netzwerkaktivität, indem du dir den Datenaustausch von Rechner 0.10 anzeigen lässt.

[2] Die Aufgaben wurden nach dem Skriptum „Netzwerke mit FILIUS“ erstellt.

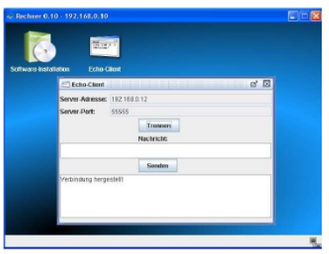
3. Teste auch andere Befehle mit Hilfe des Programms Befehlszeile, wie z. B. die Befehle „ipconfig“, oder „host localhost“ oder „dir“. Der Sinn des host-Befehls wird zu einem späteren Zeitpunkt im Zusammenhang mit einem DNS-Server evtl. deutlicher.

4. **Erweiterung des lokalen Netzwerkes**:
   Erweitere nun das Netzwerk um einen dritten Computer, einen Server, mit dem abgebildeten Namen und der IP 192.168.0.12, welcher im gleichen Netzwerk liegen soll. Achte darauf, dass du von nun an für Computer mit der Funktion eines Servers, das Rechner-Symbol wählst. 

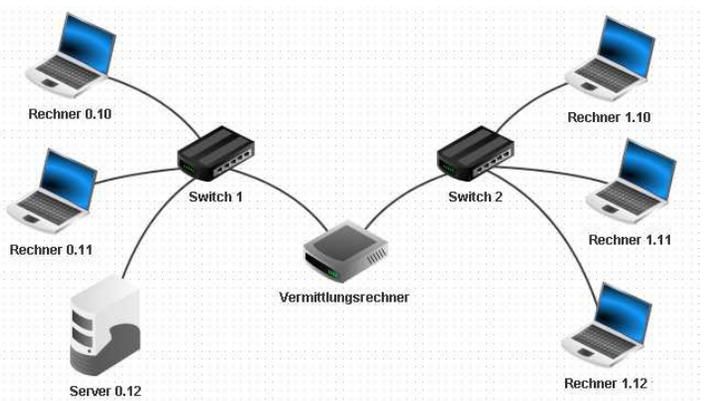


   Verbinde alle Computer mit einem Switch wie abgebildet.

5. **Überprüfung der Verbindung mit einem Echo-Server:**
   Installiere auf dem Server 0.12 einen Echo-Server und starten diesen auf dem voreingestellten Port 55555. Installiere auf einem Client einen Einfachen-Client und verbinde diesen mit dem Server. Sende vom Client einige Textnachrichten und beobachte den Effekt. Schaue dir auch die Netzwerkaktivität im Datenaustausch-Fenster des Clients an.



6. **Erweiterung des Netzwerkes mit einem Router**:



   Erstelle neben dem bereits vorhandenen Netzwerk ein weiteres Netzwerk mit drei Rechnern wie abgebildet. Die neuen Rechner sollen sich in einem logisch anderen Netzwerk befinden. Wähle dafür die IPs 192.168.1.10 bis 192.168.1.12. Verbinden anschließend die beiden Netzwerke mit einem Vermittlungsrechner (Router), welcher die Netzwerkkarten mit den IPs 192.168.0.1 und 192.168.1.1 besitzt. Prüfe anschließend im Programm Befehlszeile mit einem ping-Befehl die Verbindung von Rechner 0.10 zum Rechner 1.10.

   Wenn du alles richtig gemacht hast, erhältst du folgende Meldung:

```
root /> ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10)
--- 192.168.1.10 Paketstatistik ---
4 Paket(e) gesendet, 0 Paket(e) empfangen, 100% Paketverlust
```

   Das Problem liegt darin, dass die Verbindungsanfrage das eigene Netz verlassen müsste. Allerdings ist bei den einzelnen Rechnern noch kein Gateway eingestellt, welches bestimmt, wohin die Nachrichten geschickt werden sollen, die nicht im Netzwerk verbleiben sollen. Das erledigen wir nun in Aufgabe 7.

7. **Konfiguration des Gateways:**
   Der Vermittlungsrechner hat eine Netzwerkkarte 192.168.0.1. Diese Adresse stellst du als Gateway der drei linken Computer ein. Entsprechend stelle das Gateway 192.168.1. 1 für die drei rechten Computer ein. Teste anschließend die gleiche Verbindungsanfrage, welche nun ohne Fehler funktionieren sollte.

```
root /> ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10)
From 192.168.1.10 (192.168.1.10): icmp_seq=1 ttl=63 time=1000ms
From 192.168.1.10 (192.168.1.10): icmp_seq=2 ttl=63 time=500ms
From 192.168.1.10 (192.168.1.10): icmp_seq=3 ttl=63 time=500ms
From 192.168.1.10 (192.168.1.10): icmp_seq=4 ttl=63 time=501ms
--- 192.168.1.10 Paketstatistik ---
4 Paket(e) gesendet, 4 Paket(e) empfangen, 0% Paketverlust
```

8. **Überprüfung der Verbindung mit einem Echo-Server:**
   Teste die Netzwerkverbindung auch mit dem Einfacher-Client und Echo-Server. Installiere dazu auf dem Rechner 1.10 einen Echo-Client und verbinde ihn mit dem Server 0.12.

9. **Einrichtung eines Webservers mit gespeicherter Internetseite:**
   Installiere auf dem Server 0.12 einen Webserver und einen Texteditor. Starte den Texteditor und öffne hiermit die Datei index.html aus dem virtuellen Verzeichnis root/webserver.

index.html

Datei

```
<html>
  <head>
    <title>Standardseite</title>
  </head>
  <body bgcolor="#ccddff" style="font-family:Verdana; text-align:center;">
    <h2> FILIUS - Webserver </h2>

    <p>Herzlich Willkommen auf dem Webserver der Anwendung FILIUS!</p>

    <p> Diese Seite wurde automatisch mit der Installation des
      Webservers eingerichtet, es lassen sich jedoch auch
      eigene Seiten hier unterbringen. </p>

    <p align="center"> <img src="splashscreen-mini.png"> </p>

    <p> http://www.lernsoftware-filius.de </p>
  </body>
</html>
```

   Du siehst eine HTML-Datei.
   Zusatzaufgabe (für diejenigen, die schon etwas Ahnung von HTML haben): Passe den html-Code so an, dass eine Seite mit deinen Informationen angezeigt wird. Erstelle auch eine neue Seite kontakt.html, welche von der Startseite verlinkt werden soll.

10. **Aufruf der Internetseite von einem Client aus mit der IP-Adresse:**
    Starte die Anwendung "Webserver" mit einem Doppelklick. Starte dann den virtuellen Webserver über den Button Starten. Installiere dann auf dem Rechner 1.10 einen Webbrowser. Starte den Browser und baue eine Verbindung zum Webserver auf, indem Sie die IP-Adresse http://192.168.0.12 in die Adressleiste des Webbrowsers eingeben. Nun sollte dir die oben angezeigte HTML-Datei im Browser als Internetseite angezeigt werden.
    Allerdings kann man die Seite noch nicht über die URL http://www. filius.de aufrufen. Hierzu müssen wir zunächst einen DNS einrichten und mit dem Netzwerk verbinden.

11. **Erweiterung des Netzwerkes um einen DNS:**
    Erstelle einen neuen Server mit der IP 192.168.2.10 und dem Gateway 192.168.2.1. Ändere die Anzahl der Schnittstellen am Vermittlungsrechner auf drei ab, indem du auf der Registerkarte "Allgemein" des Vermittlungsrechners den Button "Schnittstellen verwalten" anklickst. Ergänze die auf der neuen Registerkarte zur dritten Netzwerkkarte die Einstellungen: IP-Adresse 192.168.2.1 und Netzmaske 255.255.255.0. Verbinde anschließend den neuen Server mit dem Vermittlungsrechner. (Gateway eintragen!) Trage bei jedem Rechner in den Einstellungen die DNS-Server-Adresse 192.168.2.10 ein. Dies entspricht dem gerade erstellten DNS-Server.

12. **Konfiguration des DNS:**
    Installiere auf dem Server 2.10 die Anwendung "DNS-Server" und starte diese Anwendung direkt mit einem Doppelklick. Trage in die Eingabefelder den Domainnamen www.filius.de und die zugehörige IP-Adresse 192.168.0.12 ein und bestätigen die Eingaben mit Button „Hinzufügen“. Starte abschließend den DNS-Server mit dem Button "Starten". Teste die Verbindung von Deinem Webbrowser nun mit der URL http://www.filius.de

13. **Bedeutung des host-Befehls:**
    Zu Beginn hast du schon den Befehl host kennen gelernt. Teste nun noch einmal den host-Befehl, nun allerdings mit der URL www.filius.de. Du wirst sehen, dass der DNS-Server nun seine Dienste tut und die richtige IP-Adresse ermittelt.

```
root /> host www.filius.de
www.filius.de hat die IP-Adresse 192.168.0.12
```

14. **Einrichtung eines E-Mail-Servers:**
    Installiere auf dem Server 0.12, auf dem zuvor dein Web-Server lief, nun die Anwendung "E-Mail-Server" und starte diese Anwendung direkt mit einem Doppelklick. Richte ein neues Konto mit dem Benutzernamen "bob" und dem Passwort "bob" ein. Kontrolliere deine Eingaben in der Konten-Liste. Starte anschließend den E-Mail-Server über den Button Starten.

15. **Anmelden des E-Mail-Servers beim DNS:**
    Starte die Anwendung "DNS-Server" auf dem Server 2.10. Füge einen neuen Mailaustausch mit der Maildomain filius.de und dem Domainnamen des Mailservers www.filius.de ein. Starte den Server erneut. (Wichtig! Ohne Neustart werden die Änderungen nicht wirksam!)

16. **Einrichtung eines Email-Kontos beim Client:**
    Installiere auf dem Rechner 0.10, auf dem du zu Anfang den Terminal installiert hattest, nun die Anwendung "E-Mail-Programm" und starte diese. Klicke auf den Button Konto einrichten und trage die Informationen deines E-Mail-Servers ein (Name: bob, E-Mail-Adresse: bob@filius.de, POP3-Server: www.filius.de, SMTP-Server: www.filius.de, Benutzername: bob, Passwort: bob. Schreibe anschließend an bob@filius.de (also an dich selbst) eine E-Mail und rufe diese auch wieder ab.

17. **Ergänzung eines zweiten Emailkontos:**
    Ergänze nun den E-Mail-Server um eine weitere E-Mail-Adresse bert@filius.de. Richte ein E-Mail-Programm auf dem Rechner 0.11 entsprechend ein, dass bob und bert sich gegenseitig Nachrichten schicken können.

18. **Ergänzung eines zweiten Email-Servers:**
    Ergänze das rechte Netz um einen Server 1.13, richte hierauf einen Mailserver mit der Maildomain filia.com ein. Füge abschließend ein Konto alice@filia.com hinzu. Erweitere die Mailaustausch-Tabelle des DNS-Servers, dass auch diese neue Maildomain erkannt wird. Richte auf dem Rechner 1.10 ein E-Mail-Programm für alice@filia.com ein. Teste den E-Mail-Versand zwischen bob@filius.de und alice@filia.com.

Eigenes Netzwerk erstellen

Du sollst nun ein Netzwerk erstellen, welches du anschließend zur Bewertung an mich sendest. Mindestanforderung an das Netzwerk:

- Ein Web-Server mit der IP-Adresse: 141.99.50.231, Netzmaske: 255.255.255.0.
- Dieser Server ist mit mindestens vier Clients verbunden.
- Ein DNS, der mit dem Netzwerk über einen Router verbunden ist.
- Es soll zwei weitere Rechnernetze geben, die über den Router mit dem bisherigen Netzwerk verbunden sind.
- Sorge dafür, dass in deinem Netzwerk Peter mit dem Passwort Lustig eine E-Mail an Sonja@informatik.de schreiben kann.

Kontrolliere, ob jeder Rechner mit jedem anderen Rechner verbunden ist.

Beantworte folgende Fragen:

a. Welche Funktionen hat ein Webserver?
b. Welche Aufgabe übernimmt der DNS-Server?
c. Kann man einen Rechner auch unter mehreren Namen erreichen?
d. Im Simulationsmodus kann man den Nachrichtenaustausch auf Protokollebene verfolgen. Aktiviere die Datenaustauschansicht, lösche sämtliche Tabellen (rechte Maustaste) und rufe dann parallel, d. h. bei geöffnetem Datenaustauschfenster eine Webseite auf. Einzelne Schichten kannst du evtl. ausblenden. Welche Schichten des OSI-Modells sind beteiligt, wenn du eine Webseite aufrufst?

Einführung in HTML

Die Dokumente für das Internet werden in der Dokumentenbeschreibungssprache HTML geschrieben, sie erlaubt Strukturelemente von Dokumenten wie z. B. Absätze, Überschriften, Listen oder Tabellen zu kennzeichnen.
HTML -> Hypertext Markup Language

- Auszeichnungssprache (Markup Language) mit einfachen Befehlen
- Keine Programmiersprache!
- kann von verschiedenen Browsern gelesen werden
- alle Betriebssysteme verwendbar

} d. h. systemunabhängig

- es genügt ein einfacher Editor, um die HTML-Dokumente zu erstellen

Das erste einfache HTML-Dokument/ Grundstruktur:

```
<html>
<head>
<title>Unser erstes HTML-Dokument</title>
</head>
<body>
<h1>Willkommen im World Wide Web</h1>
</body>
</html>
```

oder:

```
<html>
  <head>
    <title>Unser erstes HTML-Dokument</title>
  </head>
  <body>
    <h1>Willkommen im World Wide Web</H1>
  </body>
</html>
```

Um den Quellcode einer beliebigen Seite im Firefox anzuzeigen, verwendet man die Tastenkombination **Strg und U**. Beim Internet Explorer gelangt man über den Menüpunkt Ansicht-Quellcode dorthin. Aktuell gibt es verschiedene HTML-Versionen, mit denen Seiten im Internet strukturiert worden sind. Die verwendete Version steht im Quelltext hinter doctype. Zumeist wird HTML4 verwendet, XHTML und HTML5 sind die beiden anderen aktuellen Versionen.

HTML4 Dies ist eine stabile HTML-Version, die von allen Browsern gut unterstützt wird. Kleine Codefehler stellen für diese Version kein Problem dar. Wenn man diese Version verwendet, hat man die Sicherheit, dass alle Browser die Inhalte richtig darstellen werden.

XHTML Diese Version ist eine strenge Variante von HTML gemäß XML. Das X steht für „erweiterbar“ und bedeutet, dass Entwickler bei Bedarf eigene Tags festlegen können. Dies ist besonders für Webdatenbankanwendungen nützlich, da Webseitenentwickler eigene Tags für Datenfelder definieren können. XHMTL ist auch für die Wiedergabe von Webseiten auf mobilen Geräten gut geeignet. Wenn man XHTML verwendet, darf man sich kaum Fehler erlauben.

HTML5 Dies ist die neuste Version von HTML und der Nachfolger von HTML4. Das Ziel dieser Version ist es, Unterschiede in der Verarbeitung von HTML-Code zu beseitigen und neue Funktionalitäten anzubieten, z. B Multimedia-Elemente gezielter einbinden. Da diese Version noch nicht als Standard verabschiedet wurde, sind noch Änderungen möglich und die verschiedenen Browser unterstützen HMTL5 noch nicht vollständig. [3]

[3] Vgl. (York & Wegener, 2012, S. 9)

HTML-Befehle werden immer in spitze Klammern gesetzt < >. Sie werden **Tag** (engl. Tag . Etikett, Aufhänger) genannt. Fast alle HTML-Elemente werden durch ein einleitendes und ein abschließendes Tag markiert.

- Zur **besseren Übersichtlichkeit** sollten zusammengehörende Befehlsblöcke **eingerückt** werden.

- **Der Kopf eines HTML-Dokuments:**
  Hier können Informationen zur Autor oder andere Hinweise für Suchdienste stehen. Der Kopf wird durch die beiden Befehle **<head>** eingeleitet bzw. **</head>** beendet.

  Der Titel, der im Browser in der Titelleiste angezeigt wird, sollte mit einem aussagekräftigen Namen versehen werden, da er als Lesezeichen in der Favoritenseite anderer Nutzer viele verwendet wird und viele Suchmaschinen diesen als Suchkriterium verwenden.
  Die verwendete HTML-Version kann mit <!doctype html4.0> angegeben werden.
  Damit Umlaute und Sonderzeichen korrekt angezeigt werden, sollte man im Dokumentenkopf auch die Zeichencodierung angeben: <meta charset =“utf-8“>.

- **Die eigentliche Seite des HTML-Dokuments:**
  Der sichtbare Bereich des HTML-Dokuments wird durch den Befehl <body> ein und durch </body> abgeschlossen. Alles, was zwischen diesen beiden Befehlen steht, wird im Browser angezeigt.

- Zu Beginn jeder HTML-Seite muss der Befehl **<html>** stehen, damit der Browser sie auch als solche interpretieren kann. Das Dokument endet immer mit </html>. Nur so weiß der Browser, dass das Ende des Dokuments erreicht ist.

- **Speichern der Datei:**
  Für jedes Webseiten-Projekt muss ein eigener Ordner als Stammordner angelegt werden (auch ROOT-Verzeichnis genannt). Der Ordner- und alle Dateinamen sollten kleingeschrieben werden, damit UNIX-Webserver keine Probleme bekommen, da sie zwischen Groß- und Kleinschreibung unterscheiden. In diesen Ordner werden alle Dateien, also auch alle Bilddateien und CSS-Dateien abgelegt, die für das Webseiten-Projekt benötigt werden.
  Die Datei sollte immer mit der **Dateiendung** .html (oder .htm) gespeichert werden. Der Dateiname sollte stets aussagekräftig gewählt werden. Für die Einstiegsseite haben sich die beiden Namen index.html und welcome.html im Internet durchgesetzt.

- **Aktualisierung der Browser-Ansicht:**
  Mit der Funktionstaste F5, der Tastenkombination Strg und R kannst du die Webseite-Ansicht im Browser aktualisieren, so dass du sehen kannst, wie sich die Änderungen des Quellcodes auf die Optik der Webseite ausgewirkt haben.

Die ersten Befehle

A) *Überschriften und Absätze*

<h1> heading . Überschrift **</h1>** größtmögliche Überschrift

Es gibt insgesamt sechs verschiedene Ebenen zur Textstrukturierung. **<h6>** ist demnach die kleinstmögliche Überschrift Überschriften müssen immer durch einen abschließend Tag (der mit dem Slash) beendet werden, sonst wird der gesamte Text zur Überschrift. Die wichtigste Überschrift der Webseite ist die Hauptüberschrift, sie muss mit dem h1-Element versehen werden und darf nur einmal verwendet werden. Die Darstellung der Überschriften unterscheidet sich je

nach Browser. Die Überschriften h1 und h2 werden von Suchmaschinen ausgewertet, so dass deine Benennung entsprechend passend gewählt werden sollte.

** ** Paragraph leitet einen neuen Absatz ein, der abschließende Tag kann entfallen.

**
** Break fügt einen Zeilenumbruch ein, auch hier entfällt der abschließende Tag, da es sich um ein leeres Element handelt. Leere Elemente zeichnen keinen Inhalt aus, sondern dienen dem Browser lediglich als Hinweise.

B) *Ausrichten von Texten*

Ob ein Text rechtsbündig, linksbündig, zentriert oder im Blocksatz angezeigt wird kann mit dem folgenden Zusatz festgelegt werden.

align=center (Ausrichtung, CENTER . zentriert)

align=right (rechtsbündig)

align=justify (Blocksatz)

Gibt man nichts an, so wird der Text linksbündig angezeigt, so dass dieser Befehl entfallen kann.

Beispiele:

<h2 align=center> Überschrift des Unterkapitels ist zentriert </h2>

 Der Absatz wird als Blocksatz dargestellt.

C) *Trennlinien*

Trennlinien untergliedern den Text oder setzen optische Akzente.

<hr> horizontal rule . Querlinie, ohne abschließenden Tag

Meist – je nach Browser - wird hier eine schattierte Linie dargestellt, stört dies, kann man den Schatten mit dem Zusatz NOSHADE entfernen. Man nennt diesen Zusatz auch *Attribut.*

Die Breite der Trennlinie lässt sich verändern.

<hr width=40% size=4> (width:.Breite, size: Dicke) erzeugt eine Linie, die 40% der Breite des Browserfensters einnimmt und eine Dicke von 4 Punkten besitzt.

<hat width=40> Dieser Befehl gibt an, dass die Trennlinie eine Breite von 40 Pixeln hat, egal wie breit das Browserfenster ist.

Auch Trennlinien lassen sich mit dem align-Tag ausrichten.

D) Bilder einfügen

Bilder werden mit dem im-Befehl eingefügt. Zusätzlich benötigt man die Quellenangabe, d. h. wo sich die Bilddatei befindet. Befindet sich das Bild, welches man einfügen will, im Unterordner Bilder des Stammordners, so wird mit dem Attribut Zorc und dem Attributwert „bilder/log.gif“ das Bild eingefügt. Der gesamte Befehl lautet:

<img src=“bilder/log.gif“> Man benötigt keinen End-Tag.

Auch das img-Element ist ein leeres Element, da es dem Browser nur mitteilt, wo welches Bild eingefügt werden soll.

E) *Verwendung von Farben*

Bei der Verwendung von Farben müssen mehrere Dinge beachtet werden, zunächst müssen die Farben zur inhaltlichen Darstellung passen. Dann ist es für die bessere Lesbarkeit wichtig, eine richtige Kombination aus Textfarbe und Hintergrundfarbe zu wählen. Wenn man nicht die Standard-Farbpalette verwendet kann es beim Betrachter zu Farbverfälschungen kommen, da die Farbdarstellung dann auch von der Leistungsfähigkeit der Grafikkarte und der jeweiligen Systemeinstellung abhängt.

Die Farbe kann durch die Eingabe in Hexadezimalform mit vorangestellter Raute (#) oder durch die Eingabe des englischen Farbnamens definiert werden.

- Schriftfarbe

Ein Text in roter Farbe
Der Text in grüner Farbe

-Hintergrundfarbe

Dieser Befehl bezieht sich auf das gesamte Dokument und wird daher im einleitenden body-Tag integriert.
bgcolor: background color
<body bgcolor=#0000FF> blauer Hintergrund

Einige Tipps zur Gestaltung einer Web-Seite

Gutes Layout ist keine Glückssache, sondern will gelernt sein.

1. Damit auch Besucher mit einer langsamen Internetverbindung deine Seite aufsuchen, solltest du die **Startseite möglichst klein** (Dateigröße) halten. (Bei den heutigen Geschwindigkeiten ist es nicht mehr ganz so entscheidend)

2. Wähle zu Beginn eine **passende Hintergrundfarbe** für die Einstiegsseite und lege die **Textfarbe** zunächst für das gesamte Dokument fest. Achte dabei auf eine kontrastreiche Darstellung. Ein Wechseln zwischen verschiedenen Hintergrundfarben sollte vermieden werden, damit der Besucher der Web-Seite nicht unnötig verwirrt wird. (siehe weiter hinten CSS)

3. Wenige, gut lesbare Schriftarten wählen!

4. Nutze die verschiedenen **Überschriften** und den Fettdruck, um die Informationen besser zu bündeln. Wähle treffende Überschriften, den sie entscheiden, ob der Text überhaupt gelesen wird oder nicht.

5. Die Verwendung von Kursivschrift und reinen Großbuchstaben ist kein guter Stil, beides ist am Bildschirm schlecht zu lesen.

6. **Vermeide Unterstreichungen!** Sie werden schnell mit Verweisen verwechselt.

7. Lockere deine Seite durch eine Reihe von **Absätzen** auf. Schreibe möglichst kurze Absätze und hebe sie deutlich voneinander ab.

8. Der Text sollte möglichst nicht über die gesamte Bildschirmgröße gehen, da der Leser so schnell beim Lesen in die falsche Zeile springt und den Überblick verliert.

9. Keine aufdringlichen bewegten Bildchen!

Aufgaben:

1. Erzeugt mit Hilfe des Editors (unter Zubehör) ein HTML-Dokument, indem du die oben beschriebene Grundstruktur verwendest.

2. Speichere anschließend das neue Dokument als HTML-Datei mit der Endung *.html* unter Eigene Dateien ab.
   Da dir dieses Dokument als Einstieg für deine Web-Seite dient, solltest du als Dateinamen *index.html* verwenden.

3. Lade die Datei in einem Browser und schau dir das Resultat an.

4. Öffne nun die Datei erneut im Editor.
   Lege eine Hintergrundfarbe und eine passende Textfarbe für die gesamte Startseite fest. Die Textfarbe lässt sich durch die zusätzliche Option TEXT im BODY-Tag definieren.

5. Nun wird es Zeit, dass sich deine Seite mit Inhalt füllt.
   a. Schreibe einen kurzen Text, indem du dich vorstellst. Die Zielgruppe für deine Seite besteht aus deinen Mitschüler des Informatikkurses.
   b. Wähle zusätzlich eine passende Überschrift und ersetze den Titel der Web-Seite durch eine aussagekräftige Bezeichnung.
   c. Verwende die beschriebenen Formatierungsbefehle zur besseren Strukturierung deiner Einstiegsseite. Achte dabei auf eine ausgewogene Verwendung der Befehle.
   d. Füge an geeigneter Stelle eine oder mehrere Trennlinien ein. Kannst du diese auch farbig darstellen?

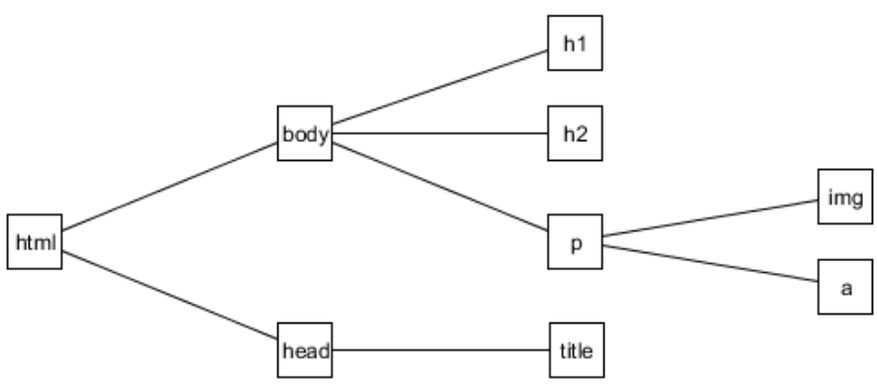
6. Um eine gute Lesbarkeit auch bei kleinen Schriftgrößen am Bildschirm zu gewährleisten, wurde die Schriftart *Verdana* für das Web entwickelt. Lege diese Schriftart nun als Standard für dein HTML-Dokument fest:
   <FONT FACE="VERDANA, ARIAL, COURIER">
   a. Kannst du dir vorstellen, warum noch weitere Schriftarten aufgeführt werden?
   b. Füge diese Befehlszeile nun an die richtige Stelle in deinem HTML-Dokument ein.
   c. Mit Hilfe der Option SIZE=2 kannst du die Schriftgröße festlegen. Füge Sie an geeigneter Stelle in die Befehlszeile ein. Probiere auch andere Schriftgrößen aus. Die Größe 1 solltest du jedoch wegen deiner schlechten Lesbarkeit vermeiden.

7. Speichere die Datei und rufe sie erneut im Browser auf. Was gefällt dir, was würdest du gerne noch ändern?

HTML mit CSS

Trennung von Inhalt und Layout

Webseiten im Internet haben eine Struktur, einen Inhalt und ein Layout. Die Struktur wird durch die HTML-Elemente beschrieben. So ergibt sich folgender Strukturbaum für die Grundstruktur jedes HTML-Dokuments:



Der Inhalt wird vom Autor geschrieben. Für ein einheitliches Layout werden alle Formatierungen wird über CSS festgelegt. Bisher haben wir das Layout einzeln festgelegt[4], dies wäre aber für größere Projekte sehr aufwändig.

Die Trennung von Inhalt und Layout hat folgende Vorteile:

- Die HTML-Dateien werden insgesamt kleiner.
- Globale Veränderungen sind mit minimalem Aufwand möglich.
- Die Inhaltsseiten und damit die ganze Homepage sind einfacher zu warten.
- Die Formatangaben werden in einer eigenen CSS-Datei gespeichert.
- Konsistentes Erscheinungsbild

Cascading Style Sheets

CSS steht für "Cascading Style Sheets" (Style Sheet = engl. Formatvorlagen) und dient dazu, Formatierungen, wie z.B. Schriftart oder -größe, sowie die Farbe, Rahmen, aber auch verschiedene Positionierungen vorzunehmen. Mit CSS kann eine ganze Homepage relativ einfach einheitlich gestaltet werden. Das Prinzip, welches es dabei zu beachten gilt, ist die strikte Trennung von der Strukturierung des Inhalts (mit HTML) und des Layouts der Seite (mit CSS). Das Layout einer Webseite legen wir in einer zentralen CSS-Datei fest, die in das HTML-Dokument eingebunden wird.

Einbinden der CSS-Datei in die HTML-Datei:

Im <head>-Element informiert man den Browser über das <link>-Element, dass er die Formatvorlagen aus der CSS-Datei benutzen soll:

```
<link rel="stylesheet" type="text/css" href="formate.css">
rel:          Verhältnis zwischen Webseite und Stylesheet: hier ständig verknüpft
type:         Art der Datei
formate.css   Dateiname
```

Die CSS-Datei:

Der Quelltext in einer solchen CSS-Datei könnte bspw. folgendermaßen aussehen:

```
h1 { text-align : right;}
h1 { margin-right: 20 px;}
h2 { color: red;}
```

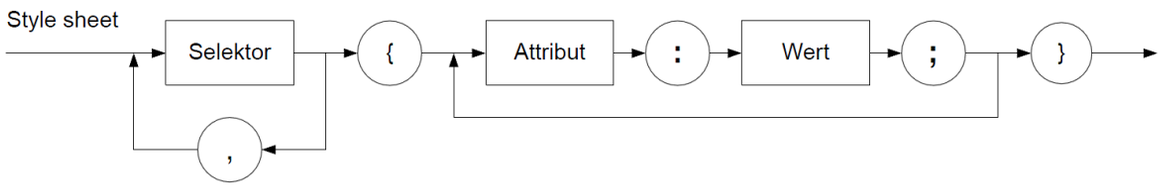
Die Syntax von CSS unterscheidet sich von HTML insofern, dass man keine Tags verwendet, sondern Regeln formuliert. Jede **Regel** ist nach demselben Muster aufgebaut: **Selektor { Eigenschaft : Wert ; }**

Als **Selektor** wird das bezeichnet, was vor den geschweiften Klammern steht. Ein Selektor wählt aus, wofür die folgenden Definitionen gelten sollen. Ein Selektor kann ein HTML-Element (Tag) sein (wie h1, p, body, …), es sind jedoch auch komplexere Selektoren möglich sowie mehrere, durch Kommata getrennte Selektoren:

```
h1, h2, h3 { text-align : center; }
```

[4] Dies gilt heute als veraltet, so dass z. B. die Hintergrundfarbe nicht mehr mit bgcolor im body-Element festgelegt wird, sondern in einer gesonderten CSS-Datei.

In diesem Beispiel werden alle Überschriften 1., 2. und 3. Ordnung zentriert auf der Seite angezeigt.
Die eigentlichen Definitionen zum Format stehen stets in **geschweiften Klammern** { und }. Sie bestehen darin, dass eine oder mehrere **CSS-Eigenschaften** notiert werden und einen **Wert** erhalten. Im obigen Beispiel werden die CSS-Eigenschaften text-align (Textausrichtung) und margin-right (Abstand rechts) verwendet. Zwischen Eigenschaft und Wertzuweisung muss stets ein **Doppelpunkt** stehen. Abgeschlossen wird eine Definition mit einem **Semikolon** (;). Sollen einem Selektor mehrere Definitionen zugewiesen werden, so listet man diese in den geschweiften Klammern auf, bzw. schreibt sie der besseren Übersichtlichkeit wegen untereinander.
Die Syntax eines Style sheet wird also durch dieses Syntaxdiagramm beschrieben:



Stylesheets kommentieren

Um bestimmte Formatierungen schnell wiederfinden zu können, empfiehlt es sich, CSS-Dateien mit Kommentaren zu versehen. Jeder Kommentar beginnt mit **/*** und endet mit ***/**.

Beispiel:

```
/* Formatierung für den body: Hintergrundfarbe, Schriftfarbe und Schriftfamilie */
Body {background-color: #F0E68C; color: #0000FF; font-family: Verdana, Arial, Helvetica;}
```

Stylesheets vererben

Wenn man CSS-Regeln für ein Element festlegt, so gilt diese Regel auch für alle Elemente, die in der Struktur untergeordnet sind. (siehe Abbildung S. 21). Wir definieren z. B. body {font-family: Arial}, so legt dieser Befehlt für die Webseite Arial als Standardschriftart fest. Wird für Überschriften, Absätze etc. nichts Anderes festgelegt, so gilt diese Schriftart für alle Elemente. Möchte man die größte Überschrift in einer anderen Schriftart darstellen, so legt man dies in einer zusätzlichen CSS-Regel fest: h1 {font-family: Verdana}.

Das class-Attribut als Selektor für Formatvorlagen

Manchmal ist es wünschenswert verschiedene Varianten von Formatvorlagen zu benutzen. Die Auswahl der gewünschten Variante erfolgt über das class-Attribut im HTML-Element:

```
<p class="normal">Textabsatz mit normaler Schriftgröße in Schwarz</p>
<p class="gross">Textabsatz mit größerer Schrift in Schwarz </p>
<p class="klein">Textabsatz mit kleinerer Schrift in Schwarz </p>
<h1 class="rot">rote Überschrift</h1>
```

Die Definition der Varianten erfolgt in der CSS-Datei.

```
.normal { font-size:100%; color:black; }
.gross { font-size:120%; color:black; }
.klein { font-size:80%; color:black; }
.rot { color: red; }
```

Relative und absolute Größenangaben:

Die relativen Größenangaben mittels Prozentwerten (siehe oben) sind günstiger als absolute Schriftgrößen, weil der Anwender dann die gewünschte Schriftgröße im Browser einstellen kann.
Relative Werte können nicht nur über Prozentwerte realisiert werden, sondern auch über die Maßeinheit Ems, z. B. h2 {font-size: 1.5em;}. Hiermit wird die 1,5-fache Größe des übergeordneten Elements (1 em = 100%) definiert. Ist z. B. festgelegt, dass die Schriftgröße der Webseite 12pt beträgt, so wird h2 mit 18pt dargestellt.
Möchte man dennoch absolute Werte eingeben, muss man Folgendes beachten:

- Kein Leerzeichen zwischen dem Wert und der Maßeinheit, z. B. 12pt
- Als Dezimaltrennzeichen wird der Punkt verwendet, z. B. 10.5pt
- Es gibt keine Standardmaßeinheit in CSS, wird keine Maßeinheit angegeben, so interpretieren die Browser die Angabe unterschiedlich.
- Mögliche Maßeinheiten: Pixel (px), Punkt (pt), Pica (pc), Inch (in), Millimeter und Zentimeter (mm, cm)

Formatvorlagen für Verweise

Verweise werden mit dem <a>-Element definiert. Browser zeigen normalerweise besuchte Verweise anders als noch nicht besuchte Verweise an. Hierfür gibt es spezielle Selektoren:

```
a:link { color: red } /* noch nicht besuchte Verweise */
a:visited { color: blue } /* besuchte Verweise */
a:hover { color: yellow } /* Maus auf dem Verweis */
```

CSS-Boxen

Moderne Webseiten werden mit Hilfe von CSS-Boxen strukturiert. Im Fenster eines Browsers werden die Inhalte der HTML-Elemente in rechteckigen Bereichen dargestellt. Im nachfolgenden Beispiel erkennt man drei solcher Rechtecke: Das Rechteck oben für den Titel, das Rechteck links für das Menü und das Rechteck für den Inhaltsbereich:

Im HTML-Dokument werden diese drei Bereiche mit dem -Element aufgeführt. Das <body>- Element enthält also genau drei -Elemente. Über deren Attribut „id“ wird der Namen des Bereichs angegeben:

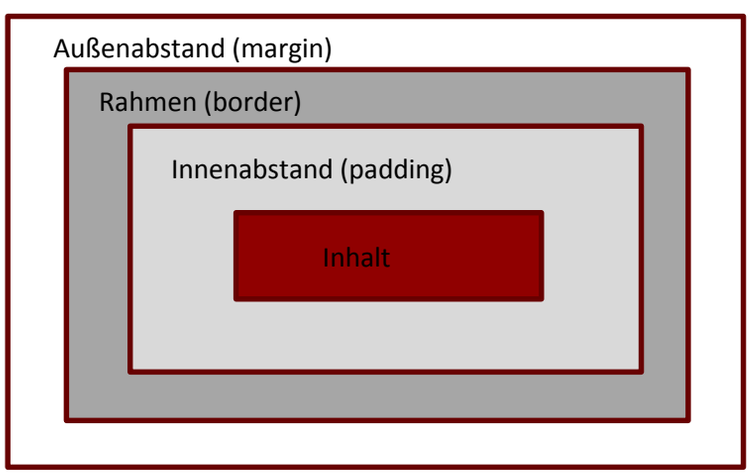
```html
<html>
 <head>
  <title>CSS-Boxen</title>
  <link rel="stylesheet" type="text/css" href="CSS-Boxen.css">
 </head>
 <body>
    <div id="Titel">
      <h1>Gustav Stresemann Gymnasium</h1>
    </div>
    <div id="Menue">
      <p>
      <a href="CSS-Boxen.html">Home</a>  <br>
      <a href="Aktuelles.html">Aktuelles</a><br>
      <a href="UnMa.html">Unterrichtsmaterial</a>
    </p>
    </div>
    <div id="Inhalt">
      <h2 class="erster">Informatik</h2>...
    </div>
 </body>
</html>
```

Das Layout der Bereiche – insbesondere Größe und Position– wird natürlich mittels CSS festgelegt. Dafür gibt es die #-Selektoren. Für das Beispiel sieht das wie folgt aus:

```css
#Titel{
  margin:20px 0px 20px 0px;
  padding:20px 0px 0px 20px;
  border-style:solid;
  border-color:black;
  background-color:#eee;
 }
#Menue{
  position:absolute;
  top:127px;
  left:10px;
  width:155px;
  padding:10px 10px 10px 10px;
  background-color:#eee;
  line-height:17px;
 }
#Inhalt{
  position:absolute;
  top:127px;
  left:185px;
  padding:0px 10px 10px 10px;
}
h1{
   font-size: xx-large;
   font-weight: bold;
   text-align: center;
   color: #800000;
  }
a:link{
   font-size: larger;
   }
a:visited {
 font-size: larger;
 color: #FF8000;
}
.erster{font-size:120%;
        color: #800000;
        }
```

(oben, rechts, unten, links)

Mit dem Attribut „position“ kann man angeben, dass eine CSS-Box, z. B. absolut positioniert wird. Mit den weiteren Attributen top, left, width und height kann man die Position und Größe in der Einheit px (engl. Pixel, Bildpunkte) festlegen. Für den Feinschliff einer CSS-Box gilt es zwischen margin (Abstand zu anderen CSS-Boxen), border und padding zu unterscheiden.



Das HTML-Element bildet den ersten Kasten. Direkt um den Inhalt herum bildet sich ein Innenabstand. Um diesen Innenabstand kommt ein Rahmen und direkt um den Rahmen herum bildet sich ein Außenabstand.

CSS-Eigenschaften und ihre Bedeutung

CSS-Eigenschaft	Beschreibung
font-family	Schriftart bzw. Schriftfamilie
font-size	Schriftgröße
font-style	Schriftschnitt (normal, italic(kursiv),..)
color	Schriftfarbe
font-weight	Fette Schrift (normal, bold)
text-align	Ausrichtung
line-height	Zeilenhöhe
background-color	Hintergrundfarbe
background-image	Hintergrundbild
padding	Innenabstände angeben, Reihenfolge: oben, rechts, unten, links
border-style	Rahmenart (none, solid, double, dotted, groove,…)
border-color	Rahmenfarbe
border-width	Rahmenstärke (medium, thin, thick,…)
border-radius	abgerundete Rahmenecken (5px) Sollen alle vier Ecken abgerundet werden, müssen 4 Werte eingegeben werden.
margin	Außenabstände

Formulare in HTML-Dokumenten

Möchte man, dass die Nutzer der Webseite den Kontakt zum Anbieter nicht nur über einen Email-Link aufbauen können, sondern auch so, dass bestimmte Angaben dem Anbieter gleich mit übermittelt werden, benötigt man Formulare. Beispiele hierfür sind Kataloganforderungen, Online-Bestellungen, Anmeldungen für geschützte Bereiche oder Suchfelder.

Formulare können an jeder Stelle eines HTML-Dokuments eingebunden werden. Sie können jedoch nicht ineinander verschachtelt werden.

Erzeugen von Formularen[5]

Zum Aufbau von Formularen benutzt man das HTML-Element <form>. Alle Elemente, die zu dem Formular gehören, müssen zwischen den beiden Tags <form> und </form> stehen.
Es kann notwendig sein, einem Formular einen eindeutigen Namen zu geben, damit man z. B. mit JavaScript darauf zurückgreifen kann. Hierzu verwendet man das Attribut id. Mit target legt man das Zielfenster für die Rückmeldung des Skripts fest. Das Attribut method bestimmt die Methode, mit der die Formulardaten versendet werden. Als mögliche Attributwerte kommen post und get in Frage. Bei der get-Methode werden die Daten des Formulars an die URL angehängt und zum Webserver übertragen. Es dürfen nicht zu viele Daten übertragen werden. Der Vorteil der get-Methode besteht darin, dass man auf diese Daten per Hyperlink verweisen kann und die Daten als Favorit im Browser gespeichert werden önnen. Die post-Methode wird normalerweise verwendet, um dem Server größere Datenmengen zur Verarbeitung zu übermitteln. Mit dem Attribut action gibt man an, welches Programm auf dem Webserver nach dem Abschicken der Formulardaten ausgeführt werden soll.

[5] Vgl. (Röhner G. , Unterrichtsskript: Grundkurs Informatik E1, 2010/2011), (York & Wegener, 2012)

Beispiel:
<form id="MeinFormular" action=mailto:name@mein.de method="post" target="Ziel">

Eingabefelder:

Mit <input ... > können verschiedenartige Eingabefelder erzeugt werden. Die Eingabeart wird durch das Attribut type festgelegt, die Größe des Eingabefeldes durch size.

- einfache rechteckige Eingabefelder (type="text")
- quadratische Mehrfach-Auswahlfelder (type= "checkbox")
- kreisförmige Alternativ-Auswahlfelder (type=" radio")
- Schalterfelder: Abschicken und Verwerfen (type="submit" und type="reset")
- Einzeiliges Eingabefeld mit nicht sichtbaren Zeichen (type="password")[6]

Auswahlfelder:

Bei Mehrfach- und Alternativ-Auswahlfeldern werden die einzelnen Werte mit dem value-Attribut angegeben. Jedes Eingabefeld erhält über das Attribut name einen Namen, um beim Abschicken des ausgefüllten Formulars die Eingaben den jeweiligen Formularfeldern zuordnen zu können.
<select> benutzt man zum Aufbau von Auswahl-Menüs, wie im Beispiel bei der Zuggattung. Jede einzelne Auswahlmöglichkeit wird über eine <option>-Kennung definiert. <textarea> verwendet man für mehrzeilige Eingabefelder.

Beispiel:

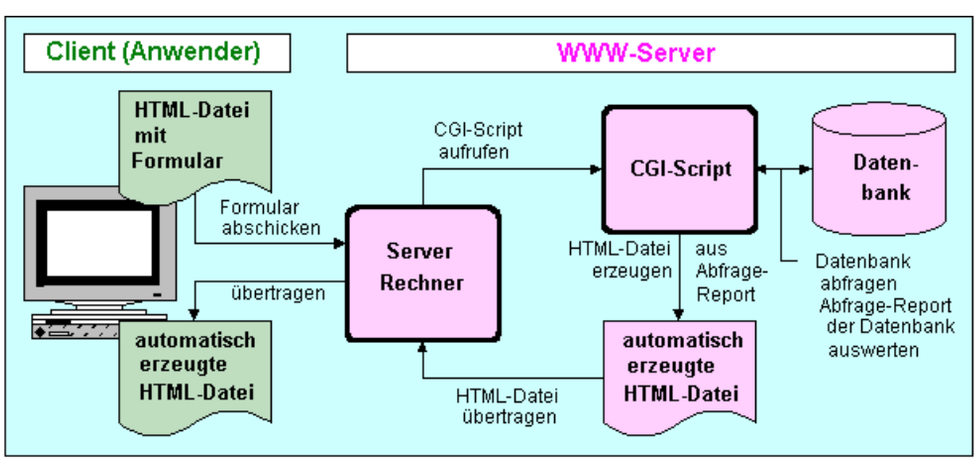
[6] In HTML5 kommen neue Eingabefelder hinzu.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
  <title>Reif für die Ferien?</title>
  <meta name="author" content="Alexandra Horn">
</head>
<body>
  <h1>Ferienhaussuche</h1>
  <form action="mailto:cahorn@web.de" method="post">
    <p><b>Ihre Email-Adresse:</b>  <br clear="all"/>
      <input name="Email" size="34">
    </p>
    <p><b>Sucheingabe für das Ferienhaus:</b></p>
    <table>
      <tr><td>Land:</td> <td><input type="text" name="Land" size="20"></td></tr>
      <tr><td>Region:</td> <td><input type="text" name="Region" size="20"></td></tr>
      <tr><td>Anreisedatum:</td> <td><input type="text" name="Datum" size="8"></td></tr>
      <tr><td>Aufenthaltsdauer in Tagen:</td> <td><input type="text" name="Land" size="3"></td></tr>
    </table>
    <p><b>Ausstattungsmerkmal:</b><br clear="all"/>
      <input type="checkbox" name="Ausstattung" value="Swimmingpool">Swimmingpool<br clear="all"/>
      <input type="checkbox" name="Ausstattung" value="Sauna">Sauna<br clear="all"/>
      <input type="checkbox" name="Ausstattung" value="TV">TV<br clear="all"/>
      <input type="checkbox" name="Ausstattung" value="Badezimmer">zweites Badezimmer<br clear="all"/>
    </p>
    <p><b>Hauskategorie:</b>
      <select name="Hauskategorie">
        <option>2-Sterne-Haus</option>
        <option>3-Sterne-Haus</option>
        <option>4-Sterne-Haus</option>
        <option>5-Sterne-Haus</option>
        <option>Premium-Haus</option>
      </select>
    </p>
    <p><b>Besitzen Sie eine Kundenkarte?</b><br clear="all"/>
      <input type="radio" name="Kundenkarte" value="Ja">Ja
      <input type="radio" name="Kundenkarte" value="Nein">Nein
    </p>
    <p>
      <input type="submit" value="Abschicken">
      <input type="reset" value="Verwerfen">
    </p>
  </form>
</body>
</html>
```

Formulare verschicken und auswerten

Die im Formular eingegebenen Daten müssen an den Empfänger verschickt werden. Im einfachsten Fall geschieht das als E-Mail. Dazu gibt man im <form>-Element im Attribut action die E-Mail-Adresse des Empfängers an und setzt das Attribut method auf post. Über den Abschicken-Schalter wird dann automatisch eine E-Mail mit den eingegebenen Daten erzeugt und verschickt.

Für eine automatische Reaktion auf ankommende Formulardaten muss der Internet-Provider eine passende Skript-Schnittstelle (CGI, PHP, ASP,...) zur Verfügung stellen. Das ausgefüllte Formular wird über das Skript ausgewertet und ein Antwort-Dokument generiert, das an den Absender des Formulars zurückgeschickt wird. Diese Technik wird beispielsweise bei den Suchmaschinen benutzt.



Eine Webseite im Internet veröffentlichen

Nun ist die Webseite erstellt, aber wie kann man sie nun im Internet veröffentlichen? Ein wichtiger Hinweise: Wer etwas im Internet veröffentlicht, muss auch sämtliche Rechte an den Inhalten besitzen. Wenn man also z. B. fremde Fotos verwendet, so ist das ein Verstoß gegen die Urheberrechte des Fotografen. Das kann sehr teuer werden. Auch alle anderen Rechte muss man im Blick haben: kein Verstoß gegen den Datenschutz, das Jugendschutzgesetz,…

Wie finden Suchmaschinen meine Seite?

Im head-Element kann man Suchmaschinen über das meta-Element Informationen zukommen lassen. Um eine Beschreibung im Suchergebnis erscheinen zu lassen, verwendet man das Attribut name="discription" (dt. Beschreibung) content="gewünschter Text". Der gewünschte Text sollte dabei in vollständigen Sätzen formuliert werden. Er sollte die wichtigsten Suchbegriffe enthalten und darf nicht länger als 255 Zeichen lang sein. Mit dem Attribut name="keywords" content="Schlüsselwörter" können Stichwörter angegeben werden, die das Angebot der Webseite möglichst passend beschreiben. Die einzelnen Schlüsselwörter müssen durch Kommas getrennt werden, wobei maximal 50 Wörter[7] eingegeben werden können. Als Schlüsselwörter kann man auch häufig vorkommende Tippfehler verwenden. Sowohl die Inhaltsbeschreibung als auch die Schlüsselwörter dürfen keine Umlaute und kein Eszett enthalten. Einige Suchmaschinen, z. B. Google ignorieren die Einträge in meta-Elementen. Sie legen mehr Wert auf die Wörter im Seitentitel und auf den Text im Body-Bereich.

Registrierung des Domainnamens

Es gibt zwei Arten, an eine Domain zu kommen. Entweder legt man sich eine eigene Domain zu, was für gewerbliche Webseiten auf jeden Fall zu empfehlen ist, oder man nutzt die kostenlose Möglichkeit des Providers, dort Speicherplatz zu nutzen. Der Nachteil ist hier, dass die URL länger ist, da der Name des Webservers mit angegeben werden muss. Der Speicherplatz ist nur begrenzt, so dass sich dies nur für Privatpersonen lohnt. Will man einen eigenen Domainnamen verwenden, so muss man zunächst klären, ob der Wunschname noch frei ist. Die Vergabe und die Registrierung von Domainnamen in Deutschland wird über www.denic.de geregelt. Wenn der Domainname dort noch nicht registriert ist, sollte man als nächstes prüfen, ob evtl. schon Dritte Rechte an dieser Domain erworben haben. Dies kann man über das internationale Markenregister (www.wipo.int) und durch die Eingabe in Suchmaschinen überprüfen.

Hat man nun einen Provider gefunden, der Speicherplatz (Webspace) kostenlos oder gegen Bezahlung zur Verfügung stellt, so müssen die Webseiten auf den Webserver geladen werden.

Webseiten hochladen – FTP

Um die Daten hochzuladen, benötigt man folgende Zugangsdaten:

- die Adresse des FTP-Servers
- den eigenen Benutzernamen und das Kennwort
- den Namen der Ordner auf dem Webserver, in den die Daten abgelegt werden sollen.

[7] Gängiger Richtwert

Um Daten auf einen Server im Internet zu laden, verwendet man das FTP (File Transfer Protocol), welches die systemunabhängige Datenübertragung ermöglicht. Auf dem eigenen PC muss ein FTP-Client installiert sein, mit dem die FTP-Verbindung hergestellt und die Datenübertragung verwaltet wird.

Recht und Gesetz im Internet

Wer eine Homepage im Internet veröffentlichen möchte, muss sich natürlich auch an rechtliche Vorgaben halten. Grundsätzlich ist das Internet kein gesetzesfreier Raum. Zwei Bereiche sollen hier noch mal herausgestellt werden: Das Urheberrecht und das Persönlichkeitsrecht.

Das Urheberrecht

„Das Urheberrecht schützt unsere selbst erschaffenen, persönlichen Werke, z. B. Bild, Computer-programme, Musikstücke oder Filme, als unser unmittelbares geistiges und künstlerisches Eigentum. Ein Urheber ist nach dem Urheberrechtsgesetz immer der persönliche Schöpfer des Werkes. Weder der künstlerische Wert noch das Alter des Autors spielen dabei eine Rolle. (…) Das Urheberrecht gilt heutzutage für fast alle Werke und Werkarten bis 70 Jahre nach dem Tod des Urhebers. Der Urheber hat das alleinige Verwertungsrecht. Er kann aber, oft gegen Bezahlung einer angemessenen Vergütung, anderen die Nutzung erlauben.“ (Kraus, Reder, & Rudel, 2012, S. 28)

Dies bedeutet natürlich nicht, dass du generell nie Werke anderer verwenden darfst. Wenn du fremde Texte, so wie ich es im oben stehenden Absatz getan habe, verwendest, so musst du sie als Zitat kennzeichnen. Tust du dies nicht, so handelt es sich um Diebstahl geistigen Eigentums, auch Plagiat genannt. Manchmal kann man auch Lizenzen erwerben, so dass man dann andere Werke veröffentlichen darf. Dies ist zum Beispiel bei Radiosendern der Fall. Sie besitzen eine Lizenz, so dass sie die Musik deiner Lieblingsband veröffentlichen dürfen. Von urheberrechtlich geschützten Programmen, Videos oder CDs darfst du dir eine private Sicherheitskopie machen, wenn du dazu keine Kopierschutzmechanismen umgehen musst, die Sicherheitskopie vernichtest, falls du das Original verkaufst oder verschenkst, und du das Original legal erworben hast.

Allerdings kann jeder, der seine eigenen Texten, Bilder, Musikdateien oder Videodateien im Internet veröffentlichen will, die Rechte an seinen eigenen Werken lockern, damit sie von anderen Anwendern kopiert und weitergegeben werden. Diese freien Lizenzen nennt man Creative-Commons-Lizenzen (CC).

Der Comic auf der nächsten Seite ist mit einer solchen Lizenz versehen. Ich kann ihn hier daher abdrucken, wenn ich den Namen des Autors nenne und den Comic an sich nicht verändere. Näheres erfährst du im Comic.

(Nerdson & Neumann, 2014)

Persönlichkeitsrecht

Im Grundgesetz der Bundesrepublik Deutschland ist im Artikel 2 Absatz 1 festgehalten: „Jeder hat das Recht auf die freie Entfaltung seiner Persönlichkeit, soweit er nicht die Rechte anderer verletzt und nicht gegen die verfassungsmäßige Ordnung oder das Sittengesetz verstößt.“ (Bundesministerium für Justiz und Verbraucherschutz, 2014). Dieses Gesetz umfasst weitrechende Rechte, die leider sehr häufig beim Veröffentlichen im Internet nicht berücksichtigt werden.

Was wird geschützt?

- Die persönliche sowie die berufliche Ehre werden geschützt, dies bedeutet, dass niemand einen anderen beleidigen oder verleumden darf.
- Du hast das Recht an deinem eigenen Bild, Ton und Text, d. h.
    - Grundsätzlich darf niemand ein Foto, ein Video oder eine Audioaufnahme von einer anderen Person erstellen, wenn die abgebildete Person der Erstellung nicht zugestimmt hat.
    - Grundsätzlich darf niemand ein Foto, ein Video oder eine Audioaufnahme von einer anderen Person veröffentlichen, wenn die abgebildete Person der Veröffentlichung nicht zugestimmt hat. Ausgenommen sind hiervon Personen des öffentlichen Interesses.
- Du alleine darfst dich unter deinem Namen ausgeben. Kein anderer darf deinen Namen missbrauchen, z. B. in dem er sich als Du ausgibt.

Aufgaben:

1. Entscheide, ob die Personen sich korrekt verhalten haben.
    a. Simon ärgert sich immer, wenn er eine CD nicht mehr hören kann, weil seine kleine Schwester seine CDs überall herumliegen lässt. Die verkratzten CDs funktionieren dann nicht mehr. Daher hat sich Simon überlegt, dass er seine Lieblings-CDs brennen will, so dass er nur noch die Kopien verwendet und die Originale in einem hohen Schrank sicher von seiner kleinen Schwester versteckt.
    b. Anna hat sich ein neues Computerspiel gekauft. Da das Spiel keinen Kopierschutz hat, brennt sie eine DVD und schenkt sie dir.
    c. Auf seiner Homepage möchte Jan etwas über Vulkane veröffentlichen. Er hat einen schönen Artikel bei wikipedia gefunden. Darf er den Artikel auf seiner Homepage veröffentlichen?
    d. Hans chattet gerne, aber er möchte seinen eigenen Namen nicht im Internet verwenden, deshalb gibt er sich als sein Mitschüler aus.
    e. Vom letzten Sommerurlaub hat Maike ein kleines Video zusammengeschnitten und mit der Musik des Sommerhits unterlegt. Sie lädt es bei youtube hoch.
    f. Olaf fotografiert gerne. Er möchte die eigenen Fotos auf seiner Homepage veröffentlichen. Er erstellt verschiedene Bereiche. Seine Landschaftsfotos sind für jedermann sichtbar. In einem gesicherten Bereich lädt er die Schnappschüsse der Partys hoch.

2. Wer darf bei Kindern und Jugendlichen entscheiden, in welcher Form personenbezogene Daten gespeichert werden dürfen?

Gefahren aus dem Internet[8]

Arten von Malware

Viele bösartige Programme, die heute unsere Computer bedrohen, können nicht mehr unbedingt einer bestimmten Kategorie zugeordnet werden, sondern enthalten mehrere Komponenten. So ist es möglich, dass ein Adware Programm zusätzlich den Benutzer ausspioniert und somit auch ein Spyware Programm ist, oder dass ein Computerwurm auch einen Trojaner installiert und so weiter.

Trojaner

Trojaner sind mit über einer Million Varianten die mit Abstand am häufigsten auftretende Malware. Der Begriff stammt aus der griechischen Mythologie, als die Griechen die Stadt Troja belagerten. Nach über 10 Jahren ohne Erfolg bauten sie ein hölzernes Pferd, in dem sich einige Krieger versteckten. Die übrigen zogen sich auf deine Schiffe zurück und taten so, als würden sie die Belagerung aufgeben und absegeln. Die Trojaner betrachteten das Pferd als Kriegsbeute und zogen es in die Stadt. In der Nacht schlichen sich die versteckten Krieger aus dem Pferd, öffneten die Tore Trojas und ermöglichten so den inzwischen zurückgekehrten Griechen, in die Stadt einzudringen und die Trojaner zu besiegen.
Ein Trojaner bezeichnet deshalb ein bösartiges Programm, das sich als nützliches oder sogar notwendiges Programm tarnt, beispielsweise zum Abspielen eines Videos, aber in Realität eine “Hintertür” zum Internet auf unserem Computer öffnet, die einem Angreifer Zugriff auf ihn ermöglicht.

Würmer

Würmer sind Programme, die sich selbständig über das Internet verbreiten. Typischerweise nutzen sie dazu Schwachstellen von Betriebssystemen oder Programmen aus. Viele Würmer wurden so programmiert, dass sie sich nur verbreiten und der einzige Schaden darin besteht, dass sie das Internet “langsamer machen”. Es gibt aber auch Würmer, die beispielsweise ähnlich wie ein Trojaner auf einem infizierten Computer eine Hintertür öffnen oder E-Mails von unserem Computer verschicken und so weiter.

Viren

Oftmals verwendet man im Alltag den Begriff Computervirus synonym für jede Art bösartiger Programme. Ein Computervirus im eigentlichen Sinn ist ein Programm, das sich selbst reproduzieren kann und sich oftmals im Programmcode eines seriösen Programms (einem “Wirtprogramm”) einnistet und auch so übertragen wird. Der Schaden, den ein Virus anrichtet, reicht von harmlosen, eher lästigen Meldungen auf dem Bildschirm bis zum Löschen von wichtigen Dateien oder Systemabstürzen.

Rootkits

Rootkits bestehen aus einem oder mehreren Programmen, die sich sehr tief im Betriebssystem einnisten, beispielsweise als Gerätetreiber. Dadurch können sie wichtige Teile des Betriebssystems verändern oder in die Abläufe eingreifen, vielfach mit dem Ziel, deine Präsenz zu verbergen. So kann ein installierter Rootkit beispielsweise das Betriebssystem so manipulieren, dass er von einem Antivirenprogramm nicht gefunden wird, oder dass er unbemerkt aufs Internet zugreifen kann.

[8] Vgl. (Skrotzky, 2010, S. 15-17)

Spyware

Unter diesem Begriff werden Programme zusammengefasst, die ohne sein Wissen Informationen über einen Benutzer sammeln. Beispielsweise so genannte Keylogger, Programme, die aufzeichnen, welche Tasten ein Benutzer gedrückt hat. Auch könnte ein Spyware Programm aufzeichnen, welche Webseiten eine Benutzerin besucht und sie sogar auf ungewünschte Webseiten umlenken.

Adware

Adware bezeichnet Programme, die eine Benutzerin mit unerwünschter Werbung eindecken. Obwohl Adware an und für sich keinen Schaden anrichtet, ist sie für einen Benutzer lästig, weil plötzlich erscheinende Fenster und ähnliches die Arbeit am Computer erschweren.

Spam

Bei diesem Begriff handelt es sich nicht um Malware, vielmehr bezeichnet man damit E-Mails, die mehr oder weniger wahllos an viele Empfänger geschickt werden. Der Inhalt der E-Mails kann von Werbung für diverse Produkte, meist von zwielichtigen Anbietern, bis zur Verbreitung von Malware reichen. Gemäß Statistiken von Organisationen, die den Mailverkehr beobachten, macht Spam ca. 90% des gesamten E-Mail Verkehrs aus. Der Name Spam ist übrigens keine Abkürzung, sondern stammt aus einem Sketch der englischen Komikertruppe Monty Python aus dem Jahre 1970 und bezeichnet eine Art Dosenfleisch.

Auswirkungen

Wie bereits oben erwähnt, kann Malware auf unserem Computer dazu führen, dass wir in unserer Arbeit am Computer behindert werden, dass Dateien verändert oder gelöscht werden oder dass unsere Internetverbindung plötzlich sehr langsam wird. Ebenfalls erwähnt wurde, dass private Informationen von unserem Computer gesammelt und an einen Angreifer übermittelt werden können. Während diese Gefahren direkt die Benutzerin des Computers betreffen, können andere weiter reichende Auswirkungen haben. Sobald es einem Angreifer gelingt, ein bösartiges Programm auf unserem Computer zu platzieren, das eine Hintertür zum Internet öffnet, hat der Angreifer eine gewisse, wenn nicht gar die volle Kontrolle über unseren Computer. Eine mögliche Folge davon ist, dass er unseren Computer in ein von ihm kontrolliertes Botnet einbindet. Es können aber auch schwerwiegendere Folgen auftreten, beispielsweise, wenn ein Angreifer unseren Computer als Webserver für illegale Inhalte missbraucht.

Botnets

Wenn dein Computer einem Angreifer eine Hintertür öffnet, so kann er diese dazu nutzen, um einen so genannten Bot (Kurzversion für Software Robot) auf deinem Computer zu installieren. Dabei handelt es sich um kleine Programme, die eigenständig und automatisch im Hintergrund laufen und über die der Angreifer Deinen Computer nach Belieben steuern kann. Dein Computer wird so zu einem Zombie Computer. Das Ziel des Angreifers ist dabei in der Regel, so viele Computer wie möglich unter seine Kontrolle zu bringen. Der Angreifer, Bot Master, kann nun sein Botnet gezielt einsetzen. Er kann beispielsweise allen Bots befehlen, Spam E-Mails zu verschicken oder gleichzeitig alle Bots auf denselben Webserver zugreifen zu lassen und diesen so blockieren (Denial of Service Angriff). Botnets können einige Tausend bis mehrere Millionen Computer umfassen. Beispielsweise wurde Anfang 2010 in Spanien eine Hackergruppe verhaftet, die ein Botnet von 13 Millionen Computern primär für Angriffe auf große Unternehmen benutzt haben.

Aufgaben:

1. Daniel meint, dass es ihm egal sei, wenn er Malware auf seinem Computer hat. Er habe nichts darauf gespeichert, was für einen Angreifer interessant sein könnte. Deshalb gebe er sein Geld lieber für andere Dinge als Virenschutz und ähnliches aus. Welche Argumente könntest du anführen, um Daniel zu überzeugen, dass auch er seinen Computer schützen sollte?
2. Warum ist es heute oft schwierig, ein Malware Programm einer bestimmten Kategorie zuzuordnen?
3. Welche Anzeichen könnten darauf hindeuten, dass dein Computer von einer Malware infiziert wurde?

Gefahren beim E-Mail-Verkehr[9]

Wenn wir einen Brief erhalten, so interessiert uns in der Regel der Inhalt des Briefes und nicht der Umschlag. Dieser hat eigentlich nur den Zweck, dass der Brief bei uns ankommt (und dass er nicht von anderen Personen gelesen werden kann).
Ganz ähnlich ist die Situation bei einer E-Mail. Der "Umschlag", der eine E-Mail vom Absender zu den Adressaten verschickt, ist das **SMTP Protokoll**. Im Gegensatz zu einem Brief wird der Umschlag jedoch "weggeworfen", sobald die E-Mail im Postfach des Adressaten eingetroffen ist. Wenn wir E-Mails empfangen, erhalten wir also sozusagen den Brief ohne Umschlag.
Eine E-Mail besteht im Wesentlichen aus drei Teilen. Ein erster Teil, in dem verschiedene Informationen zur E-Mail stehen, beispielsweise der Absender, der oder die Adressaten, die Betreffzeile, Datum und Uhrzeit, von welchem Server die E-Mail geschickt wurde und so weiter. Der zweite Teil enthält den Text der E-Mail und der dritte Teil die Anhänge.
Jeder dieser drei Teile kann falsche oder gefährliche Inhalte enthalten.

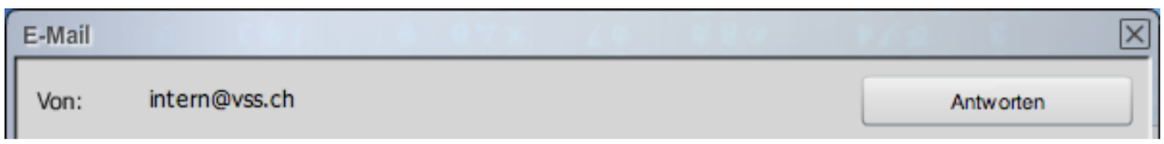
Informationsteil

Dieser Teil enthält verschiedene Informationen über die E-Mail. Die meisten dieser Informationen werden von einem Mail-Client normalerweise nicht angezeigt, weil sie für einen durchschnittlichen Benutzer wenig aussagekräftig sind. Für diejenigen, die sie trotzdem sehen wollen, bieten Mail-Clients eine entsprechende Anzeigemöglichkeit, die je nach Mail-Client Rohdaten, Kopfzeilen oder ähnlich heißt.
Die für uns wichtigsten Informationen dieses Teils bereitet ein Mail-Client für uns auf. Es sind der Absender, die Betreffzeile sowie die Adressaten der E-Mail. **Der Absender** einer E-Mail gibt uns eine erste Auskunft darüber, wie vorsichtig wir mit dem Inhalt der E-Mail umgehen sollten. Der Absender enthält die E-Mail- Adresse des Absenders und meistens auch noch dessen Namen. Wenn du einen Mail-Client einrichtest, so wirst du unter anderem aufgefordert, eine E-Mail-Antwortadresse und einen Namen einzugeben. Diese Angaben werden bei jeder E-Mail, die du verschickst, in den Informationsteil der E-Mail geschrieben. Obwohl es in der Regel sinnvoll ist, korrekte Angaben zu machen, ist es vom technischen Standpunkt her nicht zwingend.
Es können auch tatsächlich Situationen auftreten, in denen es zweckmäßiger ist, beispielsweise eine andere E-Mail Adresse anzugeben. Ein Beispiel dazu:

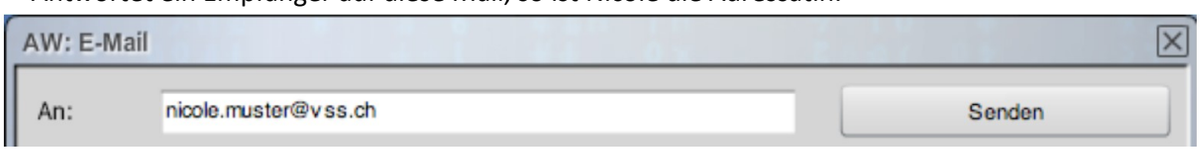
> Nicoles Schule bietet die Möglichkeit, über eine interne Webseite E-Mails an mehrere oder alle Schülerinnen und Schüler zu verschicken. Nicole möchte dieses Angebot nutzen und ruft die entsprechende Webseite auf. Nachdem sie die Nachricht geschrieben hat, schickt sie sie ab. Weil Nicole die Nachricht nicht von ihrem E-MailKonto aus verschickt hat, ist der Absender eine

[9] Vgl. (Skrotzky, 2010, S. 20ff)

allgemeine E-Mail-Adresse, in diesem Beispiel intern@vss, Antworten sollen aber an das E-Mail-Konto von Nicole gelangen. Beim Empfänger sieht die Nachricht so aus:



Antwortet ein Empfänger auf diese Mail, so ist Nicole die Adressatin.



Spammer nutzen diese Möglichkeiten gezielt, einerseits, weil sie damit den Eindruck erwecken können, die E-Mail stamme von einem seriösen Absender und andererseits, um die Herkunft der E-Mails zu verbergen.

Kannst du nun daraus folgern, dass E-Mails, deren Absender dir bekannt ist, grundsätzlich harmlos sind? Leider ist die Antwort: Nein.

Bei allgemein bekannten Absendern wie beispielsweise Facebook, MySpace oder msn ist diese Antwort eigentlich noch leicht einzusehen. Verwirrender ist es, wenn es sich um E-Mails aus deinem Bekanntenkreis handelt. Auch hier lohnt es sich, **die Betreffzeile** genauer anzuschauen. Wenn dir dein Mathematiklehrer eine E-Mail mit dem Betreff "Hey, schau mal was ich Cooles gefunden habe" schickt, oder deine beste Freundin dir Software zu Billigstpreisen verkaufen will, solltest du stutzig werden. *Aber wie kann das passieren? Wollen dir deine Lehrer und Freunde plötzlich Viren unterjubeln? Woher kennt ein Übeltäter die Namen und E-Mail-Adressen deines Bekanntenkreises?*

Eine mögliche Ursache kann der sorglose Umgang mit den Adressaten von E-Mails sein. Wenn du eine E-Mail verfasst, so gibt es drei verschiedene Arten von Empfängern. Diejenigen, an die sich die E-Mail direkt richtet (An:), diejenigen, die eine Kopie zur Kenntnis bekommen (CC:) und diejenigen, die eine versteckte Kopie bekommen (BCC:). Übrigens: CC steht für Carbon Copy und BCC für Blind Carbon Copy.

Während die ersten beiden Arten nur den Empfänger informieren, ob die E-Mail an ihn gerichtet ist oder ob er sie einfach zur Kenntnisnahme erhält, so gibt es einen wesentlichen Unterschied zur dritten Art: BCC. Alle E-Mails, die an Empfänger, die unter An: oder CC: aufgelistet sind, enthalten alle Adressen dieser Empfänger. **E-Mails, die an BCC-Empfänger gehen, enthalten nur die Adressen, die unter An: aufgelistet sind.**

Stelle dir vor, dass eine E-Mail an alle Schülerinnen und Schüler des GSG verschickt wird und alle Empfänger in der An:-Zeile aufgeführt sind. Für die Empfänger hat eine solche E-Mail drei unschöne Konsequenzen:

- Jeder Empfänger kennt nun die E-Mail-Adresse der übrigen Empfänger. Dies kann aus der Sicht des Datenschutzes unter Umständen problematisch sein.
- Empfänger, die mit einem Webmailer arbeiten, müssen sich zuerst durch alle Empfänger scrollen, um zur eigentlichen Nachricht zu gelangen.
- Einige Würmer oder Trojaner durchsuchen auf einem infizierten Computer dessen Mailboxen nach E-Mail-Adressen und senden diese an die Übeltäter. Wenn nun der Computer eines einzigen Empfängers infiziert ist, kennt der Übeltäter nun alle E-Mail-Adressen der Empfänger und kann zusätzlich den Schluss daraus ziehen, dass sich diese untereinander vermutlich kennen.

Der Mailinhalt

Eine Frage, die immer wieder gestellt wird, ist, ob man nur beim Öffnen von Anhängen vorsichtig sein muss, oder ob ein Computer bereits infiziert werden kann, wenn man eine E-Mail nur zum Lesen öffnet.
Bis Ende der Neunziger Jahre war es tatsächlich so, dass ein Computer nicht infiziert werden konnte, solange man keine Anhänge öffnete. Dies lag daran, dass Mail-Clients den Inhalt einer Mail als reinen Text interpretierten. Mit dem Aufkommen des WWW wurden auch die Mail-Clients so erweitert, dass sie in der Lage waren, den Inhalt einer E-Mail ähnlich wie ein Webbrowser zu interpretieren. Damit wurde es möglich, E-Mails zu versenden, die beispielsweise einen klickbaren Link oder Bilder enthalten oder sogar versuchen, Programme auf dem Computer zu installieren. Diese Tatsache hat durchaus seine praktische Seite. Zum Beispiel erhältst du bei vielen Internetforen bei der Registrierung eine E-Mail mit einem Link. Erst wenn du die Webseite, die mit diesem Link aufgerufen wird, besucht hast, ist die Registrierung abgeschlossen.
Diese Erweiterung der Mail-Clients bringt leider auch Gefahren mit sich. Einerseits kann eine E-Mail Links enthalten, die zwar so erscheinen, als ob sie auf seriöse Webseiten verweisen, tatsächlich aber ein Programm herunterladen oder auf irgendeine Webseite führen, die jemandem gehört, der nichts Gutes im Schilde führt. Glücklicherweise hast du schon bevor du auf den Link klickst, die Möglichkeit, festzustellen, wohin der Link führt. Wenn du die Maus ohne zu klicken auf dem Link platzierst, zeigen die meisten E-Mail Clients an, wohin der Link führt.

Falls Dein Mail-Client das nicht tut, so gibt es andere Möglichkeiten festzustellen, wohin der Link führt. Du kannst beispielsweise mit einem Rechtsklick den Link kopieren und in einen Editor einfügen.
Besonders heimtückisch sind Links, die bei flüchtigem Hinsehen auf einen seriösen Absender hinweisen, wie beispielsweise
http://www.microsoft.com.unsicher.ch/downloads/ieupdate?id=34Fcd26drg53ds3
Hier erhält man den Eindruck, der Link führe zu Microsoft. Tatsächlich führt er aber zur Domain unsicher.ch, also zu einem ganz anderen Server.
Noch gefährlicher sind E-Mails, die Code enthalten, der versucht, auf deinem Computer ein Programm zu installieren. Da dieser Code ausgeführt wird, sobald du eine E-Mail zum Lesen öffnest, kann heute bereits das Anklicken einer E-Mail auf einem ungeschützten Computer dazu führen, dass dein Computer infiziert wird. Gegen solche E-Mails kannst du dich nur schützen, indem du sie entweder gar nicht erst öffnest und ungelesen löschst oder einen aktuellen Virenscanner installiert hast, der E-Mails vor dem Öffnen prüft. Es kann auch sein, dass du von deinem Computer beim Öffnen einer E-Mail gefragt wirst, ob du ein Programm (beispielsweise eine “fehlende” ActiveX-Komponente) installieren möchtest. Solche Fragen solltest du immer ablehnen.
Eine weitere Variante von gefährlichen E-Mails sind solche, die dich auffordern, gewisse Informationen wie Benutzernamen, Passwörter oder sogar Kreditkartennummern in einer Antwort-Mail mitzuteilen. Solche E-Mails erwecken in der Regel den Eindruck, als würden sie von einer echten Organisation, beispielsweise einer Online Firma oder einem sozialen Netzwerk (Facebook etc.) stammen. Meist enthält der Mailinhalt die Schilderung irgendeines Problems und fordert dich auf, Deine Benutzerdaten zu übermitteln, damit das Problem gelöst werden kann. Solche Mails solltest du immer ignorieren und löschen.

Mailanhänge

Bei Mailanhängen ist die Art des Anhangs ein wichtiges Entscheidungskriterium dafür, ob ein Anhang gefährlich ist oder nicht. Handelt es sich um einen Anhang, der ausführbaren Code enthält, so kann dieser Code unter Umständen Schaden anrichten. Du kannst grundsätzlich zwischen vier Dateiarten unterscheiden.

- **Dateien, die nur Daten enthalten**. Solche Anhänge kannst du in der Regel ohne Gefahr öffnen. Dazu gehören beispielsweise Dateien mit den Erweiterungen “.jpg“, “.gif”, “.pdf”, “.txt”, “.mp3” und “.mov”.
- **Dateien, die Daten und ausführbaren Code enthalten.** Dazu gehören unter anderem Dokumente von Programmen, die Makrosprachen oder die Ausführung von Scripts unterstützen. Also beispielsweise Dateien mit den Erweiterungen “.doc”, “.xls” oder “.swf”.
- **Komprimierte Dateien.** Diese Dateien können einen beliebigen Inhalt haben. Möchte man einer E-Mail einen ganzen Ordner oder mehrere Dateien anhängen, so ist es oft einfacher, den Ordner oder die Dateien in einer einzigen Datei zusammenzufassen. Je nach Datei kann die Komprimierung auch eine erhebliche Verringerung der Datenmenge mit sich bringen. Da du nicht weißt, welche Dateien in einer komprimierten Datei enthalten sind, solltest du bei solchen Anhängen immer vorsichtig sein. Komprimierte Dateien verwenden oft die Erweiterung “.zip”.
- **Programme und Skripte.** Ist der Anhang ein Programm oder ein Skript, so ist es ziemlich offensichtlich, dass du hier vorsichtig sein musst. Grundsätzlich weißt du nicht, ob das Programm auch tatsächlich das bewirkt, was es zu machen vorgibt. Programme haben beispielsweise die Erweiterungen “.exe”, “.com”, “.vbs”, “.app”, “.bat” oder “.scr”.

Achte darauf, dass die **Erweiterung immer zuhinterst** steht. Ein Anhang mit dem Namen Bitte_Lesen.txt.exe ist kein Textdokument, sondern ein Programm.

Auch Dokumente, die nur Daten enthalten, beispielsweise Bilddateien, können unter Umständen gefährlich sein. Die Daten können so manipuliert sein, dass sie ein Fehlverhalten des Programms, das die Daten interpretiert, provozieren. Meist reagieren die Hersteller der Programme auf solche Gefahren, indem sie Updates zur Verfügung stellen.

Sicherheitsmaßnahmen beim E-Mail-Verkehr:

1. **Achte auf den Absender von E-Mails**. Wenn dir der Name unbekannt ist, so heißt das noch nicht unbedingt, dass die E-Mail gefährlich ist. Du solltest dann aber die Betreffzeile genau betrachten. Wenn deren Inhalt nichts mit dir zu tun hat, so stammt die E-Mail mit großer Wahrscheinlichkeit nicht von einem seriösen Absender.
2. Wenn du E-Mails an eine **größere Anzahl Empfänger** versenden willst, liste alle Empfänger unter BCC: auf. Weil du zum Versenden der E-Mail aber mindestens einen Empfänger unter An: angeben musst, ist eine empfehlenswerte Praxis, dort die eigene E-Mail-Adresse anzugeben. Vielleicht steht dir aber auch ein Multi-Mailing-System zur Verfügung. Solche Systeme sind normalerweise so konfiguriert, dass sie dieses Problem automatisch lösen.
3. Bevor du auf **einen Link** in einer E-Mail klickst, prüfe genau, wohin dich der Link führt. Dies gilt insbesondere, wenn dir der Absender der Email unbekannt ist, oder die E-Mail (scheinbar) von einem Online Dienst wie Facebook oder MySpace stammt. Verweist der Link auf eine Webseite, die nichts mit dem (scheinbaren) Absender zu tun hat, so solltest du den Link nicht anklicken.

4. Seriöse Unternehmen werden dich niemals auffordern, ihnen Passwörter, Kreditkartennummern oder ähnliches per E-Mail zuzusenden. Sollte ein Unternehmen tatsächlich Probleme mit deinem Benutzerkonto haben, so läuft die Problembehebung immer über gesicherte Webseiten.
5. Regelmäßiges Updaten verringert somit die Anfälligkeit der Programme, die du verwendest.
6. Grundsätzlich solltest du keinem Anhang blind vertrauen. Wenn du einen aktuellen Virenscanner benutzen und deine Software durch Installieren von Updates auf dem neusten Stand hältst, so kannst du Dokumente, die nur Daten enthalten in der Regel ohne Bedenken öffnen, insbesondere wenn sie von einem bekannten Absender stammen. Besondere Vorsicht ist geboten, wenn dir der Absender unbekannt und der Anhang ein Dokument ist, das Code enthalten kann. Die Wahrscheinlichkeit ist in einem solchen Fall groß, dass der Anhang gefährlich ist.

Aufgaben:

1. Kommentiere die folgenden Aussagen.
    a. Linda sagt: “Ich halte das Antivirenprogramm auf meinem Computer immer auf dem neusten Stand, also kann ich jedes E-Mail und jeden Anhang auch ohne Bedenken öffnen.”
    b. Marco meint: “Ich will auf keinen Fall ein gefährliches Programm auf meinem Computer. Deshalb lösche ich sofort jede E-Mail, deren Absender ich nicht kenne.”
2. Du arbeitest mit einem Klassenkameraden an einem Vortrag und der schickt dir seine Unterlagen per E-Mail in Form einer angehängten Worddatei. Kannst du die Datei bedenkenlos öffnen? Begründe dein Vorgehen.
3. Beim Öffnen einer E-Mail meldet dein Computer, dass ihm eine Softwarekomponente fehlt, um die Nachricht anzeigen zu können. Er fragt dich, ob du die Komponente installieren willst. Klickest du auf “Ok” oder auf “Abbrechen”? Begründe deine Wahl.

Gefahren beim Surfen auf Webseiten

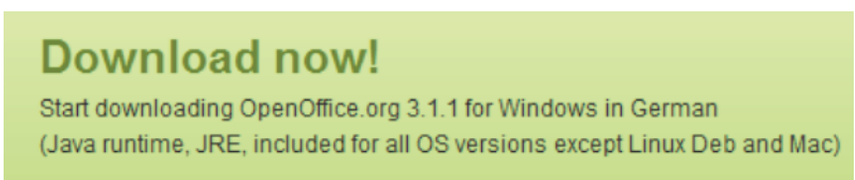
Mitteilungsbedürftige Browser[10]

Nicht nur seriöse Unternehmen analysieren das Surfverhalten der Internetnutzer. Auch für einen potenziellen Angreifer ist es interessant, möglichst viel über den Computer eines Benutzers zu erfahren und zu wissen, wie sich ein Benutzer im Internet verhält, wofür er es benutzt und welche Seiten er in letzter Zeit besucht hat.

Dass unser Computer so viele Informationen preisgibt, ist vielfach beabsichtigt. Ein Betreiber einer Website kann dadurch die Darstellung und den Inhalt der Seiten für den entsprechenden Browser und das verwendete Betriebssystem optimieren und so die Benutzung seiner Seite vereinfachen und allenfalls sinnvolle Fehlermeldungen und Hinweise zu deren Behebung anzeigen.

Positives Beispiel:

Der Server erkennt, welches Betriebssystem und welche Sprache wir verwenden und erleichtert uns so das Herunterladen der korrekten Programmversion.



Es ist durchaus überraschend, wie viele Informationen ein Betreiber einer Webseite über unseren Computer herausfinden kann. Das reicht von relativ offensichtlichen Informationen wie beispielsweise Art

[10] Vgl. (Skrotzky, 2010, S. 35ff)

und Version des Browsers und installierte Plugins bis zu installierten Schriften und kürzlich besuchten Webseiten.
Diese Informationen können natürlich auch missbraucht werden. Wenn ein Server beispielsweise erkennt, dass ein Benutzer mit einer veralteten Version einer bestimmten Komponente arbeitet, die bekannte Sicherheitsmängel aufweist, so kann er dies ausnutzen.
Auch uns gegenüber zeigen sich Browser durchaus kommunikativ und fragen uns, was wir mit einer Datei, die heruntergeladen wird, anfangen wollen oder ob blockierte Elemente zugelassen werden sollen. Dabei kann das Mitteilungsbedürfnis eines Browsers so groß werden, dass man sich als Benutzerin darüber ärgert und die Aktionen, ohne die Texte durchzulesen, akzeptiert. Ein solches Verhalten kann dadurch unterstützt werden, dass wir auch Warnungen erhalten, wenn wir seriöse Komponenten installieren. Einem Angreifer ist dieses Verhalten durchaus bewusst.

Schadenanrichtende Webseiten

Schnäppchenjäger

Vorsicht bei Schnäppchen! Im Internet gibt es viele Seiten, die kostenlose Downloads von Musik, Filmen, Spielen etc. anbieten. Die Motivationen, Produkte gratis zur Verfügung zu stellen, sind vielseitig. Als Internetnutzer haben wir uns an solche Angebote gewöhnt. Dass wir dabei oft nichts über den Anbieter wissen, kümmert die wenigsten Benutzer. Ein Angreifer kann diese Gutgläubigkeit ausnutzen, indem er Surfer mit einem Gratisangebot auf seine Seite lockt. Da nun die Seite mit dem Zweck besucht wird, Software herunterzuladen, werden auch die Dialoge, ob die Dateien heruntergeladen werden sollen, weniger kritisch durchlesen. Und schon ist die Schadsoftware vom Anwender selbst auf den Rechner heruntergeladen worden.

Soziale Netzwerke

Der virtuelle „Freundeskreis“ in sozialen Netzwerken wie Facebook ist bei den meisten Nutzern viel größer als dein realer Bekanntenkreis, dennoch wird den virtuellen Freunden viel Vertrauen entgegengebracht. Angreifer haben dadurch ein leichtes Spiel, sobald sie es geschafft haben, sich in einen Freundeskreis einzuschleichen. Nun kann er Links zu infizierten Seiten versenden, die mit einer relativ hohen Wahrscheinlichkeit geöffnet werden. Oder er nutzt die Informationen über Personen aus, um ein Vertrauensverhältnis aufzubauen, so dass er dann das Opfer zu bestimmten Aktionen bewegen kann oder an vertrauliche Informationen z. B. Zugangsdaten herankommen kann. Eine solche Vorgehensweise nennt man Social Engineering.

Manipulierte Webseiten

Eine ebenfalls äußerst beliebte Vorgehensweise von Angreifern ist, dass sie versuchen in seriöse Webseiten einzudringen und diese so abzuändern, dass beim Besuch der Seite entweder ein automatischer Download einer bösartigen Software ausgelöst wird, oder die Besucherin, ohne es zu merken, auf eine andere, vom Angreifer kontrollierte Seite umgelenkt wird. Diese Seite könnte beispielsweise mit der seriösen Anmeldeseite identisch aussehen und ein Angreifer könnte so die Zugangsdaten eines Benutzers erlangen. Es gibt mittlerweile Untergrundorganisationen, die Softwarepakete verkaufen, mit denen ungenügend geschützte Webserver auf einfache Weise infiziert werden können.

Böse Werbeüberraschung

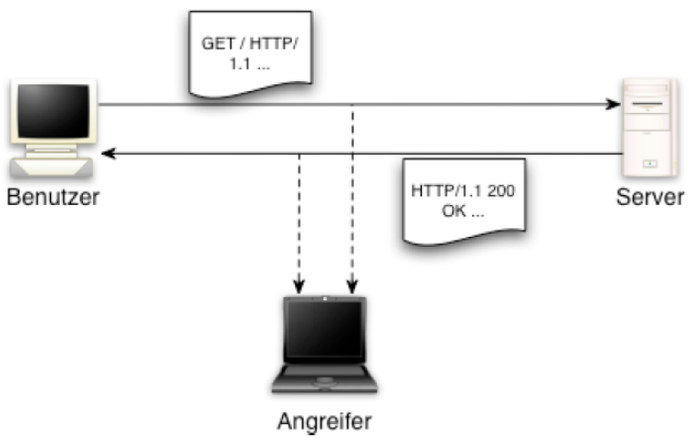
Werbung auf Webseiten ist für viele Betreiber eine nicht unwesentliche Einnahmequelle. Dabei darf ein Drittanbieter auf der Webseite des Betreibers Code platzieren, der die Werbung anzeigt. Normalerweise

handelt es sich bei diesen Drittanbietern um anerkannte Firmen, die auf Internetwerbung spezialisiert sind und denen der Betreiber vertraut. Es ist nun aber durchaus üblich, dass diese Firmen den Werbeplatz teilweise weitervermieten dürfen und so weiter. Das kann dazu führen, dass ein Angreifer die Gelegenheit erhält, seinen Code auf einer an und für sich seriösen Seite zu platzieren.

S wie Sicher

Wenn wir uns im Internet bewegen, wird in der Eingabezeile des Browsers jeweils die URL der entsprechenden Seite angezeigt. In der Regel beginnt die URL mit “http://”. Vielleicht hast du auch schon Webseiten besucht, die stattdessen mit **https://** beginnen. Das “s” steht für secure.

Normalerweise wird der Inhalt einer Webseite in einer leicht lesbaren Form übermittelt. Das gilt auch für die Daten, die wir über ein Formular (beispielsweise wenn wir uns irgendwo anmelden) zu einem Webserver zurückschicken. Somit ist es für jeden, der die Verbindung abhören kann, leicht möglich, herauszufinden, welche Informationen hin und her geschickt werden.



Wenn wir uns auf einer Webseite anmelden wollen, möchten wir natürlich nicht, dass unser Benutzername und vor allem unser Kennwort einfach so gelesen werden können. Seriöse Organisationen, auf deren Webseiten man sich anmelden muss, verwenden deshalb mindestens für die Anmeldung eine Verschlüsselung. Geschieht die Kommunikation zwischen Benutzerin und Webserver über eine verschlüsselte Verbindung, so wird dies in der Eingabezeile dadurch angezeigt, dass anstelle von “http://” “https://” steht. Außerdem wird im Browserfenster auch noch irgendwo ein Schloss oder ein Schlüssel angezeigt (wo, ist browserabhängig, meistens entweder zuoberst, zuunterst oder in der Eingabezeile selbst). Sollte ein Angreifer bei der Kommunikation mithören, so würde er nur unverständliche Zeichen empfangen.

Damit ein Webserver verschlüsselt kommunizieren kann, muss er ein Sicherheitszertifikat besitzen. Diese Zertifikate sind kostenpflichtig. Es gibt verschiedene Organisationen, die autorisiert sind, solche Zertifikate auszustellen. Bietet ein Webserver eine Verschlüsselung an, prüft der Browser der Benutzerin zunächst, ob es sich um ein autorisiertes Zertifikat handelt. Falls nicht, erscheint eine Warnung. Diese Warnung muss noch nicht unbedingt heißen, dass du im Begriff bist, auf eine Webseite mit bösartigem Code zuzugreifen. Grundsätzlich kann jeder, der einen Webserver betreibt, selbst ein (kostenloses) Zertifikat erzeugen. Du solltest aber trotzdem stutzig werden. Vor allem größere Organisation oder Online-Shops verfügen immer über autorisierte Zertifikate. Eine solche Warnung kann deshalb bedeuten, dass die Webseite, auf die du zugreifen möchtest, nicht seriös ist. Wenn dein Computer beispielsweise bereits infiziert ist, sodass er auf einen bösartigen DNS-Server zugreift, kann zwar die URL in der Eingabezeile korrekt sein, aber du landest trotzdem auf dem Server des Angreifers.

Gestohlene Identitäten

Viele Webseiten platzieren so genannte Cookies auf den Computern der Benutzer. Cookies sind für einen Computer unschädlich. Sie können verschiedene Informationen über einen Besucher einer Webseite

enthalten, beispielsweise wie oft er eine Webseite besucht hat, wann der letzte Besuch stattfand und so weiter. Cookies können auch zum Speichern von Benutzernamen verwendet werden, damit beim nächsten Besuch der Webseite ein Benutzer persönlich begrüßt werden kann oder bei der Anmeldung der Benutzername automatisch eingesetzt wird. Cookies haben immer eine bestimmte "Lebensdauer". Diese kann von wenigen Minuten bis zu mehreren Jahren reichen, oder davon abhängen, wie lange ein Browserfenster geöffnet ist. Webseiten, die es zulassen, dass Benutzer selbst Daten auf der Seite platzieren können, erlauben dies in der Regel nur registrierten und angemeldeten Benutzern. Ein Server muss sich merken können, ob sich ein Benutzer angemeldet hat oder nicht. Dies kann er beispielsweise tun, indem er nach der Anmeldung ein so genanntes Session-Cookie auf dem Computer des Benutzers platziert, das wieder gelöscht wird, wenn der Benutzer sich abmeldet oder die Seite schließt. Ein Angreifer kann versuchen, ein Session-Cookie einer Benutzerin zu stehlen und sich so als diese auszugeben. Ein solcher Angriff ist für einen Benutzer sehr schwierig zu entdecken, weil aus Sicht des Benutzers nichts Ungewöhnliches passiert. Auch Antivirenprogramme erkennen einen solchen Angriff nicht.

Aufgaben:

1. Was sollte man beim Surfen im Internet beachten? Erstelle eine Liste von Sicherheitsmaßnahmen!
2. Beim Aufrufen einer Webseite erscheint eine Meldung, dass die Webseite nicht vollständig angezeigt werden kann, weil auf deinem Computer zuerst eine entsprechende Softwarekomponente installiert werden muss. Wie reagierst du? Begründe deine Antwort.
3. Welches Ziel verfolgt ein Angreifer mit Social Engineering?
4. Weil ich kein Geld für teure Software habe, suche ich Webseiten, die gratis Programme anbieten und lade die Software von dort herunter. Dabei achte ich darauf, dass wirklich nur die Software installiert wird, die ich heruntergeladen habe. Wie beurteilst du dieses Vorgehen?
5. Wenn ich mich bei einer Webseite registrieren/anmelden muss, mache ich das nur, wenn die entsprechende Seite die Daten verschlüsselt überträgt. Das heißt, ich mache das nur, wenn die URL in der Eingabezeile mit "https://" beginnt. Würdest du dieses Vorgehen unterstützen? Begründe!
6. Starte den Browser, den du üblicherweise benutzt, und schaue nach, welche Programme Cookies auf deinem Computer platziert haben. (in der Schule und zuhause).
   Hinweis: Das Anzeigen von Cookies ist nicht bei allen Browsern gleich einfach. Beim Firefox sieht man die Cookies nur, wenn über EXTRAS->EINSTELLUNGEN->DATENSCHUTZ auf „einzelne Cookies löschen" geklickt wird. Beim IE muss man auf EXTRAS->INTERNETOPTIONEN->ALLGEMEIN im Abschnitt BROWSERVERLAUF auf „Einstellungen" und anschließend auf „Dateien anzeigen" klicken. Was stellst du fest?
7. Schaue im Browser, den du üblicherweise verwendest nach, welche Webseiten im Verlauf gespeichert sind. Temporäre Internetdateien und Internet Caches sind nicht immer leicht zu finden. Viele Browser verwenden dazu einen versteckten Ordner im Verzeichnis des Benutzers. Im Internet Explorer kannst du die temporären Internetdateien über Extras-> Internetoptionen -> Allgemein im Abschnitt Browserverlauf mit Klicken auf Einstellungen und anschließend auf Dateien anzeigen, sichtbar machen.

Grundlagen der Programmierung

Konsolenprogramme in Java

Kurzer Überblick über die Entwicklung der Programmiersprachen

Im Laufe der Geschichte der Informatik gab es Hunderte von Programmiersprachen, die sich grundsätzlich jedoch in vier Programmierparadigmen zusammenfassen lassen:

1. die prozedurale Programmierung (z. B. Pascal)
2. die funktionale Programmierung
3. die wissensbasierte Programmierung (z. B. Prolog)
4. die objektorientierte Programmierung (z. B. Delphi, Java)

Die Anforderungen an Computerprogramme haben sich im Laufe der Zeit verändert. Zu Beginn standen eine gute Speicher- und Zeiteffizienz im Vordergrund. Die Entwicklung besserer Hardware ließ dann auch komplexere Programme zu, die jedoch aufgrund der Programmiermethoden immer fehlerhafter wurden. Die Softwarekrise 1965 führte schließlich dazu, dass die **Qualitätsmerkmale** *Zuverlässigkeit, Korrektheit und Robustheit* verstärkt in den Fokus gerückt wurden. Diese Qualitätsmerkmale sollten vor allem mit der strukturierten Programmierung erreicht werden, die auf der Basis des **Top-Down-Prinzips** und der Methode der **schrittweisen Verfeinerung** agiert. Schließlich stellte man Anfang der 70er Jahre fest, dass die *Wartbarkeit* als Qualitätsmerkmal ergänzt werden musste. Seit Beginn der neunziger Jahre hat sich die objektorientierte Programmierung (OOP) in der Praxis mehr und mehr durchgesetzt. Vor allem in Zeiten des Internets und der Verbreitung von Software in allen Lebensbereichen bietet die OOP große **Vorteile**:

- **Wiederverwendbarkeit von schon programmierten Elementen**
- **Aufteilung in überschaubare Einzelteile**
- **Erweiterung durch Schnittstellen**

Das objektorientierte Paradigma teilt sich in zwei miteinander verzahnte Bereiche auf: die objektorientierte Modellierung und die eigentliche objektorientierte Programmierung. Hierauf werden wir in der Q-Phase näher eingehen.

Java gibt es seit 1995. Von Beginn an war neben der Objektorientiertheit ein weiterer wichtiger Faktor für die Verbreitung dieser Programmiersprache, dass sie portabel ist. Das bedeutet, dass in Java geschriebene Programme auf den unterschiedlichsten Plattformen, z. B. Windows, Linux, Mac etc. laufen.

Wie lernt man programmieren?

Stell dir vor, du möchtest Klavierspielen lernen. Was tust du? Du gehst regelmäßig zum Klavierunterricht, deine Klavierlehrerin zeigt dir, wie man Klavier spielt. Sie ist gut, sie kann ganz tolle Stücke spielen und manchmal darfst du auch ein Übungsstück spielen. Du gehst nach Hause und nach einem halben Jahr kannst du perfekt Klavierspielen.

Nun gut, so einfach ist es eben nicht. Erstens dauert es wesentlich länger, bis du wirklich gut spielen kannst, und zweitens musst du regelmäßig auch zu Hause üben, üben und nochmals üben.

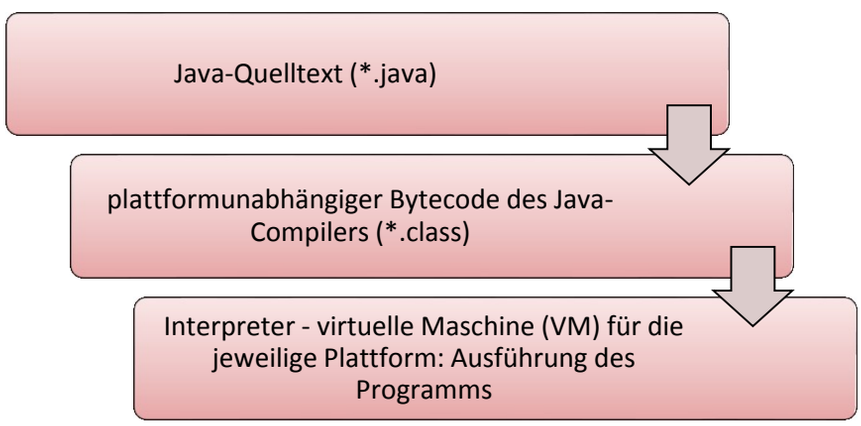
Ähnlich ist es auch mit dem Programmieren. Es kann dir keiner beibringen, indem du anderen zuguckst und zuhörst. **Du musst dich aktiv am Lernprozess beteiligen. Programmieren lernen erfordert viele Übungen, Einsatz und Durchhaltevermögen.** Es kommt nicht immer darauf an, eine Übungsaufgabe perfekt abzugeben, sondern sich mit der gestellten Aufgabe ernsthaft auseinanderzusetzen. Dazu muss man sich mit der Aufgabe gedanklich beschäftigen, sie mit dem Gehörten und Gesehenen aus dem Unterricht verknüpfen, üben, evtl. noch mal etwas nachlesen, weiter üben usw. Hilfreich ist auch, wenn man die eigene Lösung mit der Musterlösung am Ende vergleicht, um sich Tricks für das weitere Programmieren abzugucken.

Vom Quellcode zum Programm

Zunächst schreibt der Programmierer sein Java-Programm als Quellcode, dabei muss er sich an bestimmte Regeln halten. Hierzu genügt ein ganz einfacher Texteditor. Wir werden allerdings den Java-Editor verwenden. Der Java-Quellcode wird immer mit der Dateiendung „java“ abgespeichert.

Anschließend muss der Java-Quellcode kompiliert werden. Dazu wird der Quellcode an den Compiler javac übergeben. Dieser überprüft den Quellcode zunächst auf Fehler und übersetzt den fehlerfreien Quellcode jetzt in Java-Bytecode. Die Dateiendung lautet „class“. Dieser Java-Bytecode ist eine Mischung aus Maschinensprache[11] und menschlicher Sprache. Dieser Java-Bytecode ist der Schlüssel für die Portabilität von Java. Er ist plattformunabhängig. Diese Datei wird vom Compiler automatisch in demselben Ordner wie die dazugehörige java-Datei abgespeichert. Sie kann nicht mehr mit dem Editor gelesen werden.

Nur mit Hilfe der Javamaschine (Java virtual Machine JVM) kann dieser Bytecode nun auf verschiedenen Plattformen ausgeführt werden. Die Javamaschine, d. h. der Interpreter, ist plattformabhängig. Bei der Javamaschine handelt es sich nicht um eine physische Maschine sondern um eine virtuelle. Es ist also eine Software, die den Bytecode liest und ausführt.



Was versteht man unter einem Programm?

Ein Programm ist eine schrittweise Beschreibung eines Vorgangs, wie zum Beispiel ein Kochrezept, eine IKEA-Aufbauanleitung oder ein Waschmaschinenprogramm. Je besser eine solche Beschreibung ist, desto besser das Ergebnis.

[11] Maschinensprache besteht nur noch aus Nullen und Einsen und kann vom Computer direkt verstanden werden.

Eine schrittweise Beschreibung bezeichnet man in der Informatik als Algorithmus. Computerprogramme folgen dabei dem **EVA-Prinzip**. Nach der **Eingabe** von Daten, werden diese **verarbeitet** und anschließend erfolgt die **Ausgabe** der Ergebnisse.

Definition des Algorithmus:

„Unter einem Algorithmus versteht man eine **eindeutig und endlich** formulierte Verarbeitungsvorschrift, deren Operationen tatsächlich **ausgeführt** werden können. Für die Beschreibung kann eine natürliche oder auch eine künstliche Sprache (Programmiersprache) verwendet werden. Als Operationen können dabei, z. B. Methodenaufrufe dienen. Diese Methoden kann man jeweils wiederum als Beschreibung eines Algorithmus betrachten.“ (Hubwieser, Spohrer, Steinert, & Voß, Informatik 3; Lehrwerk für Gymnasien, 2008, S. 19)

Abb. 1 (Magenheim J. u., 2009, S. 47)

Von jedem Algorithmus werden bestimmte Fakten erwartet:

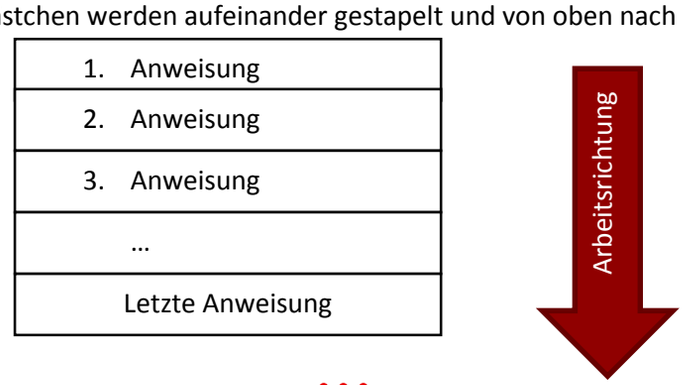
- Terminiertheit: Der Algorithmus soll irgendwann fertig werden, d. h. er soll enden.
- Determiniertheit: Bei gleicher Eingabe soll auch jedes Mal die gleiche Ausgabe erfolgen.
- Effizienz: Der Algorithmus soll möglichst schnell sein.
- Korrektheit: Das Ergebnis soll stimmen.

Grafische Darstellung von Algorithmen

Algorithmen kann man sprachlich beschreiben, häufig ist es aber einfacher und leichter zu überblicken, wenn man verschiedene Diagrammtypen verwendet, die die richtige Abfolge von notwendigen Schritten deutlich machen.

Struktogramme

Struktogramme können verwendet werden, ohne sich um die genauen Einzelheiten der benutzten Programmiersprache Gedanken zu machen. Man kann mit Struktogrammen Vorüberlegungen schematisch aufschreiben. Meist verwendet man ein Struktogramm nicht für ein vollständiges Programm, sondern nur für einzelne Programmteile. Einzelne Anweisungen werden jeweils in ein Kästchen geschrieben. Diese Kästchen werden aufeinander gestapelt und von oben nach unten abgearbeitet.

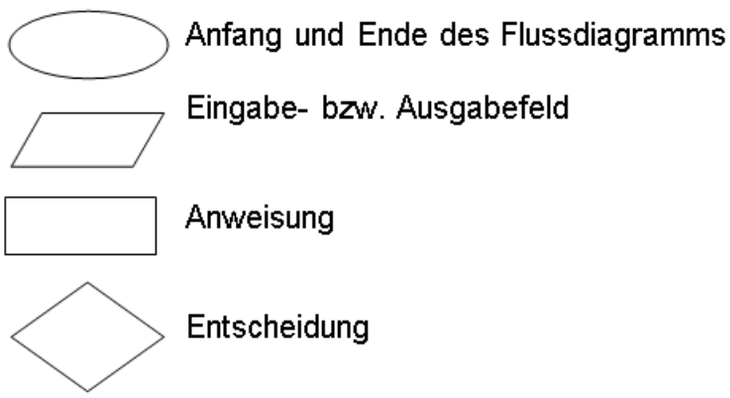


Mithilfe von Struktogrammen lassen sich die Strukturen von Algorithmen besonders übersichtlich darstellen. Eine Folge von Verarbeitungsschritten, die in einer bestimmten Reihenfolge abgearbeitet werden muss, heißt **Sequenz**. Das Struktogramm beschreibt die Abfolge beim Betreten einer Wohnung.

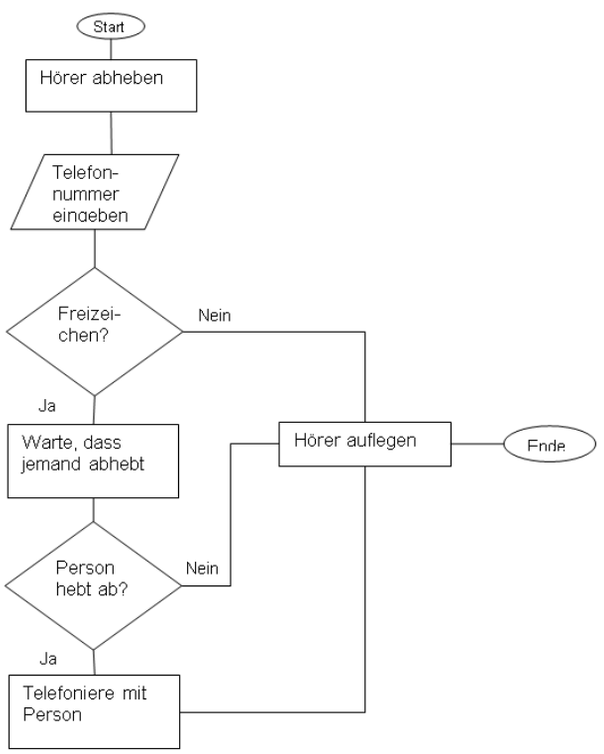
Wohnung betreten
Schlüssel ins Schloss stecken
Schlüssel bis zum Anschlag gegen den Uhrzeigersinn drehe
Tür öffnen
Schlüssel zur senkrechten Stellung zurückdrehen
Schlüssel abziehen
durch die Tür gehen
Tür schließen

Flussdiagramme

Eine weitere Möglichkeit für die Darstellung eines Programmablaufs ist das Flussdiagramm. Hierbei gelten folgende Vereinbarungen:



Das nebenstehende Flussdiagramm beschreibt die Abfolge beim Telefonieren. Zu Beginn wird der Hörer abgenommen (Anweisung). Nun muss eine Eingabe erfolgen. Die Telefonnummer wird eingegeben. Ertönt kein Freizeichen, so wird der Hörer aufgelegt. Der Vorgang ist beendet. Ertönt ein Freizeichen, wartet man, bis jemand ans Telefon geht. Es gibt wieder zwei Möglichkeiten, entweder es geht keiner ans Telefon, dann wird der Hörer aufgelegt und der Vorgang ist beendet, oder es hebt jemand ab und man kann mit der Person telefonieren. Moderne Möglichkeiten des Rückrufs etc. sind hier nicht berücksichtigt.



Auch mit Struktogrammen lassen sich Entscheidungen darstellen. Hierauf wird aber erst weiter hinten eingegangen.

Aufgaben:

1. Beschreibe mit Worten und mithilfe eines Struktogramms den Alltagsalgorithmus „Zähne putzen“.
2. Erstelle ein Flussdiagramm zum Alltagsalgorithmus „Waffelbacken“. Der Teig ist bereits fertig, du sollst nur den Algorithmus „Backen einer Waffel mit Waffeleisen“ zeichnen.

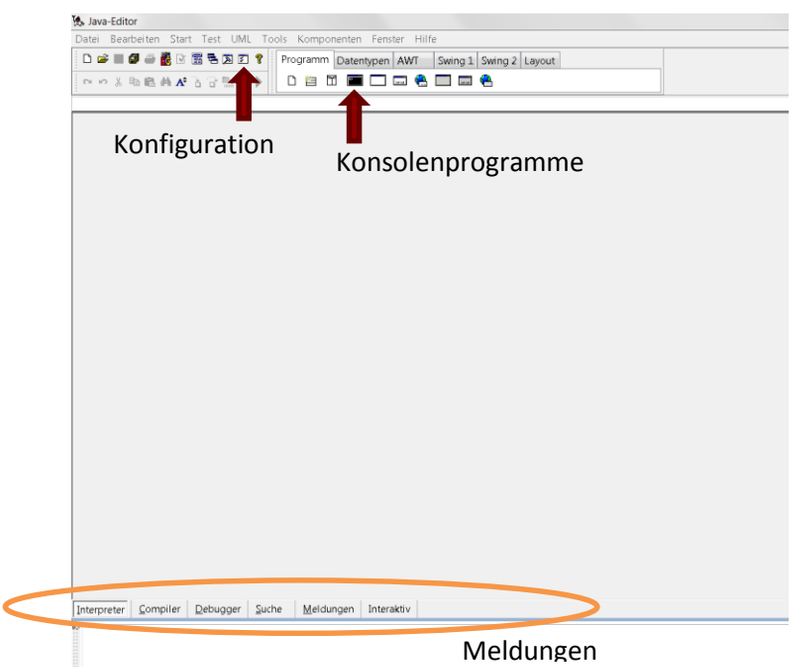
Das erste Java-Programm

Der Java-Editor

Software, mit der man sowohl den Quellcode verfassen und abspeichern, als auch kompilieren und ausführen kann, nennt man Entwicklungsumgebung. Es gibt eine größere Anzahl von Entwicklungsumgebungen, allerdings bietet der Java-Editor von Hr. Röhner einige Vorteile für den schulischen Einsatz, so dass ich befürworte, dass möglichst alle mit dieser Entwicklungsumgebung arbeiten.

Der Java-Editor wurde extra für die Schule entworfen. Um mit dem Java-Editor arbeiten zu können, muss man sich neben dem Programm selbst noch das Java-Development-Kit (JDK) in 32-Bit kostenlos herunterladen und installieren. Wer die Software unter Linux einsetzen will, braucht zudem Wine und unter Mac OS X ist CrossOver erforderlich.[12] Hr. Röhner erläutert dazu weiter: „Java-Editor funktioniert unter Linux mit der Wine-Erweiterung. Hinweise zur Installation finden Sie auf der WineHQ Seite. Unter Wine wird die Schriftart Monospace empfohlen. Manche Schriftarten funktionieren nicht im Editor. … Der Java-Editor läuft auf dem Mac sowohl mit CrossOver Office (Wine) als auch in einer virtualisierten Windows XP-Instanz problemlos. Bei einem 64-Bit Mac funktioniert er unter der Virtualisierungssoftware VMWare Fusion einwandfrei. Die Alternativen Boot Camp, VirtualBox, CrossOver und Parallels scheinen dies nicht zu ermöglichen.“[13]

Anschließend muss man dann einige Konfigurationen durchführen (siehe Unterricht).



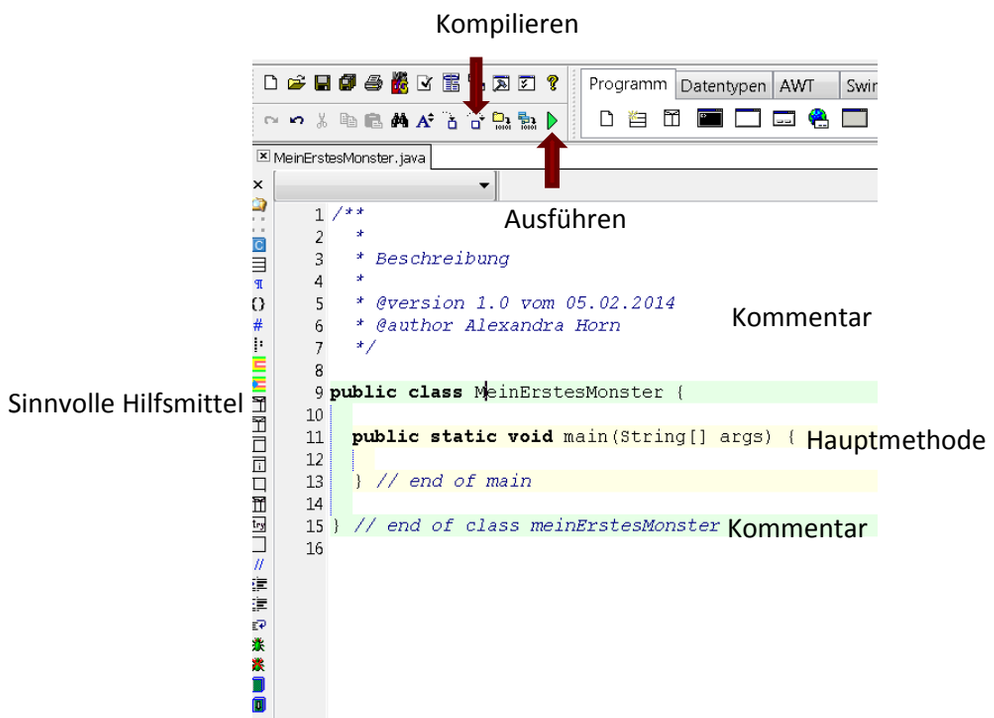
[12] http://www.heise.de/download/java-editor.html am 05.02.2014
[13] http://www.javaeditor.org/index.php/Java-Editor/de#Integrierte_Entwicklungsumgebungen am 05.02.2014

Konsolenprogramme sind für die Arbeit auf der Konsole (MS-DOS-Eingabeaufforderung, Unix-Shell) gedacht. Sie verfügen über keine grafische Oberfläche und können deshalb auch außerhalb einer grafischen Umgebung eingesetzt werden. [14]

Wir werden zunächst mit solchen Programmen beginnen. Sie sehen zwar nicht so schön aus, lenken die Aufmerksamkeit aber nicht auf die aufwändige Gestaltung einer Benutzeroberfläche, so dass man sich zunächst auf die wesentlichen Schritte der eigentlichen Programmierung konzentrieren kann.

Das erste Programm

Wir starten den Java-Editor und klicken einmal auf „Console“. Nachdem man den Namen MeinErstesMonster eingegeben hat, erscheint folgender Bildschirm:

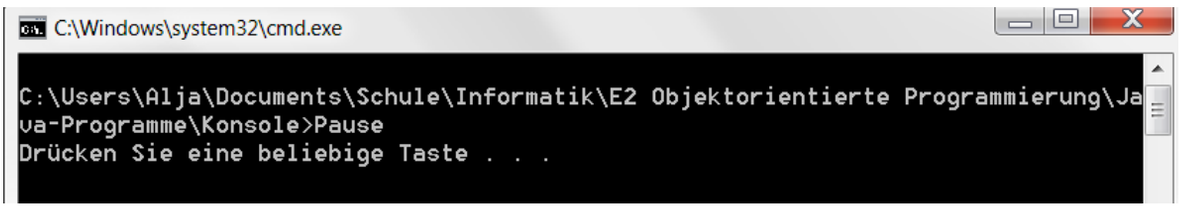


Wenn wir nun auf Kompilieren klicken, erscheint entweder eine **Fehlermeldung** oder es wird angezeigt, dass die Datei erfolgreich kompiliert worden ist. Solange wir Fehlermeldungen erhalten, müssen wir versuchen, die Fehler zu korrigieren.

Achte bei der Fehlersuche auf die angegebene Zeile, in der sich der Fehler befindet. Kontrolliere alle Klammern und Semikolons.

Wenn unser Programm erfolgreich kompiliert werden konnte, lassen wir es ausführen, indem wir auf den Ausführen-Button klicken. Es erscheint folgender Bildschirm:

[14] Vgl. (Röhner G. , Unterrichtsskript: Einführung in das Programmieren, 2007/2008)



Es ist also noch nichts Spannendes passiert, da wir ja auch noch keine wirklichen Anweisungen erteilt haben.

Abspeichern der Dateien

Alle Dateien, die zu einem Programm gehören, müssen in einem gemeinsamen Ordner abgespeichert werden. Zu Beginn wird unser Programm immer nur aus einer Datei bestehen, dennoch sollte man von vorneherein sich angewöhnen, **für jedes Programm einen eigenen Ordner anzulegen**. Alle diese Programmordner speichert man am besten in einem Ordner „Meine Java-Programme“.

Der Name der Datei muss immer mit dem Namen der Klasse übereinstimmen. **Klassennamen beginnen immer mit einem Großbuchstaben.** Im Namen dürfen keine Leerzeichen, Umlaute oder Sonderzeichen vorkommen.

Kommentare

Kommentare sind im Quelltext sehr wichtig. Damit der Programmierer und seine Teammitglieder auch nach einigen Wochen bzw. Monaten noch wissen, warum sie welche Programmteile verfasst haben, werden Kommentare in den Quelltext eingefügt. Sie sollen erläutern, welche Aufgaben von Programmteilen wie übernommen werden. Die Kommentare müssen vom restlichen Quelltext getrennt werden, so dass der Compiler sie bei der Übersetzung in Java-Bytecode ignoriert.

Einzeilige Kommentare werden durch „//“ eingeleitet. **Mehrzeilige** Kommentare beginnen mit „/**“, jede weitere Zeile beginnt mit „*“ und die letzte Zeile des Kommentars beginnt mit „*/“ oder sie werden mit „/*“ eingeleitet und enden mit „*/“. Kommentare dürfen nicht ineinander verschachtelt werden.

Quelltexte ohne Kommentare sind nach kurzer Zeit unlesbar und somit auch unbrauchbar. Man kann sie nur mühsam weiterbearbeiten, so dass es oft einfacher ist, das ganze Programm von Grund auf neu zu schreiben. Die Funktionsweise einzelner Programmteile sollte daher immer in Kommentaren erläutert werden. Um automatisch eine Dokumentation zum Quelltext zu erstellen, kann man javadoc verwenden. Dabei werden die Kommentare ausgewertet, die mit „/**“ beginnen. Der erste Satz innerhalb eines solchen Kommentars wird als Beschreibung für die Dokumentation verwendet.

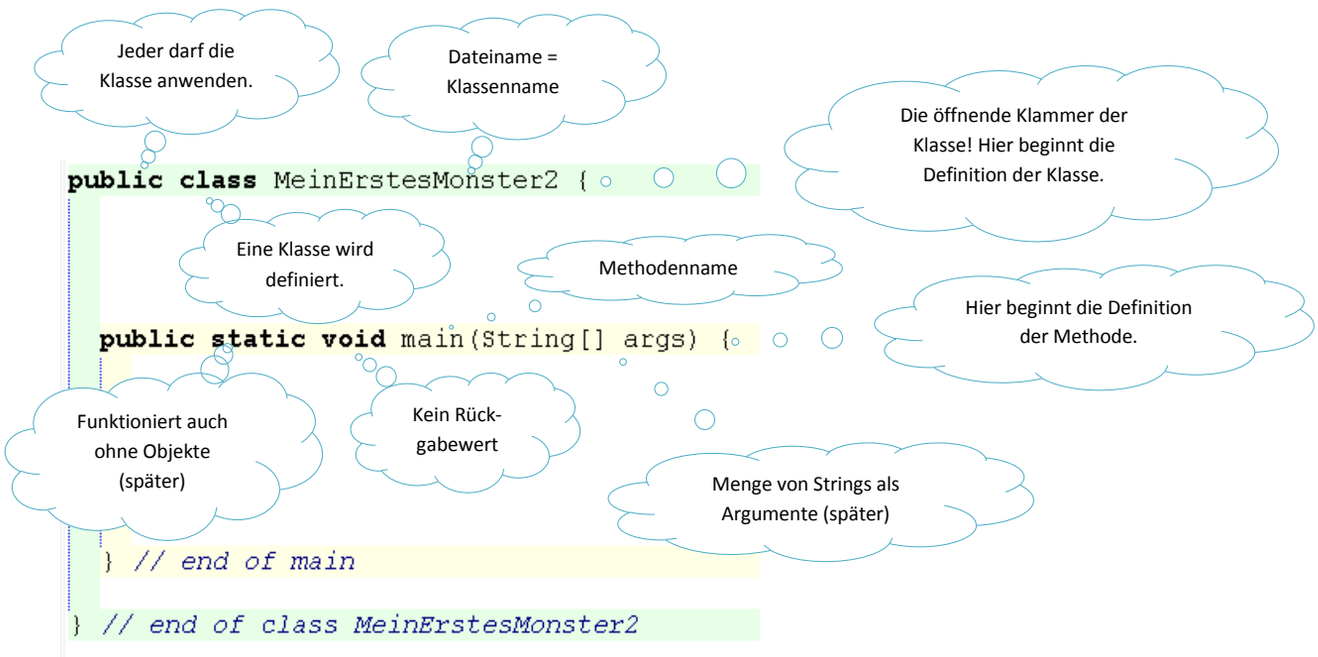
Die Hauptmethode

Jedes Programm benötigt eine Klasse, die die Hauptmethode beinhaltet. Mit ihr wird das gesamte Programm gestartet. Sie gibt an, welche Anweisungen ausgeführt werden sollen. Die Hauptmethode ist durch den Methodennamen **main** gekennzeichnet, dabei handelt es sich um ein Schlüsselwort von Java, welches nur in diesem Zusammenhang verwendet werden darf. Ein Java-Programm kann jedoch aus mehreren Klassen bestehen, von denen mindestens eine die main-Methode beinhalten muss.

Wenn du ein Programm ausführen willst, so lädt die virtuelle Maschine die entsprechende Klasse und führt dann die main-Methode aus, daher kann ein Programm ohne main-Methode nicht gestartet werden.

```
public static void main(String[] args) {

} // end of main
```

Die Hauptmethode wird immer mit dem Zugriffsoperator ‚public' eingeleitet. Dies bedeutet, dass die Methode öffentlich zugänglich ist. ‚static' gibt an, dass die Methode auch ohne bereits bestehende Objekte ausgeführt werden kann. ‚void' gibt an, dass die Methode keinen Rückgabewert zurückgibt.



Aufgaben:

1. Schreibe das oben angegebene Programm im Java-Editor, lasse es kompilieren und anschließend ausführen.
2. Ergänze das oben angegebene Programm „MeinErstesMonster" um folgende Zeilen. Beginne in Zeile 12.
   System.out.println(„Buuuuh!");
   System.out.println(„Ich bin ein Monster!");
   System.out.println(„Ich bin gefährlich.");
   Lasse auch dies Programm kompilieren und ausführen.
   TIPP: Die Tastenkombination **Strg + u** fügt System.out.println() ein.
3. Bei der Ausführung dieses Programms bewirkt System.out.println(„xxx"), dass der Text in den Anführungsstrichen auf dem Bildschirm ausgegeben wird. Man nennt dies **Ausgabe** bzw. **Output**. Ersetze println durch print, d. h. es wird nur ln abgehängt. Welchen Unterschied kannst du bei den beiden Ausgabeanweisungen feststellen?
4. Erläutere die folgenden Begriffe schriftlich (möglichst ohne im Skript nachzulesen): Quelltext, kompilieren, ausführen, höhere Programmiersprache, Eingabeaufforderung und Entwicklungsumgebung.
5. Wie lautet der Dateiname des Java-Bytecodes für das oben erstellte Programm?

6. Erstelle ein Programm, welches folgende Ausgabe erzeugt: Java-Programme bestehen aus Klassen. Speichere dieses Programm unter dem Namen ErsteAusgabe ab. Kompiliere es und lasse das Programm ausführen. Ändere den Dateinamen unter Datei – Speichern unter. Beobachte, was im Quelltext geschieht.

Grundlegende Sprachelemente

Syntax

In jeder Programmiersprache gelten bestimmte Regeln, die festlegen, welche Schreibweisen bei der Erstellung des Quellcodes erlaubt sind und welche Kombinationen aus Zeichen, Text und Symbolen verwendet werden dürfen. Die Gesamtheit dieser Regeln nennt man Syntax.

In Java gibt es einige reservierte Wörter, die man auch **Schlüsselwörter** nennt. Sie haben bereits eine vorgegebene Bedeutung für den Compiler. Diese reservierten Wörter dürfen nur in dem vorgesehenen Kontext verwendet werden. Der Java-Editor kennzeichnet die Schlüsselwörter durch Fettdruck.

Einige wichtige Regeln:

- Jede Anweisung muss mit einem Semikolon enden.
- Leerzeichen sind in der Regel egal, sie erleichtern nur das Lesen.
- Alles, was zu einem bestimmten Bereich gehört, wird in geschweifte Klammern geschrieben. Die Klammerungen sind – wie in der Mathematik – immer so zu setzen, dass jede innere Klammer erst geschlossen werden muss, bevor die äußere Klammer geschlossen werden kann.

Syntaxdiagramme

Regeln können auch mithilfe von Syntaxdiagrammen dargestellt werden. Syntaxdiagramme bestehen aus Graphen, also Kanten (Pfeilen) und Knoten (Kästchen bzw. Ellipsoide), die Teile der Programmiersprachensyntax beschreiben. Ein syntaktisch richtiges Programmstück stellt einen möglichen Weg durch den Graphen dar. Fehlerhafte Programme führen in Sackgassen. Trifft man auf dem Weg auf ein **Ellipsoid**, dann stellt dieses einen **Schlüsselbegriff** dar, der nicht weiter ersetzt werden kann. **Kästchen** enthalten **Sprachelemente**, die in anderen Syntaxdiagrammen noch weiter zu präzisieren sind. Anwendungen zu den Syntaxdiagrammen folgen im nächsten Abschnitt.

Datentypen, Variablen und Konstanten

Eingegebene Daten können vom Computer nur verarbeitet werden, wenn der Computer weiß, um was für eine Art Daten es sich handelt. Grundsätzlich unterscheidet der Computer zwischen numerischen Daten (Zahlen), alphanumerischen Daten (Buchstaben) und logischen Daten (wahr oder falsch). Bei all diesen Daten handelt es sich um primitive Datentypen, die sich im zulässigen Wertebereich und in ihrem Speicherplatzbedarf unterscheiden.

Numerische Datentypen

Bei den numerischen Datentypen werden die ganzen Zahlen und die Dezimalzahlen unterschieden. Alle Zahlen können mit positiven und negativen Vorzeichen versehen werden. Ist kein Vorzeichen vorhanden, so wird die Zahl als positiver Wert erkannt. Wenn sicher ist, dass es sich nur um ganzzahlige Werte handelt, verwendet man den Datentyp Integer. Sie

werden immer genau dargestellt und es treten keine Rundungsfehler auf. Ergeben sich durch Berechnungen Kommastellen, so werden die Nachkommastellen einfach abgeschnitten. Es gibt vier verschiedene Datentypvarianten, die sich in ihrem Wertebereich und der Größe des belegten Speicherplatzes unterscheiden. Üblicherweise verwendet man für ganzzahlige Werte den Datentyp int.

Datentyp	Wertebereich	Speichergröße
byte	-128 … 127	1 Byte
short	-32768 … 32767	2 Byte
int	-2.147.483.648 … 2.147.483.647	4 Byte
long	-9.223.372.036.854.775.808 … 9.223.372.036.854.775.807	8 Byte

Dezimalzahlen (Fließkommazahlen, Gleitkommazahlen) können nicht immer exakt dargestellt werden, so dass es zu Rundungsfehlern kommen kann. Je nach verwendetem Datentypen lassen sich nur Zahlen mit einer gewissen Genauigkeit darstellen. Üblicherweise verwendet man den Datentyp double, da sich Rundungsfehler dort nicht so stark bemerkbar machen und die Rechner über genügend Speicherplatz verfügen. Als Dezimaltrennzeichen muss der Punkt verwendet werden.

Datentyp	Genauigkeit	Speichergröße
float	7 Stellen	2 Byte
double	15 Stellen	4 Byte

Alphanumerische Datentypen

Wenn man nur einzelne Buchstaben verwenden möchte, so verwendet man den Datentyp **char**. Ganze Wörter bzw. Sätze sind vom Datentyp **String**. Wörter und Sätze nennt man Zeichenketten. Der Datentyp String gehört nicht zu den elementaren Datentypen. Genau genommen handelt es sich um eine Klasse.

Logische Datentypen

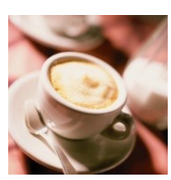
Ein Wahrheitswert kann nur zwei Werte annehmen. Er ist entweder richtig (**true**) oder falsch (**false**). Der Datentyp nennt sich **boolean**.

Variablen

Variable sind Behälter. Sie haben eine bestimmte Größe und einen bestimmten Typ. In diesem Abschnitt beschäftigen wir uns zunächst mit Variablen, die nur elementare Datentypen (s.o.) enthalten können. Jede Variable hat einen eigenen Namen. Jede Variable kann etwas enthalten.

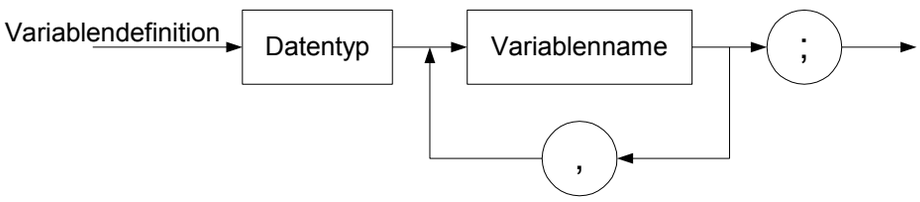
```
2  /**
3    *
4    * Auswertung der Schulsprecher-Wahl:
5    * aus den erhaltenen Stimmen soll der Stimmanteil berechnet werden
6    * @version 1.0 vom 14.12.2010
7    * @author  Alexandra Horn
8    */
9
10 public class Wahlauswertung {
11
12   public static void main(String[] args) {
13     //Definition von Variablen
14     int anzahl1, anzahl2, anzahl3;
15     float summe, anteil1, anteil2, anteil3;
16
17     //Eingabe
18     System.out.print("Stimmenanzahl Max: "); anzahl1 = Input.readInt();
19     System.out.print("Stimmenanzahl Lea: "); anzahl2 = Input.readInt();
20     System.out.print("Stimmenanzahl Franz: "); anzahl3 = Input.readInt();
21
22     //Verarbeitung
23     summe = anzahl1 + anzahl2 + anzahl3;
24     anteil1 = anzahl1 / summe * 100;
25     anteil2 = anzahl2 / summe * 100;
26     anteil3 = 100 - anteil1 - anteil2;
27
28     //Ausgabe
29     System.out.println("Anteil Max: "+ anteil1 + " %");
30     System.out.println("Anteil Lea: "+ anteil2 + " %");
31     System.out.println("Anteil Franz: " + anteil3 + " %");
32   }
33 }
34
```

Im oben stehenden Programm[15] werden insgesamt sieben Variablen deklariert. Die Variablen anzahl1, anzahl2 und anzahl3 sind vom Typ int. Die vier anderen Variablen sind vom Typ float. Durch die Definition in den Zeilen 14 und 15 wird zunächst nur Speicherplatz reserviert. Bis hierher haben wir uns also nur für drei Behälter der Größe int und vier Behälter der Größe float entschieden. Der Speicherplatz wurde mit den entsprechenden Namen versehen. Klar ist damit, dass man die Zahlen vom Typ float nicht in einen Behälter vom Typ int hineinstecken kann, weil er dort nicht hinein passt. Man schüttet ja auch keinen Cappuccino in eine Espressotasse.



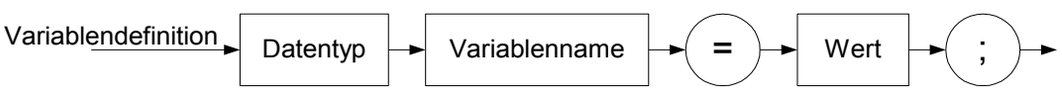
[15] Damit dieses Programm funktioniert, muss die Datei Input.java im gleichen Verzeichnis wie die Datei Wahlauswertung abgespeichert sein. Eine andere Art der Dateneingabe werden wir weiter hinten kennen lernen.

Syntaxdiagramm zur Variablendefinition:



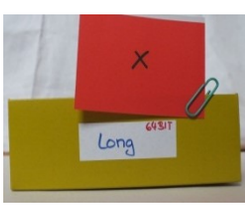

```
//Variablen deklarieren
int gewicht;
int x,y;
```

Es gibt noch eine weitere Möglichkeit in Java eine Variable zu definieren:




```
//Variablen deklarieren und einen Wert zuweisen
int größe = 175; //heutzutage funktionieren auch Sonderzeichen etc.
String schüler = „Paul“; //doppelte Anführungsstriche beim String                    //
keinen Zeilenumbruch!
```

Bei dieser Möglichkeit werden nicht nur der Datentyp und der Variablenname festgelegt, sondern die Variable wird auch gleichzeitig **initialisiert**, d. h. es wird ein Wert festgelegt.



Bei der ersten Möglichkeit wird nur der Speicherplatz reserviert, d.h. die Tasse wird bereitgestellt.

Bei der zweiten Möglichkeit wird der Speicherplatz reserviert und ein Wert wird im Speicher abgelegt, d. h. die Tasse bereitgestellt und sofort mit Kaffee gefüllt. Das Gleichheitszeichen ist in Java ein **Zuweisungsoperator**.

Im Java-Programm Wahlauswertung wird die erste Möglichkeit genutzt. Initialisiert werden die drei int-Variablen durch Einlesen einer Eingabe in der Eingabeaufforderung. Anschließend werden die Werte verarbeitet. Dabei wird der Variablen summe das Ergebnis der Addition der drei int-Variablen zugewiesen. Das Ergebnis der Berechnung wird also in der Variablen abgelegt. Die jeweiligen Stimmanteile werden dann berechnet und die drei float-Variablen jeweils belegt. Anschließend erfolgt die Ausgabe.

Beim Verwenden von Variablen muss man beachten, dass eine Variable zunächst definiert werden muss, bevor man ihr einen Wert zuweisen kann, und dass eine Variable zunächst initialisiert sein muss, bevor man sie verwenden kann. Außerdem darf man Variablen nur Werte zuweisen, die dem vereinbarten Typ entsprechen. Bildlich kann man es sich so merken, dass man den Kaffee auch erst in die Tasse schütten kann, wenn man die Tasse bereitgestellt hat.

Variablen sind nur in dem Anweisungsblock gültig, indem sie definiert sind.

Welcher Wert passt in welche Variable?

Einen großen Wert kannst du nicht in einen kleinen Behälter stecken. Du könntest es zwar, aber dann würdest du beim Cappuccino etwas verschütten und beim Programmieren würde der Compiler versuchen, dies zu verhindern. Beispielsweise kannst du nicht den Inhalt eines int-Behälters in einen byte-Behälter füllen, auch wenn der Wert aus dem int-Behälter eigentlich klein genug wäre, um in den byte-Behälter zu passen. Allerdings kannst du einen byte-Wert in eine int-Variable stecken, weil ein byte-Wert immer in den Behälter mit int-Größe passt. Das nennt man implizite Erweiterung. Der Compiler verbietet nur, dass du falsche Dinge oder zu große Dinge in eine Variable steckst.


```java
public class VariablenDeklaration {

  public static void main(String[] args) {
    //Variablen deklarieren und initialisieren
    float a = 14;
    byte b = 14;
    long c = 14;
    //Kann man den byte-Wert 14 in a abspeichern?
    a = b;
    //Kann man den long-Wert 14 in a abspeichern?
    a = c;
    //Kann man den float-Wert 14 in b abspeichern?
    b = a;
  } // end of main

} // end of class VariablenDeklaration
```

Beim Kompilieren des Programms „VariablenDeklaration“ erscheint folgende Fehlermeldung:

```
Compiliere C:\Users\Alja\Documents\Schule\Informatik\E2 Objektorie
VariablenDeklaration.java:21:9: error: possible loss of precision
    b = a;
        ^
  required: byte
  found:    float
1 error
```

Warum ist das so? Du weißt, dass der Wert 14 definitiv klein genug ist, um auch in eine byte-Variable zu passen. Der Compiler weiß dies jedoch nicht. Der Compiler kümmert sich nur darum, ob du versuchst, etwas Großes in etwas Kleines zu zwängen, und dabei evtl. etwas verschütten würdest.

Wie erhält eine Variable ihren Wert?

Hier gibt es mehrere Möglichkeiten.

1. Ein Wert wird nach dem Gleichheitszeichen eingegeben, z. B. x = 14
2. Eine Variable erhält den Wert einer anderen Variablen, z. B. x = y
3. Einer Variable wird der Wert eines Ausdrucks übergeben, z. B. x = y + 14
   Hierbei ist zu beachten, dass es sich bei dem Gleichheitszeichen um den Zuweisungsoperator handelt. Es ist keine mathematische Gleichung. Für den Computer bedeutet diese Eingabe folgendes: Schau in der Variablen y nach, welcher Wert dort abgelegt ist. Addiere zu diesem Wert 14 und speichere das Ergebnis in der Variablen x ab.

Konstanten

Manchmal kommen in Programmen Werte vor, die sich nicht ändern. Es handelt sich also um konstante Werte, wie z. B. den Umrechnungsfaktor von mm in inch, die Mehrwertsteuer oder die Schallgeschwindigkeit. Um eine Abänderung zu verhindern, wird eine Konstante genauso definiert und initialisiert wie eine Variable, jedoch wird vor den Datentyp das Schlüsselwort **final** gesetzt. Damit man

Konstanten besser von Variablen im Quelltext unterscheiden kann, schreibt man den Konstantennamen nur mit Großbuchstaben.

```
final double PI = 3.1415;
```

Aufgaben:

1. Notiere alle Schlüsselwörter, die du bisher kennen gelernt hast, mit ihrer Bedeutung. Notiere alle elementaren Datentypen, die du kennen gelernt hast.
2. Kreise die Anweisungen ein, die erlaubt wären, wenn diese Zeilen in einer Methode ständen.
   a) int x = 34.5;
   b) boolean boo = x;
   c) int g = 17;
   d) int y = g;
   e) y = y+10;
   f) short s;
   g) s = y;
   h) byte b = 3;
   i) byte v = b;
   j) short n = 12;
   k) v = n;
   l) Byte k = 128;
   m) n = n + 12;

3. Welche Werte haben die Variablen *Summe* und *Produkt* nach folgender Sequenz von Wertzuweisungen?

```
int Summe, Produkt;
Summe = 2;
Summe = Summe + 1;
Produkt = Summe;
Produkt = Produkt * Produkt;
Produkt = Summe * Produkt;
Summe = Summe + Produkt;
```

Einlesen von Daten über die Tastatur

Wenn wir Eingaben über die Tastatur einlesen wollen, können wir dies mithilfe der Klasse Input tun. Es gibt jedoch noch eine andere Möglichkeit, die hier kurz vorgestellt werden soll.

```
1  import java.util.Scanner;
2
3  public class C13SystemIn {
4    public static void main(String[] args) {
5      String name;
6      int alter;
7      System.out.println("Meine erste Eingabe-Anwendung");
8      Scanner tastatur = new Scanner(System.in);
9      System.out.print("\nWie heisst du? ");
10     name=tastatur.nextLine();
11     System.out.print("\nWie alt bist du? ");
12     alter=tastatur.nextInt();
13     System.out.println("\nHallo "+name+" \nmit "+alter+" Jahren lernst du Java leicht?");
14   }
15 }
```

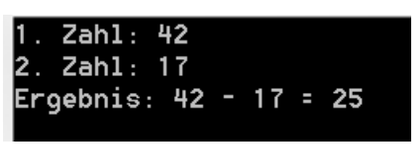
Java bietet bereits etliche Klassen an, auf die jeder Programmierer zurückgreifen kann. Um die Tastatur zu scannen, benötigen wir die Klasse java.util.Scanner. Damit unsere Anwendung weiß, dass wir Methoden einer anderen Klasse verwenden wollen und wo sich diese Methoden befinden, importieren wir die oben angegebene Klasse (Zeile 1). Anschließend erzeugen wir einen neuen Scanner, den wir in unserem Fall tastatur nennen (Zeile 8). Dabei handelt es sich um eine Variablendeklaration, wobei hier kein elementarer Datentyp verwendet wird sondern ein Objekt.[16] In Zeile 10 wird nun die Methode nextLine() von Scanner aufgerufen, sie liest alle Wörter und Leerzeichen ein und ordnet diese der Variablen name zu. In Zeile 12 wird die Methode nextInt() verwendet, welche nur eingegebene Integerwerte in der Variable alter speichert. In Zeile 13 wird die Anweisung zur Ausgabe programmiert. Das +-Zeichen fügt die einzelnen Strings und Variableninhalte aneinander. Durch das vorgestellte \n wird eine neue Zeile eingefügt.

Methode	**Liest von der Tastatur**
String nextLine()	Liest alles, was eingegeben wird
String next()	liest das nächste Wort
int nextInt()	liest eine Integerzahl ein
float nextFloat()	liest eine Dezimalzahl vom Typ Float ein
double nextDouble()	liest eine Dezimalzahl vom Typ Double ein

Wenn du im Java-Editor programmierst, unterstützt das Programm dich bei der Wahl der Methoden, indem sich ein Klappmenü öffnet, sobald du den Punktoperator hinter dem Wort tastatur eingibst.

Aufgaben:

4. Tippe das Programm ab und teste es.
5. Schreibe ein Programm, welches zwei Zahlen als Eingabe einliest und diese dann addiert. Das Ergebnis soll ausgegeben werden.
6. Notiere dir wichtige Erkenntnisse, die du während des Programmierens der folgenden Anwendungen bemerkt hast. Deine einzelnen Programme sollten immer Ausgaben tätigen, damit du sehen kannst, was passiert.
    a. Erstelle ein Konsolenprogramm Subtraktion, indem du die Variablen ersteZahl, zweiteZahl und ergebnis definierst. Weise den Variablen ersteZahl und zweiteZahl Werte zu. In der Variablen ergebnis soll die Differenz der beiden Zahlen berechnet werden. Die Konsolenaufgabe soll so aussehen:



    b. Erstelle eine Anwendung zur Addition der beiden Werte und speichere diese unter Addition.java.
    c. Lasse zwei ganze Zahlen vom Typ int addieren und ordne dem Ergebnis den Datentyp short zu. Kannst du die Anwendung trotz Fehlermeldung starten?
    d. Lasse nun eine ganze Zahl und eine Dezimalzahl addieren. Experimentiere für die Dezimalzahlen mit den Datentypen float und double.

[16] Näheres hierzu kommt in der Q-Phase.

e. Schreibe eine Anwendung, in der du zwei ganze Zahlen dividieren kannst, und speichere sie unter Division ab. Experimentiere mit verschiedenen Werten. Welche Ergebnisse erhältst du, wenn die ergebnis-Variable vom Typ int ist, welche beim Typ double?

7. Schreibe ein Programm, welches die Oberfläche und das Volumen einer Dose berechnet. Vorgegeben werden Durchmesser und Höhe der Dose. Der Klassenname lautet: Zylinderberechnung. Kommentiere den Quelltext. Du benötigst zur Berechnung π. Speichere den Wert in einer Konstanten.
   Zusatz für die Schnellen: Die Ausgabe enthält viele Nullen, die den Blick auf das Wesentliche trügen, daher soll die Ausgabe noch formatiert werden. Hierzu verwendet man System.out.printf(). Lies im Internet, im Tutorial oder im Javabuch nach, wie du die Ausgabe formatieren kannst.
8. Eine Maklerfirma verkauft Grundstücke. Schreibe ein Programm, das für ein rechteckiges Grundstück die Längen der Seiten (in Meter) und den Quadratmeterpreis einliest. Gib dazu eine Rechnung auf dem Bildschirm aus, die die eingegebenen Daten, den Grundstückspreis, die Maklergebühr von 3%, die Mehrwertsteuer, sowie den Gesamtbetrag enthält.
9. Schreibe ein Programm, indem du die Werte zweier Variablen vertauschen kannst, d. h. der Wert, der in der Variablen a steckt, soll nach der Vertauschung in der Variablen b stecken und umgekehrt. Erzeuge eine Ausgabe, um deine Lösung zu kontrollieren.
10. Zeichne das Syntaxdiagramm für Konstanten, so dass die Konstante sowohl definiert als auch initialisiert wird.

Die Struktur von Algorithmen

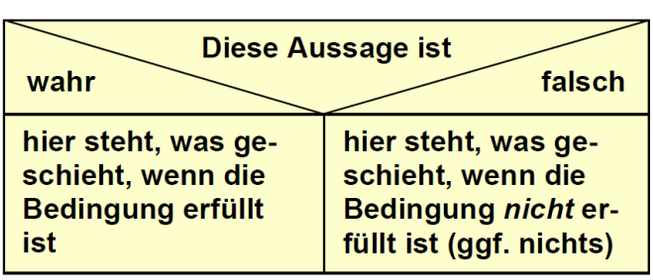
Bedingte Verzweigung

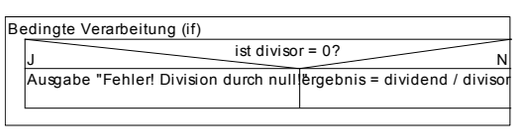
In vielen Fällen sollen Operatoren oder auch Sequenzen von Operationen nur unter einer bestimmten Voraussetzung ausgeführt werden, d. h. etwas soll ausgeführt werden, falls eine Bedingung erfüllt ist.

Beispiele:

- Wenn mehr als 100 Kunden am Tag der offenen Tür kommen, wird der Tag als erfolgreich gewertet.
- Wenn bis zu einem bestimmten Datum gezahlt wird, gewährt die Firma Skonto von 2 %.
- Wenn der Schüler mindestens 18 Jahre alt ist, darf er seine Entschuldigungen selbst schreiben, sonst muss es ein Erziehungsberechtigter tun.

Struktogramm:





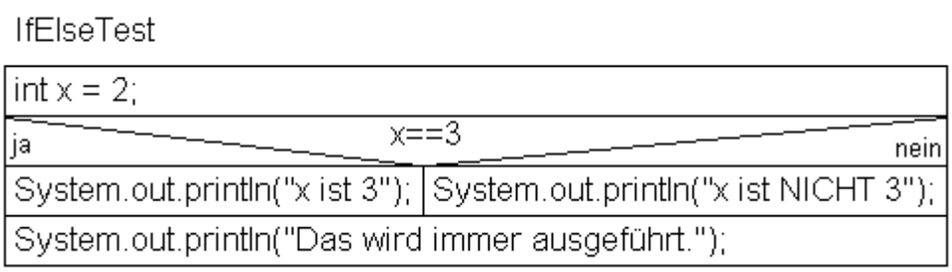
Wie sieht der Java-Quellcode aus?

```
1  public class ifTest {
2
3    public static void main(String[] args) {
4      int x = 3;
5      if (x ==3) {
6        System.out.println("x ist 3");
7      } // end of if
8      System.out.println("Das wird immer ausgeführt");
9    } // end of main
10
11 } // end of class ifTest
```

Die Anweisung in Zeile 5 sorgt dafür, dass alle Anweisungen, die zwischen den geschweiften Klammern (Zeile 5 und Zeile 7) stehen, nur ausgeführt werden, wenn x identisch mit dem Wert 3 ist. Die Zeile 8 steht außerhalb der geschweiften Klammern und wird unabhängig von dem Wahrheitsgehalt des Ausdrucks x == 3 ausgeführt.

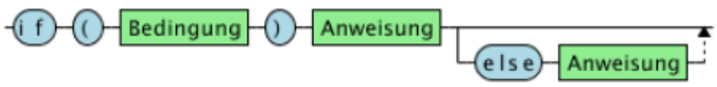
```
1  public class ifElseTest {
2
3    public static void main(String[] args) {
4      int x = 2;
5      if (x == 3) {
6        System.out.println("x ist 3");
7      } // end of if
8      else {
9        System.out.println("x ist NICHT 3");
10     } // end of if-else
11     System.out.println("Das wird immer ausgeführt");
12   } // end of main
13
14 } // end of class ifElseTest
15
```

Mit dem Java-Editor kann man (ab Version 12.0) auch Struktogramme erstellen. Für das Programm ifElseTest sieht dies folgendermaßen aus:



In diesem Programm wird zunächst überprüft, ob x identisch mit 3 ist. Dies ist nicht der Fall, also wird Zeile 6 nicht ausgeführt. Anstelle von Zeile 6 wird nun der else-Teil ausgeführt.

Syntaxdiagramm:



Flussdiagramm einer if-Anweisung:

Flussdiagramm-Beispiele (if-Anweisungen)

Die Bedingung:

Eine Bedingung muss immer eindeutig richtig oder falsch sein. Es handelt sich also um einen booleschen Test. Die einfachste Möglichkeit für einen booleschen Test ist es, wenn der Wert einer Variablen mithilfe eines Vergleichsoperators überprüft wird. Zu diesen Vergleichsoperatoren zählen:

< (kleiner als)
> (größer als)
== (gleich) (zwei Gleichheitszeichen hintereinander entsprechen also dem mathematischen Gleichheitszeichen)

Aufgaben:

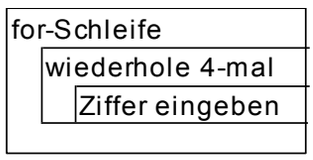
1. Schreibe ein kleines Taschenrechnerprogramm. Dein Programm soll vom Anwender zwei Zahlen sowie die gewünschte Rechenoperation erfragen. Anschließend soll das Programm die gewünschte Rechnung ausgeben. Notiere zunächst das zugehörige Flussdiagramm.
2. Ein Versandhaus berechnet die Versandpauschale bei einer Bestellung folgendermaßen: Unter 200 € werden 20 € Versandpauschale berechnet. Ab 200 € ist der Versand kostenlos und der Kunde erhält einen Rabatt von 4 %.
    a. Notiere das entsprechende Struktogramm.
    b. Schreibe ein Java-Programm, welches für einen einzugebenden Betrag die Versandkosten berechnet und den Endpreis ausgibt.
3. Eine Bank führt bei jedem Girokonto die ersten 10 Buchungen kostenlos durch, für die nächsten Buchungen berechnet sie 15 Cent pro Buchung. Erstelle ein Java-Programm nach folgenden Vorgaben:
    - Eingabe der Anzahl der Buchungen
    - Berechnung der Gebühr
    - Ausgabe des Ergebnisses

Wiederholungen / Schleifen

Eine Folge von Verarbeitungsschritten wird wiederholt. Hier gibt es prinzipiell zwei unterschiedliche Formen.

Wiederholungsstruktur mit fester Anzahl an Wiederholungen (for-Schleife)

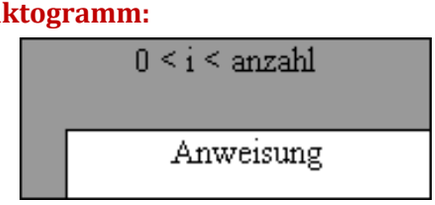
Die Anzahl der Durchläufe steht bereits vor der ersten Abarbeitung fest. Als Beispiel wird die Eingabe einer genau 4-stelligen Geheimzahl betrachtet.



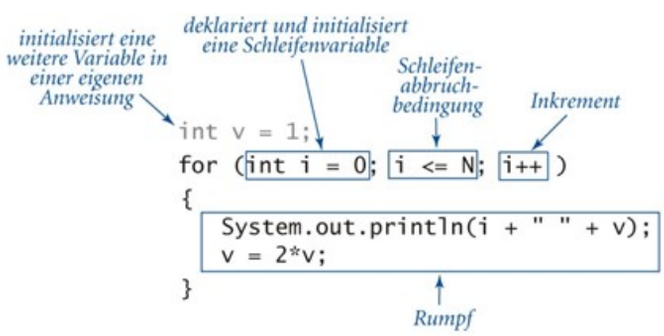
Beispiele:

- Die ersten 10 Quadratzahlen sollen aufgeschrieben werden.
- Ein bestimmter Geldbetrag soll für 10 Jahre auf einem Sparbuch hinterlegt werden und die Zinsen sollen für jedes Jahr ausgegeben werden.

Struktogramm:



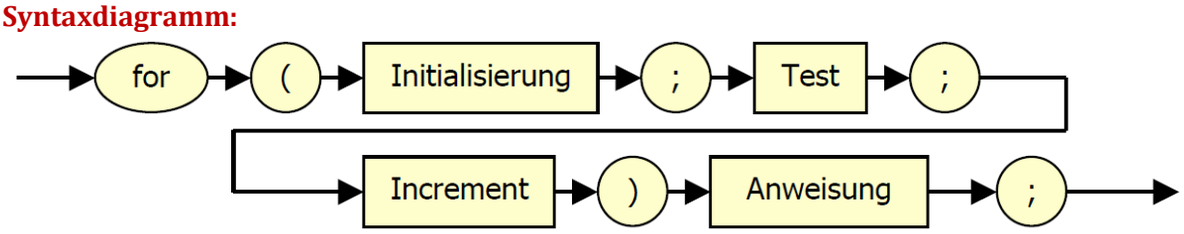
Wie sieht der Java-Quellcode aus?



Anatomie einer for-Schleife (die Zweierpotenzen ausgibt)

In der Variablen v wird 1 abgelegt. In der Zählvariablen i befindet sich die 0. Der Schleifencode wird solange durchlaufen, solange die Schleifenabbruchbedingung erfüllt ist. Die Schleifenabbruchbedingung ist erfüllt, solange i kleiner als ein zuvor definiertes N ist. i++ bedeutet, dass der Wert von i um eins erhöht wird, d. h. nach dem ersten Schleifendurchlauf ist i = 1. i++ ist gleichbedeutend mit i = i+1.

Syntaxdiagramm:



Aufgaben:

1. Ein funktionierendes Java-Programm ist vollständig durcheinander geraten. Ordne die Codeschnipsel wieder so an, dass ein funktionierendes Java-Programm entsteht, welches folgende Ausgabe erzeugt: Einige viele davon hinzu, wie Notiere den Quellcode.

```
0 4
0 3
1 4
1 3
3 4
3 3
```

der geschweiften Klammern sind zu Boden gefallen. Füge so benötigt werden.

```
x++;
```

```
if (x == 1) {
```

```
System.out.println(x + " "+y);
```

```
public class Aufräumen {
```

```
for(int y = 4; y > 2; y--) {
```

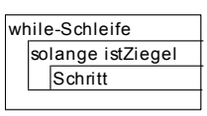
```
for(int x = 0; x < 4; x++) {
```

```
public static void main(String[] args) {
```

2. Schreibe ein Programm, welches die ersten 20 Quadratzahlen ausgibt.

Wiederholungsstruktur mit Bedingung (while-Schleife)

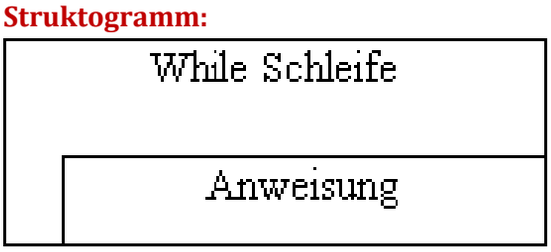
Alles, was hier im Schleifenblock steht, wird solange ausgeführt, wie die Bedingung wahr ist. Der Abbruch der Wiederholungen wird durch eine Bedingung gesteuert, so dass sich die Anzahl der Wiederholungen erst während der Abarbeitung der Struktur ergibt.

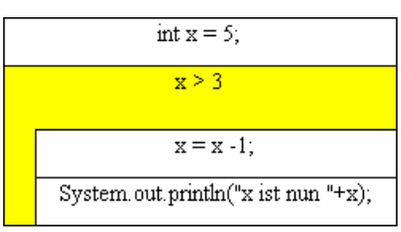


Beispiele:

- Solange noch Geld auf dem Konto ist, kann der Kunde Geld abheben.
- Solange noch Bier im Kühlschrank steht, kann Hr. Meier sich ein kühles Bier holen.
- Solange die Spülmaschine noch läuft, kann man sie nicht ausräumen.

Struktogramm:





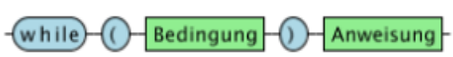
Wie sieht der Java-Quellcode aus?

```
1  public class whileTest {
2
3    public static void main(String[] args) {
4      int x = 5;
5      while (x > 3) {
6        //Der Schleifencode wird ausgeführt, da 4 > 3 ist
7        x = x -1;
8        // x ist nun 3, würde man den Wert von x in der Schleife nicht verändern,
9        // würde man für immer in der Schleife hängen bleiben (Endlosschleife!)
10       System.out.println("Der Wert von x beträgt nun "+x);
11     } // end of while
12
13     int z = 30;
14     while ( z == 20) {
15       //Der Schleifencode wird nicht ausgeführt, weil 30 ungleich 20 ist.
16     } // end of while
17   } // end of main
18
19 } // end of class whileTest
20
21
```

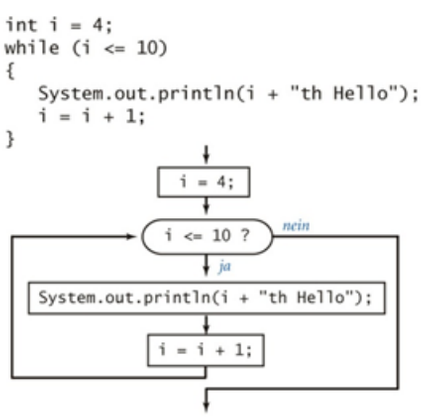
Die Ausgabe, die von dem Programm whileTest erzeugt wird, sieht wie folgt aus:

```
Der Wert von x beträgt nun 4
Der Wert von x beträgt nun 3
```

Syntaxdiagramm:



Flussdiagramm:



Flussdiagramm-Beispiel (while-Anweisung)

Boolesche Operatoren

Um Bedingungen zu Überprüfen werden boolesche Operatoren verwendet.

Operator	Bedeutung	Beispiel
==	ist gleich	if (Zahl1 **==** 3) ....
!=	ist ungleich	if (Zahl1 **!=** 3) ....
<	ist kleiner als	if (Zahl1 **<** Zahl2) ....
>	ist größer als	if (Zahl1 **>** Zahl2) ....
<=	ist kleiner oder gleich	if (Zahl1 **<=** 5) ....
>=	ist größer oder gleich	if (Zahl1 **>=** 4) ....

Aufgaben:

1. Das Programm DooBee soll folgende Ausgabe erzeugen: DooBeeDooBeeDo. Trage den fehlenden Code ein:

```
1  public class DooBee {
2
3    public static void main(String[] args) {
4      int x = 1;
5      while (x<___) {
6        System.out._______("Doo");
7        System.out._______("Bee");
8        x = x+1;
9      } // end of while
10     if (x==___) {
11       System.out.print("Do");
12     } // end of if
13   } // end of main
14
15 } // end of class DooBee
16
```

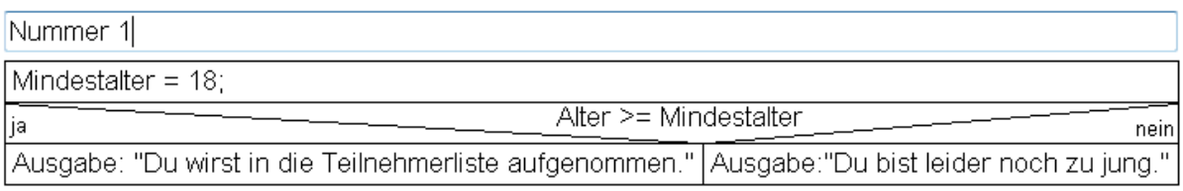
2. Erläutere, welche Ausgaben von dem Programm ProgWhile erzeugt werden.

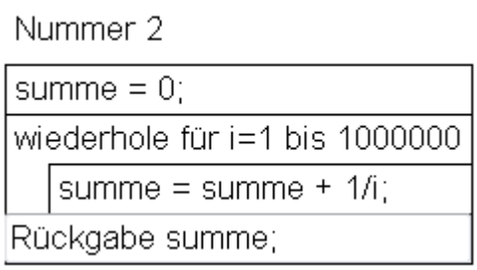
```
public class ProgWhile {
  public static void main (String[] args) {
    int guthaben = 1000;
    int sparziel = 8000;
    int einzahlung = 600;
    System.out.println ("Guthaben   = " + guthaben);
    System.out.println ("Sparziel   = " + sparziel);
    while ( guthaben < sparziel ) {
      guthaben = guthaben + einzahlung;
      System.out.println ("neues Guthaben = " + guthaben);
    }
    System.out.println( "Sparziel erreicht.");
  }
}
```

3. Erläutere, welche Ausgaben von dem Programm ProgFor erzeugt werden. Wie unterscheidet sich das Programm ProgFor von dem Programm ProgWhile?

```
public class ProgFor {
  public static void main (String[] args) {
    int guthaben = 1000;
    int einzahlung = 500;
    System.out.println("Guthaben    = " + guthaben);
    System.out.println("Einzahlung = " + einzahlung
      + " pro Monat");
    for ( int monat=1;  monat<=6; monat = monat + 1 ) {
      guthaben = guthaben + einzahlung;
      System.out.println(monat + ". Monat:");
      System.out.println("  Guthaben = " + guthaben);
    }
  }
}
```

4.
   a) *Interpretiere die Struktogramme.*
   b) *Schreibe zu den Struktogrammen den entsprechenden Java-Code.*

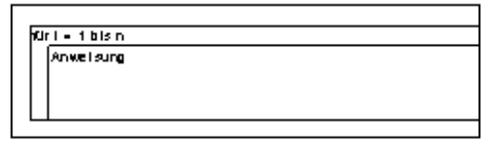
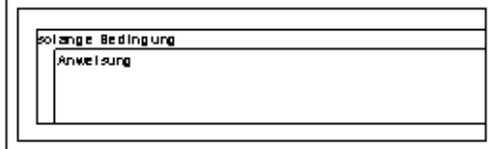
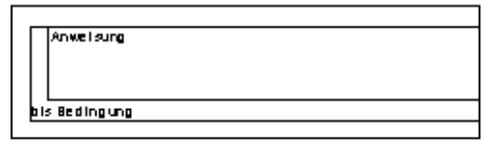
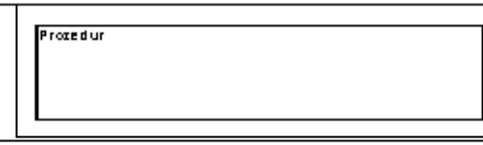




5. Erstelle zu dem Flussdiagramm auf Seite 7 „Telefonieren“ das passende Struktogramm.
6. In einer Prüfung können maximal 120 Punkte erreicht werden. Ab 60 Punkten gilt die Prüfung als bestanden, ab 85 Punkten als gut bestanden und ab 110 Punkten als hervorragend bestanden.
   a) *Schreibe ein Java-Programm, welches bei Eingabe der erreichten Punktzahl ausgibt, wie die Prüfung absolviert wurde.*
   b) *Zeichne das Struktogramm.*
7. Erstelle ein Programm, welches solange eine Passworteingabe anfordert, bis das korrekte Passwort eingegeben worden ist. Hinweis: Du musst das korrekte Passwort zuvor definieren. Zusatzaufgabe: Es sollen nur drei falsche Eingaben möglich sein.

Übersicht: Kontrollstrukturen und Struktogramme

Alltagsbeschreibung / Beispiel	Strukturblock	Java-Struktur	Kommentar
Ziehe deine Schuhe an; Ziehe deine Jacke an; öffne die Haustür; Gehe durch die Haustür; Schließe die Haustür		*Block in geschweifte Klammern* { Anweisung1; Anweisung2; ...; AnweisungN; }	Eine Folge von Anweisungen, die alle der Reihe nach abgearbeitet werden, bezeichnet man als **Sequenz**.
Wenn man das Licht am Auto länger anlässt, dann ist die Batterie leer.		*if-Anweisung* if (Bedingung) { Anweisung1; } else { Anweisung2; }	Mit einer Anweisung der Form *Wenn Bedingung erfüllt* *dann führe Anweisung1 aus* *sonst führe Anweisung2 aus* führt man eine ***Fallunterscheidung*** durch.
Wenn man keinen Fehler im Diktat hat, bekommt man eine 1; wenn man einen Fehler hat, bekommt man eine 2; wenn man zwei Fehler hat... ansonsten bekommt man eine 6 (Abfangen aller weiterer Fehlerzahlen)		*switch-Anweisung* switch (Ausdruck) { case Wert1 : Anweisung1; break; case Wert2 : Anweisung2; break; ... default : Anweisung4	***Mehrfachauswahl*** Der Ausdruck muss ganzzahlig sein. Das Programm wird an der case-Anweisung fortgesetzt, deren Wert dem Ausdruck entspricht. Falls Ausdruck keinem der Werte zugeordnet werden kann, geht es mit der default-Anweisung weiter.

Alltagsbeschreibung / Beispiel	Strukturblock	Java-Struktur	Kommentar
Von einer Liste können nur die ersten 15 Schüler an einer Veranstaltung teilnehmen.		*for-Schleife* for (int i = 1; i <= n; i ++) { Anweisung; }	Eine Anweisung der Form *für Zähler = Anfang bis Ende* *Anweisung* heißt **gezählte Schleife**. Gezählte Schleifen werden dann benutzt, wenn man weiß, wie oft eine Schleife durchlaufen werden muss.
Es können solange Schüler in den Bus einsteigen, solange noch Plätze frei sind.		*while-Schleife* while (Bedingung) { Anweisung; }	Eine Anweisung der Form *Solange Bedingung erfüllt* *führe Anweisung aus* heißt **Schleife mit Eingangsbedingung**. Trifft die Bedingung anfangs nicht zu, so wird die Wiederholungsanweisung nicht ausgeführt.
Ein Fahrschüler muss die Fahrprüfung solange wiederholen, bis er sie bestanden hat.		*do-while-Schleife* do { Anweisung; } while (Bedingung);	Eine Anweisung der Form *Wiederhole Anweisung* *solange Bedingung erfüllt* heißt **Schleife mit Ausgangsbedingung**. Im Unterschied zur While-Schleife wird die zu wiederholende Anweisung mindestens einmal ausgeführt.
		*Prozeduraufruf* Prodedurname (Arg1,.., ArgN);	**Prozeduren** werden über ihren Namen aufgerufen. In Klammern kann man Argumente übergeben.

Aufgaben:

8.

```
1  import java.util.*;
2
3  public class Teiler {
4
5    public static void main(String[] args) {
6       int zahl;
7       Scanner sc = new Scanner(System.in);
8       System.out.println("Geben Sie eine Zahl ein.");
9       zahl = sc.nextInt();
10      System.out.print(1+"  ");
11      for (int teiler = 2; teiler <= zahl; teiler++) {
12         if (zahl % teiler == 0) {
13            System.out.print(teiler+"  ");
14         }
15      }
16   }
17
18 }
```

a) Was passiert, wenn man die Zahl 12 eingibt? Was passiert, wenn man die Zahl 9 eingibt?
b) Erläutere, was das Programm allgemein leistet.
c) Zeichne das zugehörige Struktogramm.

9.

a) Erläutere, warum es sich hier um eine Endlosschleife handelt.
b) Ergänze den Quellcode so, dass es sich nicht mehr um eine Endlosschleife handelt.

```
1  public class Programm8 {
2
3    public static void main(String[] args) {
4       int ende = 0;
5
6       while (ende < 10) {
7         System.out.println("Das nimmt kein Ende...");
8       }
9    }
10 }
```

10. Stefans Mutter trinkt gern Tee. Verschiedene Teesorten haben verschiedene Ziehzeiten. Oft vergisst Stefans Mutter den Teebeutel rechtzeitig aus ihrer Tasse zu nehmen und ärgert sich anschließend, weil ihr Tee nicht den gewünschten Geschmack hat. Stefan hat sich den folgenden Algorithmus überlegt:

Lege die Variablen minuten und sekunden vom Typ Integer an.
Fordere den Benutzer mit „Wie viele Minuten?“ auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in der Variablen minuten.
*Weise der Variablen sekunden den Wert minuten*60 zu.*
Mache Folgendes so lange, bis sekunden = 0 wahr ist:
    Gib „Noch“ und danach den Wert von sekunden aus.
    Gib den Text „Sekunden“ aus und wechsele in die nächste Zeile.
    Warte mit der Abarbeitung des Programms eine Sekunde.
    Weise der Variablen sekunden den Wert sekunden-1 zu.
Gib den Text „Der Tee ist fertig“ aus.

a) Übertrage den Algorithmus in ein Programm.
b) Modifiziere das Programm so, dass die Ziehzeit direkt in Sekunden eingegeben werden kann.

11. Franka möchte sich ein neues Fahrrad für 530 € kaufen. Ihr Opa hat angeboten, ihr das benötigte Geld zinsfrei zu leihen. Franka möchte ihrem Opa jeden Monat etwas von ihrem Taschengeld zurückzahlen.
    Sie hat einen Plan für die Rückzahlung geschrieben:

    Kredit: 530 €

Jahr	Monat	Rückzahlung	Restkredit
2014	7	35 €	495 €
2014	8	35 €	460 €
2014	9	35 €	425 €
…			

    a) Fertigen einen schriftliche Entwurf für ein Programm an, bei dem für einen zinslosen Kredit nach der Eingabe der Kredithöhe, des Startjahres, des Startmonats sowie der monatlichen Zahlung ein Plan für die Rückzahlung ausgegeben wird.
    b) Implementiere das entworfene Programm benutzerfreundlich.
    c) Teste dein Programm mit folgenden Eingaben:
        a. Kredit: 200 €, Rückzahlung: 20 € Jahr 2015, Monat 11
        b. Kredit: 100 €, Rückzahlung: 30 € Jahr 2014, Monat 3
        c. Frankas Kredit

    Notiere die Ausgaben des Programms für alle drei Testfälle.
    d) Reflektiere deinen Lösungsweg: Welche Schwierigkeiten traten beim Entwurf und bei der Implementierung auf? Wie wurden die Probleme gelöst?
    e) für die Schnellen: Frankas Opa vereinbart mit ihr, dass er ihr jedes Jahr zu Weihnachten 70 € von dem Kredit erlässt. Erweitere dein Programm entsprechend.
    f) Sende das Programm an mich.

Das Feld (array): Ein wichtiger Datentyp

Häufig werden in Computerprogrammen viele gleichartige Daten gespeichert, z. B. Telefonnummern, Messwerte, Rechnungsdaten etc. Um eine größere Anzahl gleicher Daten besser speichern und verarbeiten zu können, verwendet man das Feld.

Ein Feld (engl. **array**) ist eine **Reihe von Elementen gleichen Datentyps**. Die einzelnen Elemente können über einen Index angesprochen werden. In Felder kann man eine begrenzte Anzahl gleichartiger Daten speichern. Im Gegensatz zu einfachen Datentypen wie int, float, char oder boolean hat ein Feld einen strukturierten Datentyp. Die Struktur eines Feldes wird im Bild deutlich.

Zahl[0]	Zahl[1]	Zahl[2]	Zahl[3]	...	Zahl[i]	Zahl[i+1]	...	Zahl[n-1]	Zahl[n]
17	13	7	42		46				

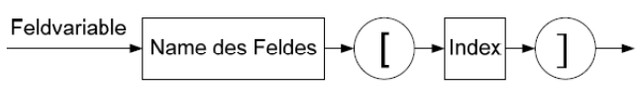
Wir sehen ein Feld ganzer Zahlen. Jede Zahl befindet sich in einer Zelle des Feldes. Alle Zellen sind gleich groß und können genau eine ganze Zahl aufnehmen. Nicht alle Zellen des Feldes sind belegt, es können noch weitere Zahlen im Feld gespeichert werden. Das erste Feldelement ist 17 und hat den Index 0 (**Merke: Computer zählen ab Null, Menschen zählen ab Eins**), das zweite Element ist 13 und kann über den Index 1 angesprochen werden. Das letzte Feldelement ist 46.

Deklariert werden Felder durch Kennzeichnung der Feldvariablen mit dem Klammernpaar []. In der Variablen Elemente merken wir uns, wie viele Elemente im Feld tatsächlich gespeichert sind. Diese Variable braucht man nicht, wenn das Feld immer komplett gefüllt ist, was häufig in Programmen vorkommt.


```
long[] Zahl;
int Elemente;
```

Zum Anlegen eines Feldes benötigt man die **new-Methode,** da es sich bei einem Array um ein Objekt handelt. Im Beispiel wird das Feld zur Aufnahme von maximal 100 Elementen angelegt.

```
Zahl = new long[100];
```



Auf ein einzelnes Feldelement greift man über den Index zu:

```
Zahl[7] = 15;

for (int i= 7; i < 20; i++) {
   Zahl[i]= Math.round(50*Math.random());
   System.out.println(Zahl[i]);
}
```

Die **Kapazität eines Feldes** ermittelt man mit der Methode length:

```
System.out.println("Maximales Fassungsvermögen: " + Feld.length);
```

Einfügen, Suchen und Löschen eines Elements, sowie Anlegen, Füllen und Sortieren eines Feldes sind Standardoperationen auf Feldern. Mit einem Java-Programm sollen diese Operationen realisiert werden.[17] Arrays kannst du verwenden, wenn du eine schnelle, geordnete, effiziente Liste für Dinge gleichen Typs haben möchtest, von denen du vorher weißt, wie viele Dinge es maximal sein werden. Arrays bieten über die Indexposition einen schnellen Zugriff auf die Elemente.

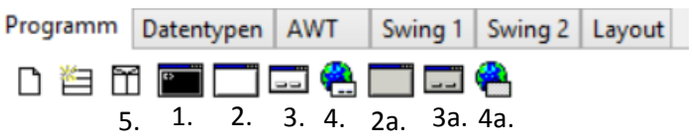
Aufgaben:

1. Schreibe ein Konsolenprogramm, das ein Feld der Kapazität 20 mit 15 Zufallszahlen füllt und dann ausgibt: die größte Zahl, die kleinste Zahl, den Mittelwert der Zahlen, die Zahl, die dem Mittelwert am nächsten kommt.
2. Schreibe ein Konsolenprogramm, das ein Feld mit 100 ganzzahligen Zufallszahlen von 1 und 9 füllt und dann die Häufigkeitsverteilung dieser Zahlen ermittelt und ausgibt.

[17] vgl. (Röhner G. , Unterrichtsskript: Einführung in das Programmieren, 2007/2008)

Programme mit GUI

Oberflächenprogrammierung mit dem Java-Editor



Der Java-Editor bietet verschiedene Möglichkeiten, wenn man ein neues Programm erstellen möchte.

1. **Konsole:** Hier kann man keine Benutzeroberfläche erstellen, alle Ein- und Ausgaben können nur über das cmd-Fenster betrachtet werden. Das ist nicht sonderlich ästhetisch.
2. **Frame / JFrame**: Ein Fenster hat einen Rahmen und eine Titelleiste. Außerdem kann es Buttons zum Schließen, Minimieren und Maximieren von sich selbst haben.
3. **Dialog / JDialog**: Ein Dialogfenster ist meist an ein anderes Fenster gebunden. Wird das übergeordnete Fenster geschlossen, so schließt auch der Dialog. Man kann den Dialog so programmieren, dass man erst wieder in das übergeordnete Fenster wechseln kann, wenn alle „Arbeiten“ im Dialog erledigt sind. Das Dialogfenster besitzt keine Buttons zur Minimierung bzw. Maximierung.
4. **Applet / JApplet**: Möchte man ein Java-Programm in eine Internetseite einbetten, so verwendet man Applets.
5. **Struktogramm**: Hier kann man das oben genannte Struktogramm zeichnen.

Welche Bedeutung hat das J?

Die erste Möglichkeit eine bedienungsfreundliche Oberfläche zu programmieren wurde bereits mit JDK 1.0 geliefert. Hierbei handelt es sich um die **Klassenbibliothek AWT (= Abstract Window Toolkit)**. Zur Darstellung greift AWT auf die Funktionen des Betriebssystems zu, auf dem das Java-Programm gerade läuft. Vorteil dieser Programmierung ist, dass der Anwender die Komponenten in der gewohnten Darstellung vorfindet. Die Komponenten des AWT werden auch **schwergewichtig** genannt. Die Ergebnisbehandlung ist allerdings nicht optimal und insgesamt weist die Klasse Schwächen auf.

Ab JDK 1.2. wird zusätzlich die **Klassenbibliothek Swing** mitgeliefert. Wie man alleine an der Abbildung erkennt, gibt es wesentlich mehr Swing-Komponenten als AWT-Komponenten, die die Arbeit des Programmierers erleichtern. Bei den Swing-Komponenten handelt es sich um **leichtgewichtige** Komponenten, d. h. die Komponenten werden auf dem Bildschirm dargestellt ohne auf die Funktionen des Betriebssystems zurückzugreifen. Dies bietet für den Programmierer den Vorteil, dass das Aussehen seines Programms immer gleich ist, egal auf welchem Betriebssystem es läuft.

Was können Swing-Komponenten mehr als AWT-Komponenten:

- Tabellen oder Bäume
- Schaltflächen und Labels nehmen Symbole auf, die sie beliebig um den Text angeordnet darstellen
- Swing-Komponenten können transparent und beliebig geformt sein; eine Schaltfläche kann wie unter Mac OS X abgerundet sein.

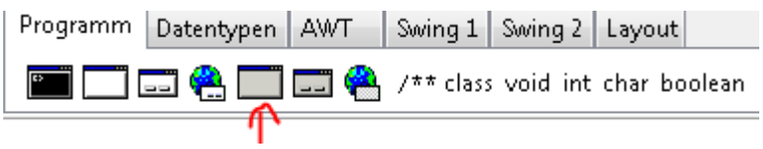
- Jede Swing-Komponente kann einen Rahmen bekommen.

Man sollte keine AWT-Komponenten mit Swing-Komponenten vermischen. Um diese Gefahr auszuschließen, kann man beim Java-Editor die AWT-Komponenten ausblenden, so dass man nur mit den Swing-Komponenten arbeiten kann.

Programme mit einer grafischen Oberfläche (GUI) erstellen

Die Ausgabe auf der Konsole hat natürlich so seine (ästhetischen) Grenzen, darum wollen wir uns jetzt an ein Programm mit einer grafischen Oberfläche (GUI = graphical user interface) wagen.

Der Java-Editor nimmt uns dabei eine Menge Schreibarbeit ab. Wir starten ein neues GUI-Programm mit dem Symbol:



Wir speichern die Datei unter dem Namen GUI_Programm1 und der Java-Editor erzeugt im Hintergrund gleich eine neue Programmdatei, die so aussieht (ohne Erläuterungskommentare):

```
1  import java.awt.*;            //Java-Klassen, die für die graphische Oberfläche
2  import java.awt.event.*;      //nötig sind; Zugriff auf AWT-Komponenten
3  import javax.swing.*;         //in JFrame: zusätzlicher Zugriff auf die
4  import javax.swing.event.*;   //Swing-Kompnenten
5
6  public class GUI_Programm1 extends JFrame {  //von JFrame abgeleitet
7    // Anfang Attribute              //diese Kommentare müssen stehen bleiben!
8    // Ende Attribute
9
10   public GUI_Programm1(String title) {     //Konstruktor
11     // Frame-Initialisierung
12     super(title);                          //erzeugt ein Fenster mit Titel
13     setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
14     int frameWidth = 300;                  //Fenstergröße wird festgelegt
15     int frameHeight = 300;
16     setSize(frameWidth, frameHeight);
17     Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
18     int x = (d.width - getSize().width) / 2;
19     int y = (d.height - getSize().height) / 2;
20     setLocation(x, y);
21     setResizable(false);
22     Container cp = getContentPane();
23     cp.setLayout(null);
24     // Anfang Komponenten               //Aufbau der GUI
25
26     // Ende Komponenten
27
28     setVisible(true);
29   } // end of public GUI_Programm1
30
31   // Anfang Methoden
32   // Ende Methoden
33
34   public static void main(String[] args) {  //Hauptprogramm
35     new GUI_Programm1("GUI_Programm1");      //neues Objekt der Klasse erzeugen
36   } // end of main
37
38 } // end of class GUI_Programm1
39
```

Der Quellcode enthält unter anderem die drei noch leeren Bereiche Attribute, Komponenten und Methoden.

- *Attribute*: Hier werden die benötigten Variablen deklariert.
- *Komponenten*: In diesem Abschnitt werden die Größe und Beschriftung der Swing-Komponenten gesetzt, der Schalter dem Programm-Panel Cup (Content-panel) zugeordnet und eine Ereignisprozedur für das Anklicken des Schalters implementiert.
- *Methoden*: Hier werden die Prozeduren, u. a. die Ereignisprozeduren programmiert.

Das Programm können wir schon kompilieren und ausführen. Es öffnet sich ein leeres Fenster.

GUI- Erstellen mit Eingabe-/ Ausgabefeldern, Button etc.

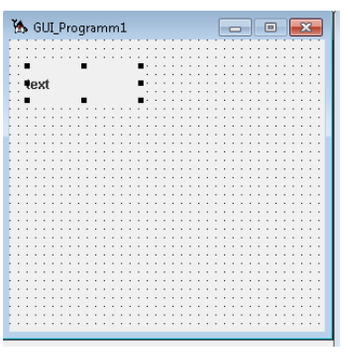
Daher werden wir jetzt ein Textfeld einfügen, in welches der Benutzer seinen Namen eingeben kann und einen Button, auf den er klicken kann.

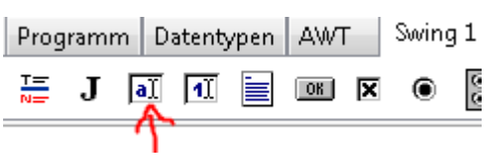
Zuerst wählen wir in der Symbolleiste Swing1 ein Anzeigefeld (=Label) aus:

Und ziehen auf der Programmoberfläche (an den Punkten zu erkennen) einen Rahmen auf.

Im Objektinspektor können wir jetzt für jedes Label einen Text eingeben: „Wie heißt du?“

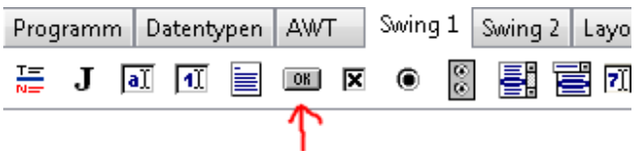
Wir können auch eine andere Schriftart („Font“) auswählen, Schriftfarbe, …

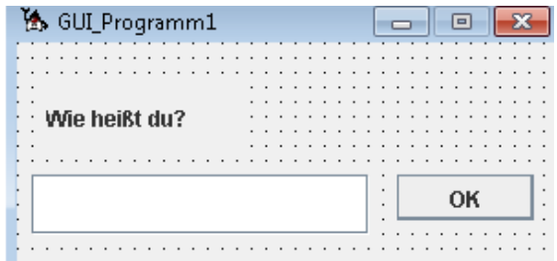


Nun brauchen wir ein 

Eingabefeld (=JTextField):

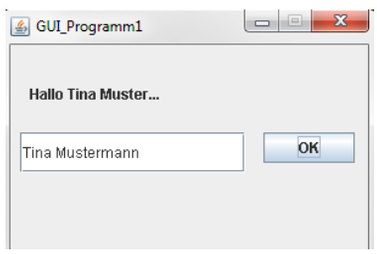
Und zuletzt benötigen wir noch einen Schalter (=JButton) mit der Aufschrift (=text) „OK“:





Jetzt kommt noch ein wichtiger Punkt: Wenn der Benutzer seinen Namen eingegeben und auf „OK“ geklickt hat, wollen wir ihn mit seinem Namen begrüßen, d. h. der Text unseres Anzeigefeldes (JLabel1) soll geändert werden.

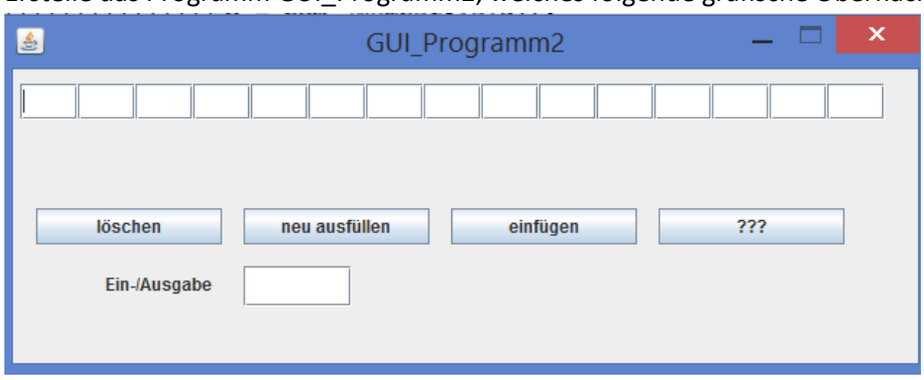
Das passiert in der sogenannten Ereignisprozedur. Wir machen einen Doppelklick auf den Button und springen damit wieder in den Programmtext und zwar in eine neu erzeugte Methode namens jButton1ActionPerformed(ActionEvent evt). Der Name der Methode kommt vom Namen des Buttons, der heißt jButton1. Bei aufwendigeren Programmen sollte man alle Button mit aussagekräftigen Namen versehen! Innerhalb der geschweiften Klammer muss nun noch der Programmcode ergänzt werden. Mit der Methode getText() wird der Name aus dem Eingabefeld jTextfield1 ausgelesen. Mit der Methode setText() wird der neue Text für das Anzeigefeld jLabel1 gesetzt.




```
// Anfang Methoden
public void jButton1_ActionPerformed(ActionEvent evt) {
  String name =jTextField1.getText();
  jLabel1.setText("Hallo "+name+"!");
} // end of jButton1_ActionPerformed
```

Aufgaben:

3. Erstelle das Programm GUI_Programm1, indem du den Anweisungen genau folgst.
   Führe das Programm aus.
4. Betrachte nun den Quellcode, du wirst an einigen Stellen Veränderungen vorfinden, die das Programm automatisch beim Einfügen der Swing-Komponenten getätigt hat.
5. Erstelle das Programm GUI_Programm2, welches folgende grafische Oberfläche hat:[18]



   zu beachten:

   - Unter den Schaltern gibt es eine jNumberField-Komponente, um eine Zahl bequem ein- und ausgeben zu können.

[18] Nach einer Idee von G. Röhner

- Da es mühsam ist, 15 Nummernfelder einzeln zu platzieren und zu verwalten, sollen die Zahlenfelder vom Programm erzeugt und im Fenster platziert werden. Die 15 Nummernfelder sollen mit einer for-Schleife erzeugt werden. (//Anfang Komponenten):

```
// Anfang Komponenten
for (int i = 0;i<15 ;i++ ) {
  JNumberField jNFZahlenfeld = new JNumberField();
  jNFZahlenfeld.setBounds(5+i*40, 10, 40, 25);
  cp.add(jNFZahlenfeld);
} // end of for
```

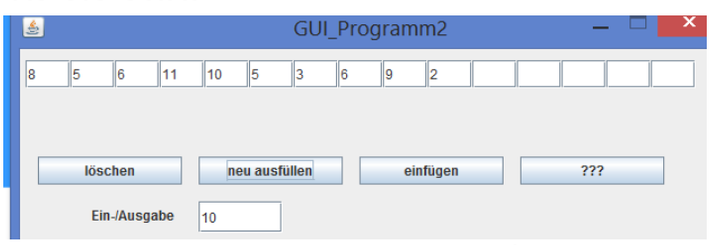
  Der Nachteil liegt darin, dass wir nach der Anzeige im GUI-Formular keinen Zugriff mehr auf die Nummernfelder haben, da die Variable jNFZahlenfeld nur in der Schleife deklariert ist. Die jNFZahlenfelder dienen also nur der Darstellung der Daten für den Anwender.
- Zur Verwaltung der 15 Nummernfelder brauchen wir ein weiteres Feld (array), wobei jedes Element des Feldes ein Nummernfeld ist:

```
private JNumberField[] verwaltungsfeld = new JNumberField[15];
// Ende Attribute
```

- Nun speichern wir jedes jNFZahlenfeld im verwaltungsfeld ab, indem wir die for-Schleife um folgende Zeile ergänzen:
  Verwaltungsfeld[i] = jNZahlenfeld

6. Erweitere den Variablenbereich (= Attribute) des GUI_Programm2 um ein Feld von int-Zahlen der Kapazität 15. Die Zahl der tatsächlich sich darin befindlichen Zahlen soll in der Variablen „elemente“ gespeichert werden.

7. Durch diese Vorgehensweise haben wir die Verarbeitung der Daten von der Darstellung getrennt. Die visuellen Komponenten dienen der Ein- und Ausgabe. Die Verarbeitung findet auf der internen Datenebene statt.



Darstellung der Daten (im jNFZahlenfeld[] über verwaltungsfeld[] erreichbar)

ausgeben

eingeben

8 5 6 11 10 5 3 6 9 2 **interne Daten im „rechenfeld“**

Für die bequeme Arbeit mit diesen beiden Ebenen ist es sinnvoll, Prozeduren zu schreiben, die für den Datenaustausch sorgen.
Zunächst benötigen wir eine Methode „eingeben()“, die die Zahl ausliest und an das Programm übergibt, die im Eingabefeld unterhalb der Buttons eingetippt worden ist.

```
// Anfang Methoden
//Methode liest den Wert aus dem Eingabefeld aus und gibt ihn zurück
public int eingeben(){
  try {
    return jNFEingabe.getInt();
  } catch(NumberFormatException e) {
    return -1;
  }  // end of try
}
```

Die Methode ist parameterlos, d. h. es wird nichts übergeben (leere runde Klammer). Sie liefert einen int-Wert zurück. Zum Abfangen von Fehl-Eingaben, z. B. leeres Eingabefeld, unzulässiges Zahlenformat oder nicht numerisches Zeichen, wird die try-Anweisung verwendet. Sie sorgt dafür, dass auftretende Fehler erkannt und sinnvoll verarbeitet werden können.
Ergänze dein GUI2_Programm um die Methode eingeben() und ausgeben().

ausgeben()

wiederhole für i=0 bis elemente
verwaltungsfeld[i].setInt(rechenfeld[i]);
wiederhole für i=elemente bis 15
verwaltungsfeld[i].setText("");

8. Die Schalter sollen zum Leben erweckt werden.
   Bei einem Java-Programm müssen sich die Objekte, die auf Ereignisse reagieren, anmelden. Ein Schalter tut dies mittels addActionListener. Der dabei angegebene ActionListener enthält eigentlich nur die Ereignismethode actionPerformed, welche beim Anklicken ausgeführt wird. Der benötigte Programmcode wird vom Java-Editor automatisch erstellt. Im Abschnitt Methoden muss dann nur noch implementiert werden, was beim jeweiligen Buttonklick passieren soll. Wenn man einen Doppelklick auf jeweiligen Button ausführt, gelangt man an die richtige Stelle im Quellcode.
   a. Beim Anklicken des Löschen-Schalters sollen alle Felder gelöscht werden. Beachte, dass sowohl die visuelle Ausgabe als auch die internen Daten gelöscht werden müssen.
   b. Beim Anklicken des NeuAusfüllen-Schalters sollen so viele Zufallszahlen im Feld erzeugt und angezeigt werden, wie im Eingabe-Textfeld angegeben wird.

jBu_NeuFuellen

elemente = eingeben();
wiederhole für i=0 bis elemente
rechenfeld[i]=(int)Math.round(1+10*Math.random());
ausgeben

   c. Ein Klick auf den Einfügen-Schalter soll die Zahl im Eingabefeld am Ende des Feldes einfügen, falls das Feld noch nicht voll ist.
   d. Für den letzten Schalter ist folgendes Struktogramm gegeben.

jBu_Fraglich_ActionPerformed(ActionEvent evt)

zahl = eingeben();	
i = 0;	
wiederhole solange (i<elemente und rechenfeld[i] ungleich zahl	
i++	
ja — i<elemente — nein	
wiederhole für k=i bis elemente-1	∅
rechenfeld[k] = rechenfeld[k+1];	
elemente = elemente-1;	
ausgeben();	

   Erläutere, was dieser Button leistet?

Turtle-Programmierung

„Mit **Turtle-Grafik**, auch **Igelgrafik**, wird eine Bildbeschreibungssprache bezeichnet, bei der man sich vorstellt, dass ein stifttragender Roboter (die Schildkröte, engl. „turtle“) sich auf der Zeichenebene bewegt und mit einfachen Kommandos, wie Stift heben, senken, vorwärts laufen und drehen, gesteuert werden kann. Diese Idee wurde mehrfach realisiert, zum Beispiel als Steuersprache für Stiftplotter (HPGL), als Teil der Programmiersprache für Heimcomputer (Basic, Pascal auf Amiga, Atari) und als Grundidee der pädagogischen Programmiersprache LOGO.“[19] Auch in den Java-Editor ist die eine Turtle-Komponente eingebettet.

Die Turtle im Java-Editor starten

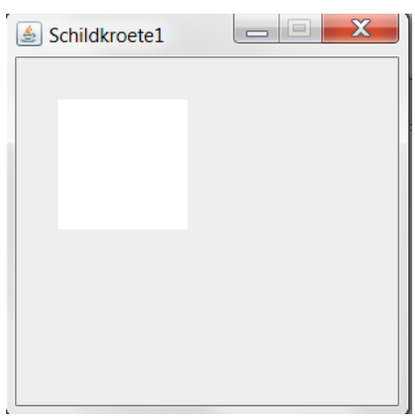
Um mit der Schildkröte Spuren auf den Bildschirm zu zeichnen, starten wir zunächst ein JFrame-Programm (siehe vorhergehende Seite). Als nächstes wählen wir die Swing-Komponente Turtle aus und fügen sie der GUI hinzu. Dadurch werden folgende Zeilen im Quelltext automatisch ergänzt:

```
public class Schildkroete1 extends JFrame {
  // Anfang Attribute
  private Turtle turtle1 = new Turtle(100, 100);
  // Ende Attribute
```

Es wird eine neue Turtle mit dem Namen turtle1 (Name lässt sich im Objektinspektor ändern) erzeugt, die eine Zeichenfläche von 100 x 100 (Width x Height) besitzt.

```
// Anfang Komponenten

turtle1.setBounds(32, 32, 100, 100);
cp.add(turtle1);
// Ende Komponenten
```

Führt man dieses Programm nun aus, erscheint folgendes Fenster:



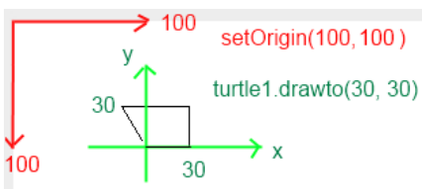
Das helle Quadrat zeigt den Zeichenbereich der Turtle an. Zu Beginn befindet sich die Turtle in der Mitte und schaut nach rechts. Die Position der Turtle wird demnach durch ihre x- und y-Koordinate beschrieben. Die aktuelle Bewegungsrichtung wird mithilfe eines Winkels w angegeben, wobei 0° eben angibt, dass die Turtle nach rechts schaut. Lässt man die Turtle sich jetzt bewegen, so hinterlässt sie ihre Spur auf dem Bildschirm. Um die Turtle zu bewegen, stehen folgende Methoden zur Verfügung:

[19] http://de.wikipedia.org/wiki/Turtle-Grafik (19.02.2013, 12:45 Uhr)

turn (double angle)	Ändert die Zeichenrichtung relativ um den Winkel angle.
turnto (double angle)	Legt die Zeichenrichtung absolut fest
draw (double length)	Die Turtle zeichnet in der aktuellen Richtung eine Strecke der Länge *length*.
drawto(double x, double y)	Die Turtle zeichnet von der aktuellen Position eine Strecke zum Punkt P(x/y) im **grünen** Koordinatensystem
move (double length)	Bewegt die Turtle in der aktuellen Zeichenrichtung um eine Strecke der Länge *length* ohne zu zeichnen.
moveto (double x, double y)	Setzt die Turtle auf den Punkt P(x/y)
setForeground(Color c)	Setzt die Farbe c als Zeichenfarbe
setBackground(Color c)	Setzt die Farbe c als Hintergrundfarbe
clear()	Löscht die Zeichenfläche.
setOrigin(int x, int y)	Legt den Ursprung des **grünen** Koordinatensystems der Turtle auf der Zeichenfläche fest. Die Zeichenfläche hat ihren eigenen Ursprung immer links oben im **roten** Koordinatensystem.

(Röhner G. , Informationsmaterial zur MNU-Landestagung, 2012)

```
turtle1.setOrigin(100, 100);
turtle1.drawto(30, 0);
turtle1.drawto(30, 30);
turtle1.drawto(-20, 30);
turtle1.drawto(-5, 5);
```



Einfaches Zeichnen

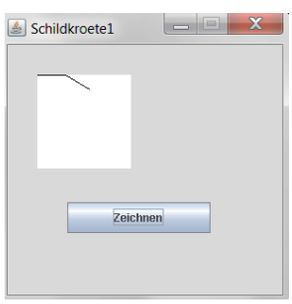
Für die Programmsteuerung fügen wir noch einen Button „Zeichnen“ hinzu. Der Button soll den Namen jBu_Zeichnen erhalten.

Das erste Programm besteht nur aus der hintereinander Ausführung der folgenden drei Anweisungen: Gehe 30 Schritte vorwärts, drehe dich um 30° nach rechts und gehe wieder 30 Schritte vorwärts.

30 Schritte vorwärts
30° nach rechts drehen
30 Schritte vorwärts

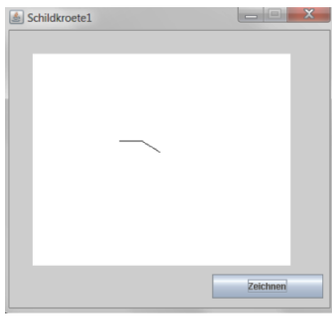
Der zugehörige Quelltext und die Ausgabe sehen wie folgt aus, wobei die erzeugte Turtle den Namen Tina trägt:

```
// Anfang Methoden
public void jBu_Zeichnen_ActionPerformed(ActionEvent evt) {
    tina.draw(30);
    tina.turn(-30);
    tina.draw(30);
} // end of jBu_Zeichnen_ActionPerformed
```



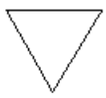
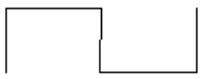
Optisch sind nun noch zwei Änderungen ratsam. Die Zeichenfläche insgesamt ist ebenso wie das Fenster recht klein, daher sollten die Fenstergröße und die Größe der Zeichenfläche verändert werden. Diese Änderungen können entweder direkt im Quellcode vorgenommen werden, indem die entsprechenden Werte vergrößert werden, dann muss man jedoch auch die Position des Button verändern, oder man zieht das Fenster und die Zeichenfläche in der Programmoberfläche verändert.

Anschließend verwenden wir noch den Befehl setOrigin(x,y) mit passenden Werten, so dass unsere tina-Turtle möglichst in der Mitte beginnt. Dann erhalten wir folgendes Bild:



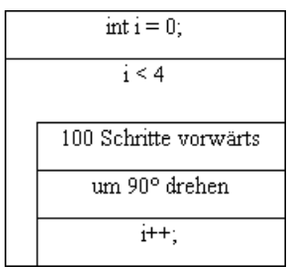
Aufgaben:

1. Zeichne folgende Figur:

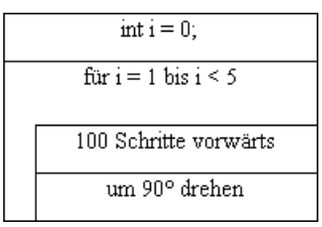
a.  b.  c) 

Zeichnen mit Schleifen

Das Zeichnen eines Quadrates kann mithilfe einer while-Schleife programmiert werden. Das zugehörige Struktogramm sieht wie folgt aus:

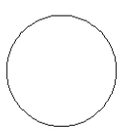


Analog könnte man das Quadrat auch mit einer for-Schleife programmieren.



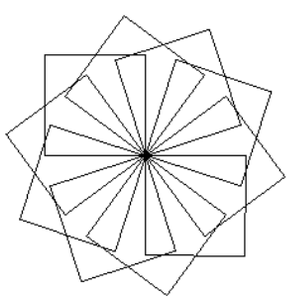
Man erkennt schon an der Struktur, dass sich jede while-Schleife in eine for-Schleife umwandeln lässt.

Das Zeichnen eines Kreises führt jedoch auf diese Art und Weise zu einer Endlosschleife, wie folgender Quellcode zeigt:



```
public void jBu_Zeichnen_ActionPerformed(ActionEvent evt) {
  tina.setOrigin(120,120);
  while (true) {
    tina.draw(3);
    tina.turn(-3);
  } // end of while
} // end of jBu_Zeichnen_ActionPerformed
```

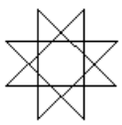
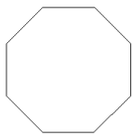
Verschachtelt man zwei for-Schleifen, so kann man interessante Figuren zeichnen lassen.



```
public void jBu_Zeichnen_ActionPerformed(ActionEvent evt) {
  tina.setOrigin(160,160);
  for (int i = 0; i < 10 ;i++ ) {
    for(int j = 0; j < 4; j++) {
      tina.draw(80);
      tina.turn(-90);
    } //end of for j
    tina.turn(36);
  } // end of for i
} // end of jBu_Zeichnen_ActionPerformed
```

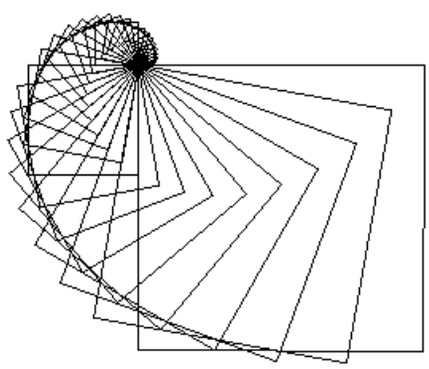
Aufgaben:

2. Zeichne folgende Figuren:

a.  b.  c.  d. 

3. Mithilfe einer bedingten Verzweigung und der Methode setForeground(Color c) soll die Zeichenfarbe so geändert werden, dass jede Trittfläche der Stufen von 2d rot ist und die senkrechte Kante grün ist. Hinweis: tina.setForeground(Color.RED).

Variablen und Methoden verwenden



Es sollen Quadrate gezeichnet werden. Das erste Quadrat hat eine Seitenlänge von 180. Nach jedem gezeichneten Quadrat soll die Seitenlänge auf 90 % verkleinert werden. Das nächste Quadrat wird um 10° gedreht gezeichnet. Der Vorgang soll solange fortgesetzt werden, solange die Seitenlänge größer als 5 ist.

Um diese Aufgabe zu realisieren, müssen wir für die Seitenlänge eine Variable einführen (double a) und können das Zeichnen eines Quadrates in eine eigene Methode zeichneQuadrat schreiben. Der Quelltext wird dadurch übersichtlicher und man kann die Methode später noch weiter verwenden.

Schreiben wir zunächst die eigene Methode:

```
public void zeichneQuadrat(double seitenlänge) {
  for (int i = 0 ; i < 4 ; i++) {
    tina.draw(seitenlänge);
    tina.turn(-90);
  } // end of for
}
```

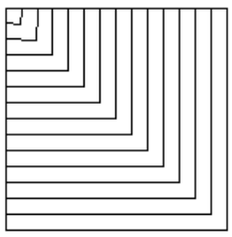
Die Methode, die auf das Klicken des Zeichen-Button reagiert, sieht nun so aus:

```
public void jBu_Zeichnen_ActionPerformed(ActionEvent evt) {
  double a = 180;
  tina.setOrigin(100,80);
  while (a > 5) {
    zeichneQuadrat(a);     //Aufruf der Methode mit der Variablen a
    tina.turn(-10);        //kleine Drehung
    a = a * 0.9;           //verkleinern auf 90 %
  } // end of while
} // end of jBu_Zeichnen_ActionPerformed
```

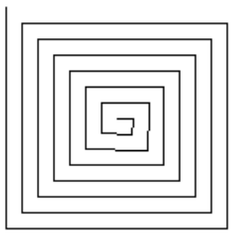
Aufgaben:

4. Schreibe ein Javaprogramm, welches solange Quadrate zeichnet, bis die Seitenlänge von 180 auf 200 angewachsen ist, wobei die Seitenlänge nach dem Zeichnen eines Quadrates immer um eins erhöht werden soll. Zudem soll das nächste Quadrat immer um 70° gegen den Uhrzeigersinn gedreht werden.

5. Zeichne folgende Figuren:



a.



b.

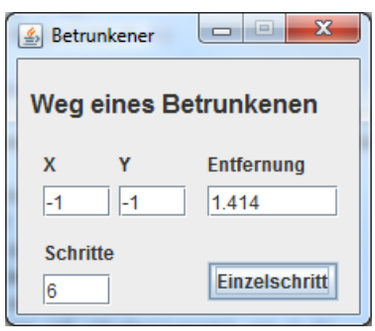
c. Zusatz: Lasse die Spirale bunt zeichnen.

Simulationen

„Simulationen sind ein wichtiges Anwendungsgebiet der Informatik. Neue Produkte werden am Computer entworfen und ihre Eigenschaften im Modell durch Simulationen überprüft, bevor erste Prototypen hergestellt werden. Sind keine analytischen Methoden für die Simulation verfügbar, so kann man Zufallszahlen verwenden (Monte-Carlo-Methode)." (Röhner G. , Informationsmaterial zur MNU-Landestagung, 2012, S. 1)

Der Weg eines Betrunkenen

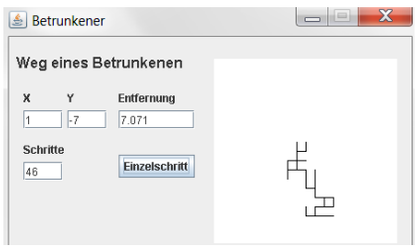
„Ein Betrunkener verlässt morgens früh den Club an der Position (0/0) und wankt bei jedem Schritt zufällig in eine der vier Himmelsrichtungen. Wie weit ist er nach n Schritten vom Club entfernt? Die aktuelle Position des Betrunkenen halten wir in den Variablen x und y fest. Zur Ausführung eines Einzelschritts bestimmen wir mittels Math.random() eine Zufallszahl zwischen 0 und 1. Ist diese kleiner gleich 0.25 wird die x-Position um 1 vergrößert, liegt sie zwischen 0.25 und 0.5 wird die x-Position um 1 verkleinert. Für Werte größer 0.5 wird entsprechend die y-Position verändert. Damit die Werte für x, y und Schritte zwischen zwei Einzelschritten erhalten bleiben, müssen diese außerhalb der Ereignisprozedur definiert werden. Die Entfernung kann über den Phythagoras und Math.sqrt(..) (square root = Quadratwurzel) berechnet werden." (Röhner G. , Informationsmaterial zur MNU-Landestagung, 2012, S. 1)




```
int x = 0;
int y = 0;
int Schritte = 0;


// Anfang Methoden
public void bEinzelschritt_ActionPerformed(ActionEvent evt) {

  double Zufallszahl = Math.random();

  Schritte++;
  if (Zufallszahl <= 0.25)
    x = x + 1;
  else if (Zufallszahl <= 0.5)
    x = x - 1;
```

Aufgaben:

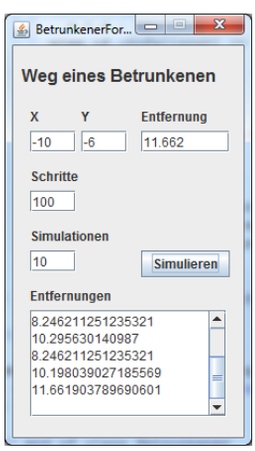
1. Erstelle das GUI-Formular und vervollständige die Ereignisprozedur für den Einzelschritt.
2. Füge dem GUI-Formular eine Turtle hinzu und lasse dort jeden Einzelschritt mithilfe von turtle1.drawto(10*x, 10*y) zeichnen. Das Ergebnis könnte so aussehen:



3. Es ist mühsam, wenn man jeden Schritt einzeln durch Klicken auf den Button simulieren muss. Daher soll unser Programm abgeändert werden. Der Anwender soll zuvor angeben, wie viele Schritte er simulieren möchte. Diese Anzahl kann er im JNumberField (mit dem Namen nfSchritte) unter Schritte eingeben. Anschließend wird nun ja kein Einzelschritt ausgeführt, so dass wir die Beschriftung des Buttons ändern müssen. Der Button wird in „Simuliere“ umgeändert. Führe diese Änderungen durch und speichere das entstandene Programm unter einem neuen Namen. Nun muss noch die Anzahl der Schritte mithilfe einer for-Schleife simuliert werden. Hierzu sind Änderungen in der Methode nötig, die ausgelöst wird, wenn der Button angeklickt wird:

```
// Anfang Methoden
public void bEinzelschritt_ActionPerforme
  turtle1.setOrigin(150, 150);
  x = 0;
  y = 0;
  Schritte = nfSchritte.getInt();

  for (int i = 1; i <= Schritte; i++) {
    double Zufallszahl = Math.random();
    if (Zufallszahl <= 0.25)
      y = y + 1;
```

4. **„Mehrere Simulationen**



Wir können nun zwar problemlos 100 oder 1000 Schritte ausführen lassen, aber was können wir aus einer Simulation erkennen? Recht wenig, man muss schon mehrmals simulieren, um die Ergebnisse interpretieren zu können. In einem nächsten Entwicklungsschritt erweitern wir das Programm so, dass es eine gewünschte Anzahl von Simulationen ausführen kann. Eine 100-Schritt-Simulation wird im Beispiel zehnmal ausgeführt. Dies erreicht man durch eine weitere for-Schleife um die bisherige for-Schleife. Zur Anzeige der Simulationsergebnisse setzen wir eine TextArea-Komponente ein, welche mehrere Textzeilen anzeigen kann. Im Beispiel sieht man die Ergebnisse der fünf letzten Simulationen. Wenn die TextArea-Komponente den Namen taAusgabe hat, so wird eine Zeile ausgegeben mit:

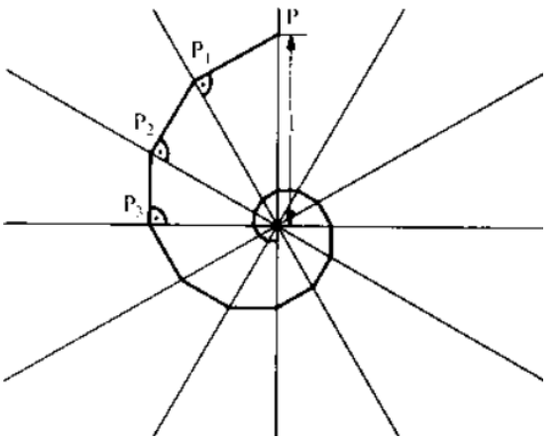
```
taAusgabe.append(Entfernung + "\n");
```

Implementiere diese Variante.

Zusatzaufgabe: Vergleiche dein Ergebnis mit dem im Wikipedia-Artikel zu *Random Walk*.“ (Röhner G. , Informationsmaterial zur MNU-Landestagung, 2012, S. 3)

Gemischte Übungen zur Turtle-Programmierung:

1. Lass deine Turtle das Haus vom Nikolaus in einem Zug zeichnen. Erweitere dein Programm so, dass unterschiedlich große verschiedenfarbige Nikolaushäuser gezeichnet werden können.
2. Die kleinsten Katheten der dargestellten rechtwinkligen Dreiecke sollen zusammen die dargestellte Spirale bilden. Der spitze Winkel sei in allen Dreiecken gleich groß.



   a. Formuliere dazu die Methode void zeichneSpirale(double winkel, double starthypotenuse)
   b. Durch den Aufruf zeichneSpirale(30, 200) soll eine Spirale gezeichnet werden, deren größte Hypotenuse 200 Einheiten beträgt. Die Spirale soll so weit gezeichnet werden, bis die Länge der zu zeichnenden Kathete kleiner als 0.5 ist.
   c. Teste verschiedene Winkel.
   d. Zeichne die Katheten abwechselnd mit zwei Farben.
3. Die Turtle „**Jim**“ spielt gerne Roulette. Verwende als Roulette eine Zufallszahl x. Ist x < 0,45 gewinnt **Jim**. Er färbt seine Welt orange, jubelt (beep) und dreht einen schnellen (speed) Freudeskreis. Ist 0.45 <= x < 0.7 hat Jim den Hauptpreis gewonnen. Er dreht sich 2x superschnell im Kreis und jubelt bei jeder Drehung. Ansonsten verliert Jim, wird traurig und färbt seine Welt schwarz. Jim macht einen langsamen Trauermarsch von 50 Schildkrötenschritten nach rechts. Jim spielt immer 2x.
   Für die Schnelleren: Wenn Jim gewinnt schreibt er mit seiner Spur „WIN“ in seine Welt.

Iteration – Rekursion

Allgemeine Einführung

Zunächst ein paar Beispiele, die in das Thema Rekursion einführen sollen.

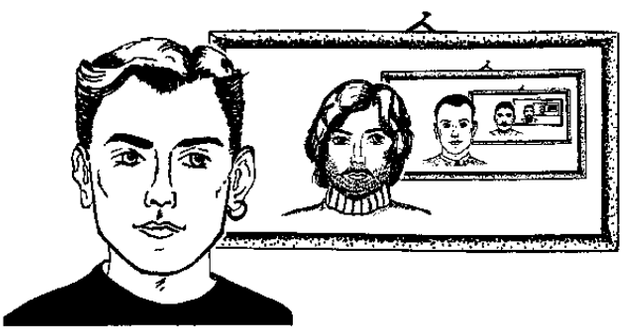
1. Um die Rekursion zu verstehen, muss man erst die Rekursion verstehen.
2. Defintion im Lexikon: Re|kur|sion (lat.), die, siehe: Rekursion
3.  [20]
   

4. n! = (n-1)! · n

All die Beispiele beruhen auf einer einfachen und scheinbar paradoxen Technik, die Rekursion genannt wird. Rekursion kommt vom Lateinischen recurrere, was „zurücklaufen“ bedeutet. Die Einfachheit liegt darin, dass die Ähnlichkeit so groß ist, dass immer das gleiche Teilproblem vorliegt. Das Paradoxe ist, dass

[20] http://unix.rulez.org/~calver/pictures/drawing_3.jpg

ich alle Teilprobleme auf ein einziges, ursprüngliches Teilproblem reduzieren lassen. Das Problem löst sich somit durch sich selbst.



Das Bild zeigt einen Mann neben einem Bild. Auf dem Bild ist ein Mann neben einem Bild zu sehen, auf dem eine Mann neben einem Bild zu sehen ist, …

Beschreibt man dies nun in Form einer Java-Methode, so könnte diese wie folgt aussehen:

```
public void zeichneMannMitBild(double Größe) {
  …
  zeichneMannMitBild(Größe1)
  …
} //end of zeichneMannMitBild
```

Wie man sieht, wird die Methode in der Methode selbst wieder aufgerufen, wobei sich nur die Größe der Darstellung ändert (wie auch immer). Beim erneuten Aufruf der Methode beginnt die Bearbeitung der Methode mit der neuen Größeneingabe von vorne. Dabei wird wieder die Methode zeichneMannMitBild mit einer anderen Größeneingabe aufgerufen usw. Die Folge der ineinander geschachtelten Bilder scheint nicht aufzuhören.

Hier liegt eine Gefahr der Rekursion. Der Computer würde endlos so weiterarbeiten und immer wieder die Methode aufrufen. Wir hätten eine Endlosschleife programmiert und unser Problem wäre nicht gelöst. Um dies zu vermeiden, muss man einen Abbruch der Methodenverarbeitung erzwingen. In diesem Beispiel könnte man berücksichtigen, dass die Darstellung irgendwann so klein wäre, dass man nichts mehr erkennen könnte. Daher müssten wir die Methode im Pseudocode umformulieren:

```
public void zeichneMannMitBild(double Größe) {
  …
  solange Größe noch groß genug
  zeichneMannMitBild(Größe1)
 sonst brich die Methode ab
  …
} //end of zeichneMannMitBild
```

Merke: Wenn in einer Methode die Methode sich selbst aufruft, so spricht man von einem **rekursiven Aufruf** der Methode. Beim rekursiven Aufruf einer Methode entsteht eine Endlosschleife, wenn nicht irgendwann der erneute Aufruf zu einem Abbruch führt. Jede Rekursion benötigt also eine **Abbruchbedingung**.

Veranschaulichung: Rekursion mit einer Turtle

„Schreibe eigenständige Methoden zum Zeichnen folgender Bilder, welche über Ereignismethoden innerhalb eines Java-Programms aufgerufen werden.

Bild 0

Bild 1

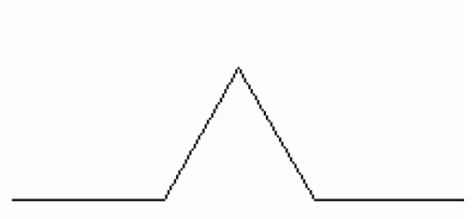


Bild 2

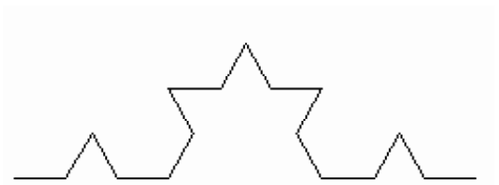
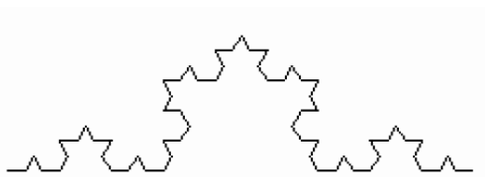


Bild 3



Initiator-Generator

Der **Initiator** der Kochschen Kurve ist eine Strecke:

```
public void initiator(double laenge) {
  myTurtle.draw(laenge);
}
```

Die Kochsche Kurve wird durch den **Generator** von Bild 1 erzeugt. Jede Strecke des Initiators wird durch den Generator ersetzt.

```
public void zeichneKurve1(double laenge) {
  myTurtle.draw(laenge/3);
  myTurtle.turn(60);
  myTurtle.draw(laenge/3);
  myTurtle.turn(-120);
  myTurtle.draw(laenge/3);
  myTurtle.turn(60);
  myTurtle.draw(laenge/3);
  }
```

Bild 2 lässt sich mit der Zeichenmethode von Bild 1 wie folgt zeichnen:

```
public void zeichneKurve2(double laenge) {
  zeichneKurve1(laenge/3);
  myTurtle.turn(60);
  zeichneKurve1(laenge/3);
  myTurtle.turn(-120);
  zeichneKurve1(laenge/3);
  myTurtle.turn(60);
  zeichneKurve1(laenge/3);
  }
```

Bild 3 lässt sich wiederum mit der Zeichenmethode von Bild 2 wie folgt zeichnen:

```
public void zeichneKurve3(double laenge) {
  zeichneKurve2(laenge/3);
  myTurtle.turn(60);
  zeichneKurve2(laenge/3);
  myTurtle.turn(-120);
  zeichneKurve2(laenge/3);
  myTurtle.turn(60);
  zeichneKurve2(laenge/3);
```

```
  }
```

Gesucht ist nun eine Methode, mit welcher man das Bild mit der Nummer n erzeugen kann. Vergleicht man die Methoden miteinander, so fällt auf, dass sie sich nur letztlich nur in der Endung des Namens unterscheiden. Wir machen daher aus der Endung des Namens einen Parameter und erhalten folgende unvollständige Zwischenlösung:

```
public void zeichneKurve(int n, double laenge) {
  zeichneKurve(n-1, laenge/3);
  myTurtle.turn(60);
  zeichneKurve(n-1, laenge/3);
  myTurtle.turn(-120);
  zeichneKurve(n-1, laenge/3);
  myTurtle.turn(60);
  zeichneKurve(n-1, laenge/3);
  }
```

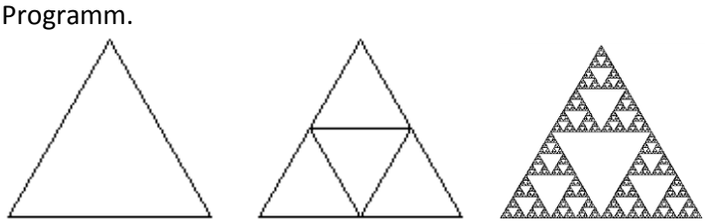
Beginnen wir mit n = 4, so wird n der Reihe nach auf 3, 2, 1, 0, -1, -2, ... reduziert. Wir müssen also dafür sorgen, dass bei n = 0 die Turtle zeichnet. Daraus ergibt sich die Lösung:

```
public void zeichneKurve(int n, double laenge) {
  if ( n > 0 ) {
    zeichneKurve(n-1, laenge/3);    // Rekursionsfall
    myTurtle.turn(60);
    zeichneKurve(n-1, laenge/3);
    myTurtle.turn(-120);
    zeichneKurve(n-1, laenge/3);
    myTurtle.turn(60);
    zeichneKurve(n-1, laenge/3);
    }
  else
     myTurtle.draw(laenge);          // direkter Fall - Terminierung
}
```

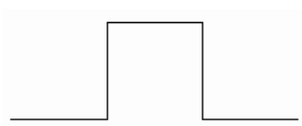
Das Charakteristische dieser Methode ist, dass sie sich in ihrem Anweisungsteil selbst aufruft!“ (Röhner G. , Unterrichtsskript: JAVA-Kursmaterial: Algorithmen und Datenstrukturen, 2005/2006, S. 29f)

Aufgaben:

1. Verwende als Initiator ein gleichseitiges Dreieck. Jede Strecke des Initiators soll durch den Koch-Generator erstetzt werden, so dass folgende Bilder entstehen. Erstelle das entsprechende Programm.



2. Der Initiator ist ein Quadrat, die nebenstehende Figur der Generator.
   Erstelle das entsprechende Programm.



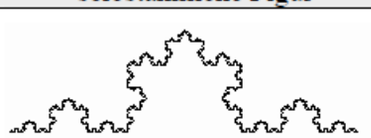
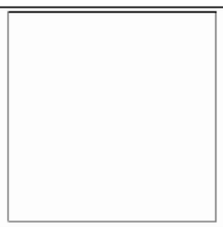
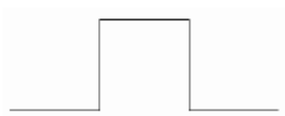
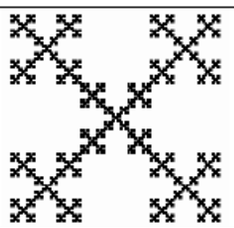
Selbstähnlichkeit

Eine rekursive Zeichenprozedur ruft sich selbst im Anweisungsteil auf. Das muss sich natürlich auf die erzeugte Grafik auswirken. Der Selbstaufruf führt zur Selbstähnlichkeit.

Eine geometrische Figur heißt **selbstähnlich**, wenn sie sich in kongruente Teile zerlegen lässt, die ihr alle ähnlich sind. Vergrößern wir eine der Teilfiguren, so ergibt sich das Ganze.

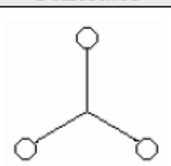
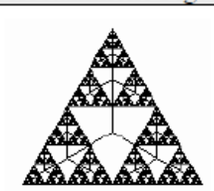
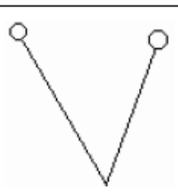
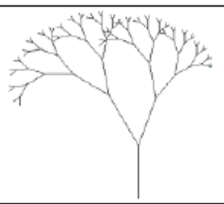
Selbstähnliche Figuren lassen sich mit der ***Initiator-Generator-Methode*** erzeugen. Sie funktioniert wie folgt: In einem Streckenzug I, dem *Initiator*, wird jede Strecke durch eine Figur G, den *Generator*, ersetzt. Daraufhin wird jede Strecke der erzeugten Figur durch die Generatorfigur G ersetzt.

Diese *Streckenersetzung* wird beliebig lange wiederholt.

Initiator	Generator	selbstähnliche Figur
		
		

Eine Variante der ***Initiator-Generator-Methode*** besteht darin, statt der Strecken die Ecken durch den Generator zu ersetzen. Sie funktioniert wie folgt: Zeichne eine Anfangsfigur I und ersetze jede Ecken von I durch eine Figur G. In der so entstandenen Figur ersetze jede Ecke durch G. Diese *Eckenersetzung* wird beliebig lange wiederholt.

Bei der Streckenersetzung werden die Draw-Befehle des Generators durch rekursive Aufrufe ersetzt. Bei der Eckenersetzung werden an allen Ecken des Generators rekursive Aufrufe vorgenommen. Eine Variante hiervon besteht darin, nur einige ausgewählte Ecken zu ersetzen. Die ausgewählten Ecken sind nachfolgend durch Kreise markiert.

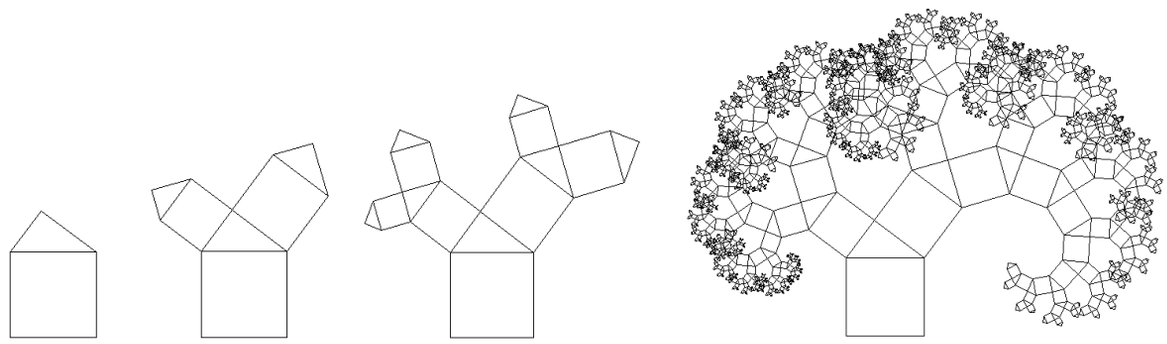
Initiator	Generator	selbstähnliche Figur
(Punkt)		
		

(Röhner G. , Unterrichtsskript: JAVA-Kursmaterial: Algorithmen und Datenstrukturen, 2005/2006, S. 31)

Aufgaben:

3. Erstelle ein Programm, welches den Pythagoras-Baum zeichnet. Du kannst zunächst von festen Winkelgrößen ausgehen.

Hinweis: Die rechtwinkligen Dreiecke haben stets die Winkel $\gamma = 90°$, $\alpha = 53{,}1301°$ und $\beta = 36{,}8699°$ sowie die Seitenverhältnisse $\frac{a}{c} = \frac{4}{5}$ und $\frac{b}{c} = \frac{3}{5}$.



n = 1 n = 2 n = 3 n = 10

Zusatz für die Schnellen: Erweitere dein Programm so, dass der Pythagoras-Baum für beliebige rechtwinklige Dreiecke gezeichnet wird.

Vertiefung des informatischen Konzepts Rekursion

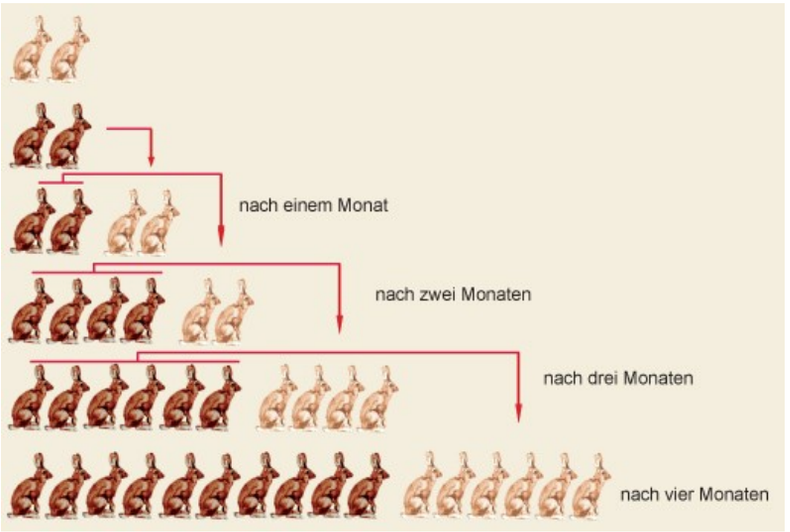
Grundsätzlich ist ein Problem gegeben. Eine Problemlösestrategie lautet: Teile und Herrsche. „Teile und Herrsche“ geht dabei davon aus, dass ein großes Problem zunächst in kleinere Teilprobleme zerlegt wird, so dass man diese lösen kann. Anschließend wird aus den gesamten Teillösungen eine Lösung für das große Problem gebildet.

Für die Problemlösung muss ein Algorithmus gefunden werden. Dieser Algorithmus kann nun aus einzelnen Anweisungen, Schleifen und Verzweigungen bestehen, die schrittweise nacheinander ausgeführt werden. Diese Art von Algorithmen bezeichnet man als **iterativ**. Dem gegenüber stehen die **rekursiven** Algorithmen, die während der Bearbeitung Selbstaufrufe starten, so dass eine verschachtelte Abarbeitung erfolgt. Im Vergleich sind die rekursiven Algorithmen meist eleganter, benötigen jedoch einen wesentlich größeren Speicherbedarf.

Beispiel: Fibonacci-Folgen oder Entwicklung einer Kaninchenpopulation

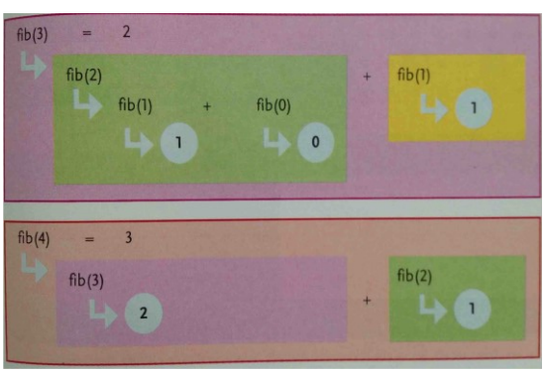
In einem Käfig werden Kaninchen gehalten. Zunächst hat man nur ein Pärchen. Wie viele Paare werden in einem Jahr gezüchtet, wenn pro Monat genau ein weiteres Paar zur Welt kommt und die neuen Kaninchen im 2. Monat nach ihrer Geburt ebenfalls Nachwuchs bekommen.

In der folgenden Darstellung gilt: Die dunkelbraunen Hasen sind geschlechtsreif, sie können also Nachkommen produzieren - die hellbraunen noch nicht.



(Deffke, 2009)

Es ergibt sich somit für die Funktion k(n), wobei k die Anzahl der Paare und n die Anzahl der Monate angibt, dass:
k(0) = 1
k(1) = 1
k(2) = 2
k(3) = 3 …
Allgemein kann man ableiten, dass die Anzahl der Kaninchen im n. Monat der Summe der Kaninchen aus den beiden Vormonaten entspricht. Mathematisch ausgedrückt: k(n) = k(n-1) + k(n-2). „Zahlen, die auf diese Weise gebildet werden, heißen Fibonacci-Zahlen (sprich "Fibonatschi"). Sie sind benannt nach dem italienischen Mathematiker Leonardo von Pisa, genannt Fibonacci. Er hat ungefähr von 1170 bis 1240 gelebt und auf seinen Reisen im Mittelmeerraum das gesamte mathematische Wissen dieser Regionen in einem Buch mit dem Titel "Liber abbaci – Buch der Rechenkunst" zusammengetragen. Unter anderem empfahl er darin die Einführung der arabischen Ziffern, die wir auch heute noch benutzen. Sie sollten die recht unpraktischen römischen Zahlen ersetzen.“ (Deffke, 2009)Es handelt sich also um einen doppelten rekursiven Aufruf. Anschaulich lässt sich die Berechnung der Fibonaccizahlen von 3 und 4 wie folgt darstellen: (Wir ersetzen k(n) durch fib(n))



(Hubwieser, Spohrer, Steinert, & Voß, Informatik 2; Lehrwerk für Gymnasien, 2007, S. 127)

Für die Programmierung der rekursiven Methode benötigen wir nun zunächst eine Abbruchbedingung. Diese ergibt sich aus der Definition für Fibonaccizahlen. Sie sind nur für natürliche Zahlen und null definiert. Für alle Zahlen größer als 1 kann man die Fibonaccizahlen also rekursiv ermitteln.

```
1  public int fib(int n) {
2    if (n <= 1) {           //Abbruchbedingung (Rekursionsanker)
3      if (n==0) {
4        return 0            // fib(0) = 0
5      } // end of if
6      else {
7        return 1            //fib(1) = 1
8      } // end of if-else
9    } // end of if
10   else {
11     return fib(n-1)+fib(n-2);      //Rekursionsschritt
12   } // end of if-else
13 }  //end of fib(n)
```

Aufgaben:

4. Die Berechnung des größten gemeinsamen Teilers (ggT) zweier Zahlen lässt sich auf die Berechnung des ggT der kleineren der beiden Zahlen und ihrer Differenz zurückführen. Dieser Schritt wird so lange wiederholt, bis die Argumente gleich groß sind. Dieser Wert entspricht dem gesuchten ggT:

   Rekursionsvorschrift: $ggT(a,b) = \begin{cases} ggT(a-b,b), falls\ a > b \\ ggT(a,b-a), falls\ a < b \\ a, falls\ a = b\ (Abbruchbedingung) \end{cases}$

   a und b sind natürliche Zahlen!

   c) Erstelle eine GUI, so dass du die Zahlen a und b eingeben kannst.
   d) Überprüfe die Eingabe von a und b, so dass nur die Eingabe nur verarbeitet wird, wenn auch natürliche Zahlen eingegeben worden sind.
   e) Erstelle die Methode public int ggT(int a, int b). Diese Methode soll aufgerufen werden, wenn ein entsprechender Button gedrückt wird.

5. Eine Bakterienkultur vermehrt sich jede Stunde um 20 %. Zu Beginn der Beobachtung besteht die Kultur aus 300 Bakterien.
   a. Gib eine Rekursionsvorschrift für die Bestimmung der Bakterienanzahl nach n Stunden an.
   b. Implementiere die Funktion und berechne, wie viele Bakterien nach 24 Stunden vorhanden sind.

Anhang

Stylekonventionen für die Java-Programmierung

1. **Namen von Klassen, Methoden, Attributen** etc. werden mit eindeutigen, verständlichen Namen versehen.
2. **Klassen** beginnen immer mit einem **Großbuchstaben**.
   public class Beispiel() {…}
3. **Methoden** beginnen mit einem **Kleinbuchstaben**. Kommen mehrere Wörter in einem Methodennamen vor, so wird jeder Anfangsbuchstabe groß geschrieben.
   public void methodeMitMehrerenNamen() {…}
4. **Attribute** verwenden die gleiche Konvention wie Methoden.
   int natuerlicheZahl = 0;
5. Bei der **Klammerung** wird – für eine bestmögliche Übersicht – auf das Öffnen und Schließen auf der gleichen Ebene geachtet. Die Einrücktiefe in einer Ebene sollte ebenfalls einheitlich sein.

```
public class ErsteKlasse
{
    public void zeichne(Graphics g)
    {
        g.drawString(„ErsteKlasse“);
        g.drawLine(70, 50);
    }
}
```

6. Jedes **Programm** wird sauber und ausführlich **dokumentiert**: jedes (wichtige) Attribut, jede Methode und auch jede Klasse (Was bezwecken sie? Warum brauche ich sie? etc.) Jede neue Datei wird mit einer „Titelleiste“ versehen:

```
/**
* Autor: Max Mustermann
* Information zum Programm
```

Literaturverzeichnis

Bundesministerium für Justiz und Verbraucherschutz. (21. 09 2014). Von http://www.gesetze-im-internet.de/gg/art_2.html abgerufen

Deffke, U. (06. 10 2009). *tk-logo.de*. Von http://www.tk-logo.de/cms/beitrag/10002277/204703/ abgerufen

Garmann, D. (11 2011). *www.lernsoftware.filius.de.* Von http://www.lernsoftware-filius.de/downloads/Einfuehrung_Filius.pdf abgerufen

http://www.heise.de. (kein Datum). Von http://www.heise.de/download/java-editor.html abgerufen

Hubwieser, P., Spohrer, M., Steinert, M., & Voß, S. (2007). *Informatik 2; Lehrwerk für Gymnasien.* Stuttgart: Klett.

Hubwieser, P., Spohrer, M., Steinert, M., & Voß, S. (2008). *Informatik 3; Lehrwerk für Gymnasien.* Stuttgart: Klett.

Kraus, M., Reder, M., & Rudel, T. (2012). *klickITsafe 2.0; Sicher und kompetent das Netz erleben; Arbeitsheft mit Onlinetest.* Bodenheim: herdt.

Magenheim, J. u. (2009). *Informatik macchiato; Cartoon-Informatikkurs für Schüler und Studenten.* München: Pearson Studium.

Magenheim, J. (kein Datum). *www.pearson-studium.de.* Von http://www.pearson-studium.de/main/main.asp?page=bookdetails&ProductID=166537 abgerufen

Nerdson, & Neumann, L. (21. 09 2014). *Comic zu "Creative Commons"*. Von http://www.checked4you.de/comic_zu_creative_commons abgerufen

Röhner, G. (2005/2006). Unterrichtsskript: JAVA-Kursmaterial: Algorithmen und Datenstrukturen.

Röhner, G. (2007/2008). Unterrichtsskript: Einführung in das Programmieren.

Röhner, G. (2010/2011). Unterrichtsskript: Grundkurs Informatik E1.

Röhner, G. (13. 09 2012). Informationsmaterial zur MNU-Landestagung.

Röhner, G. (05. 02 2014). *Java Editor*. Von http://www.javaeditor.org/index.php/Java-Editor/de#Integrierte_Entwicklungsumgebungen abgerufen

Skrotzky, P. (04 2010). *http://www.swisseduc.ch.* Von http://www.swisseduc.ch/informatik/internet/gefahren_aus_dem_internet/docs/leitprogramm_gefahren_aus_dem_internet_einfuehrung.pdf abgerufen

unix.rulez.org. (kein Datum). Von http://unix.rulez.org/~calver/pictures/drawing_3.jpg abgerufen

wikipedia.org. (19. 02 2013). Von http://de.wikipedia.org/wiki/Turtle-Grafik abgerufen

York, L., & Wegener, T. (2012). *HTML5; Grundlagen der Erstellung von Webseiten.* Herdt.