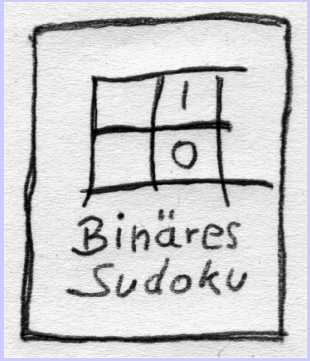
Jörn Fischer
j.fischer@hs-mannheim.de

Willkommen zur Vorlesung

Einführung in die Informatik



Vorstellung

Zu meiner Person...

Inhalt

1 - Organisatorisches

1 - Organisatorisches

Kapitel 1 - Organisatorisches

- Prüfungsvoraussetzung
- Fragen?
- Fragen an Sie...
- Literatur

1 - Organisatorisches

Prüfungsvoraussetzung

- Als Prüfungsleistung bzw. Voraussetzung die Prüfung mitschreiben zu dürfen, werden Online Tests bereitgestellt, von denen mindestens 80% richtig sein müssen!
- Falls Sie bereits bei anderen Dozenten solche Prüfungsleistungen bestanden haben, zählt das natürlich auch!

1 - Organisatorisches

Fragen?

- ...fragt mich einfach...
- ...z.B. in den Vorlesungs-, Frage- oder Übungsstunden...
- …ansonsten auch per E-Mail
- Rückkopplung ist ausdrücklich erwünscht!

1 - Organisatorisches

Fragen zu organisatorischen Dingen?

1 - Organisatorisches

Frage an Sie...

Wer von Ihnen

- hat einen eigenen Computer?
- hat ein Laptop?
- Hat schon programmiert?
- Hat schon mit dem Raspberry Pi oder Arduino gearbeitet?

1 - Organisatorisches

Literatur

Streng genommen brauchen sie für dieses Fach keine eigenen Bücher.

Wenn sie dennoch Interesse haben, finden Sie sicher Bücher in der Bibliothek.

Einen Überblick bietet z.B. das Buch:

Uwe Schneider, Taschenbuch der Informatik im Hanser Verlag (ISBN 978-3-446-42638-2)

2 - Einführung

2 - Einführung

Kapitel 2 - Überblick

- Was ist Informatik?
- Woraus besteht ein Computer?
- Geschichte der Informatik

2 - Einführung

Was ist eigentlich Informatik?

- Informatik ist ein Kunstwort, welches sich aus den Wörtern **Information** und **Mathematik** zusammensetzt.
- Der Ausdruck wurde 1957 von Karl Steinbuch geprägt
- Im Englischen benutzt man den Begriff „**Computer Science**“, für bestimmte Teilgebiete aber auch “**Informatics**“, so wie bei „**Medical Informatics**” oder „**Bioinformatics**“

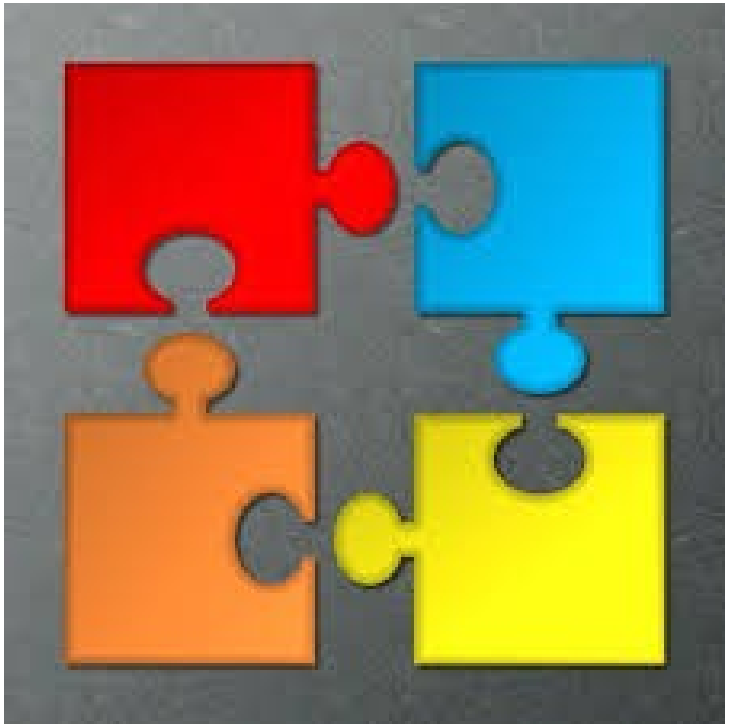
- Definition: Wissenschaft von der systematischen Darstellung, Speicherung, Verarbeitung und Übertragung von Informationen, besonders der automatischen Verarbeitung mithilfe von Digitalrechnern [1]

[1] Duden Informatik A - Z: Fachlexikon für Studium, Ausbildung und Beruf, 4. Aufl., Mannheim 2006. ISBN 978-3411052349

2 - Einführung

Teilgebiete der Informatik

- Theoretische Informatik
- Technische Informatik
- Praktische Informatik
- Angewandte Informatik

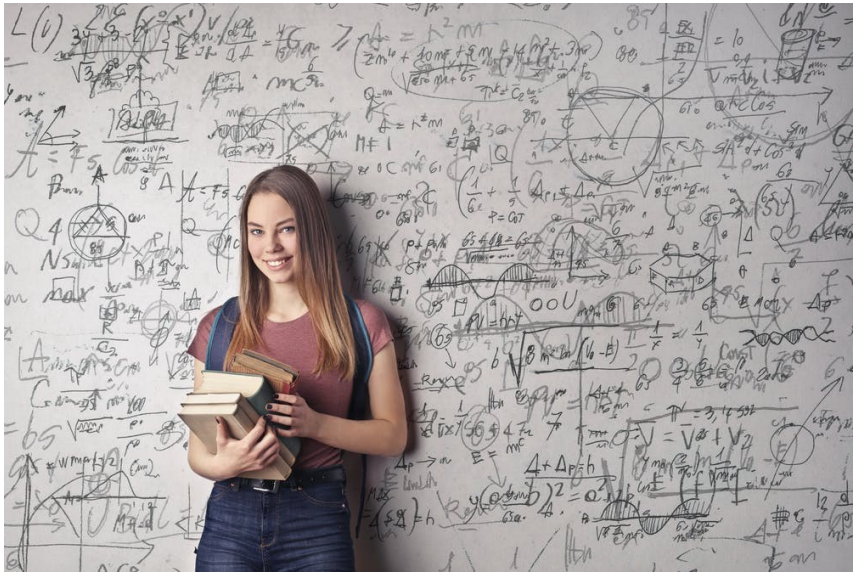


2 - Einführung

Theoretische Informatik

Die theoretische Informatik beschäftigt sich mit den Grundlagenfragen und liefert das mathematische Fundament der Informatik.

- Automatentheorie
- Formale Sprachen
- Theorie der Berechenbarkeit
- Komplexitätstheorie
- Formale Semantik
- Verifikation

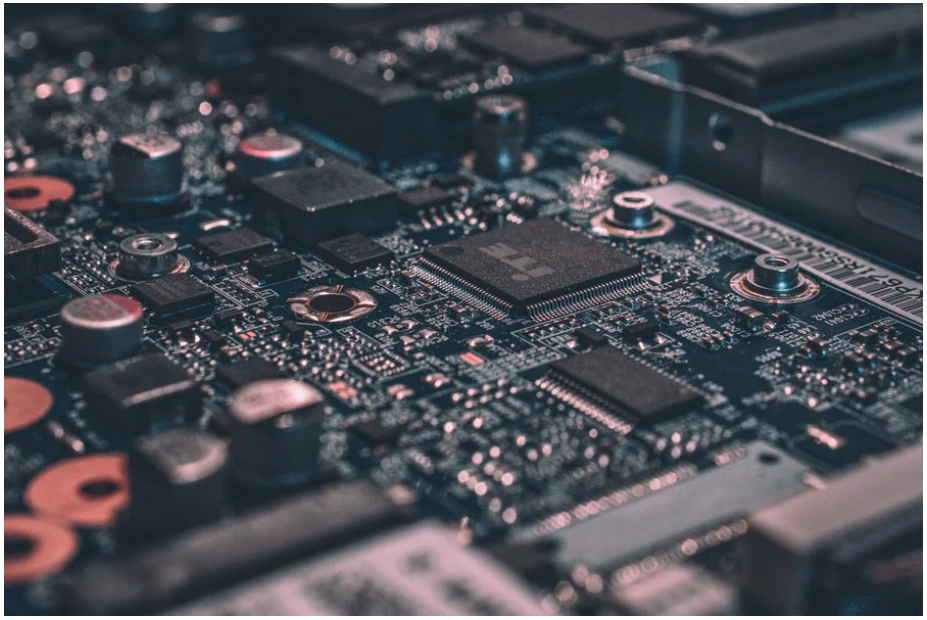


2 - Einführung

Technische Informatik

Die technische Informatik behandelt die Hardware oder Hardwarenahen Konzepte, also wie Computer aufgebaut und vernetzt sind

- Hardware Komponenten
- Schaltnetze, Schaltwerke, Prozessoren
- Mikroprogrammierung
- Rechnerorganisation
- Rechnerarchitektur
- Rechnernetze

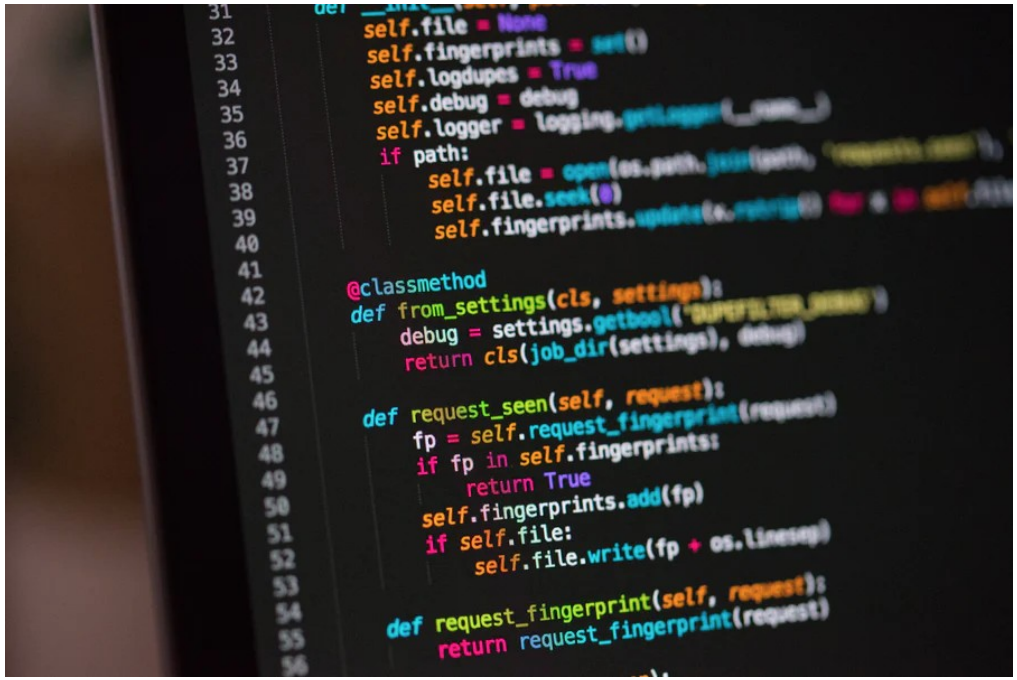


2 - Einführung

Praktische Informatik

Zentrales Thema der praktischen Informatik ist die Programmierung und den damit zusammenhängenden Algorithmen und Datenstrukturen

- Algorithmen und Daten-strukturen
- Programmiersprachen
- Programmiermethoden
- Betriebssysteme
- Software Engineering
- Verteilte Systeme
- Mensch Maschine Kommunikation

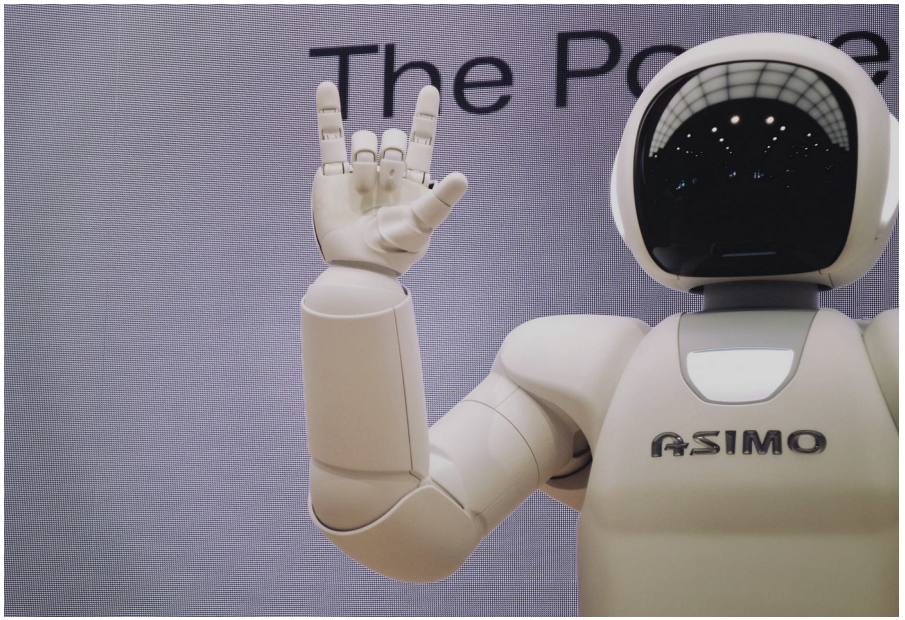


2 - Einführung

Angewandte Informatik

Die Angewandte Informatik untersucht Anwendungsmöglichkeiten des Computers.

- Computergrafik
- Datenbanken
- Künstliche Intelligenz
- Digitale Signalverarbeitung
- Simulation und Modellierung
- Büroautomation
- Anwendungsprogrammierung



2 - Einführung

Woraus besteht ein Computer?

- Hardware

2 - Einführung

Woraus besteht ein Computer?

- Software (Betriebssystem)

2 - Einführung

Woraus besteht ein Computer?

- Software (Programme)

Empfehlenswert: https://www.youtube.com/watch?v=QdVFvsCWXrA&feature=youtu.be

2 - Einführung

Woraus besteht ein Computer?

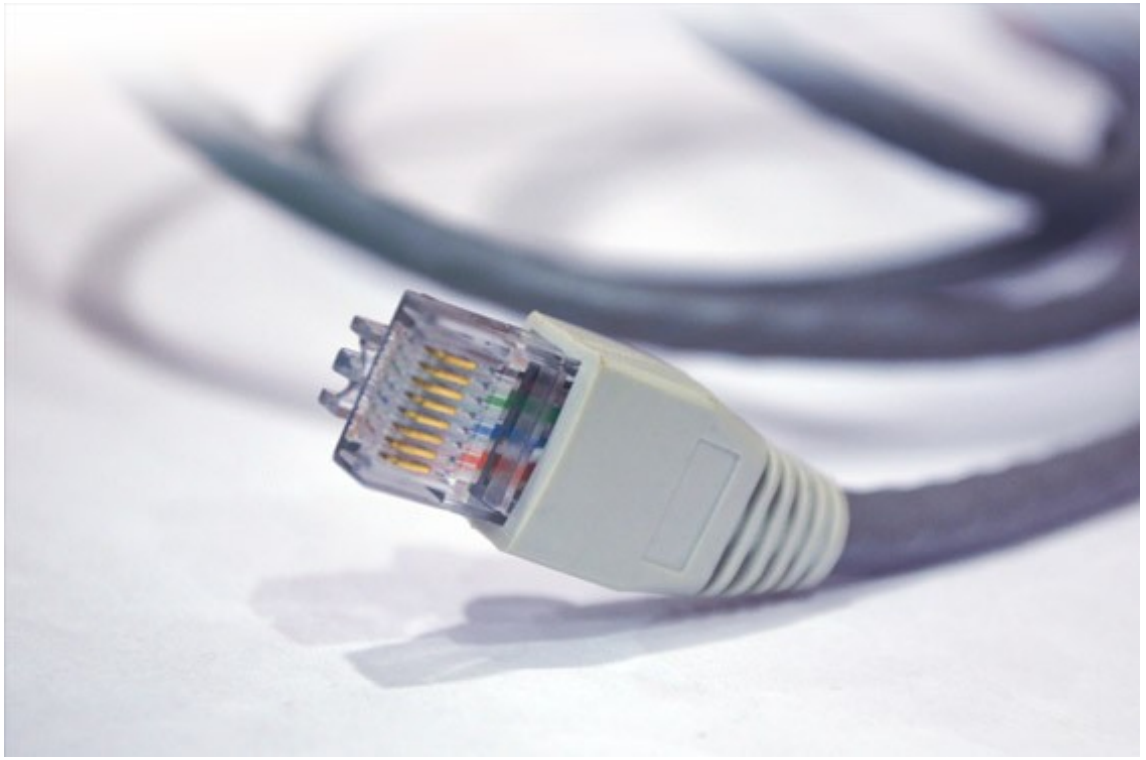
- Software (Daten)

2 - Einführung

Woraus besteht ein Computer?

Kommunikation

- Internet
- Intranet
- LAN
- Ethernet
- WLAN
- WiFi
- Bluetooth
- Mobilfunk
- LTE, 4G, 5G



2 - Einführung

Geschichte der Informatik

2300-2700 v. Chr. Der Abakus ist eines der ältesten bekannten Rechenhilfsmittel (vermutlich sumerischen Ursprungs)

2 - Einführung

Geschichte der Informatik

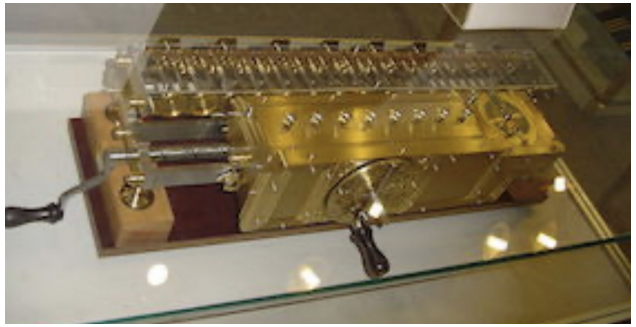
1624 Wilhelm Schickard konstruiert eine rein mechanische Maschine, die die vier Grundrechenarten beherrschte

2 - Einführung

Geschichte der Informatik

1641 Blaise Pascal konstruiert eine Maschine zur Konstruktion sechsstelliger Zahlen

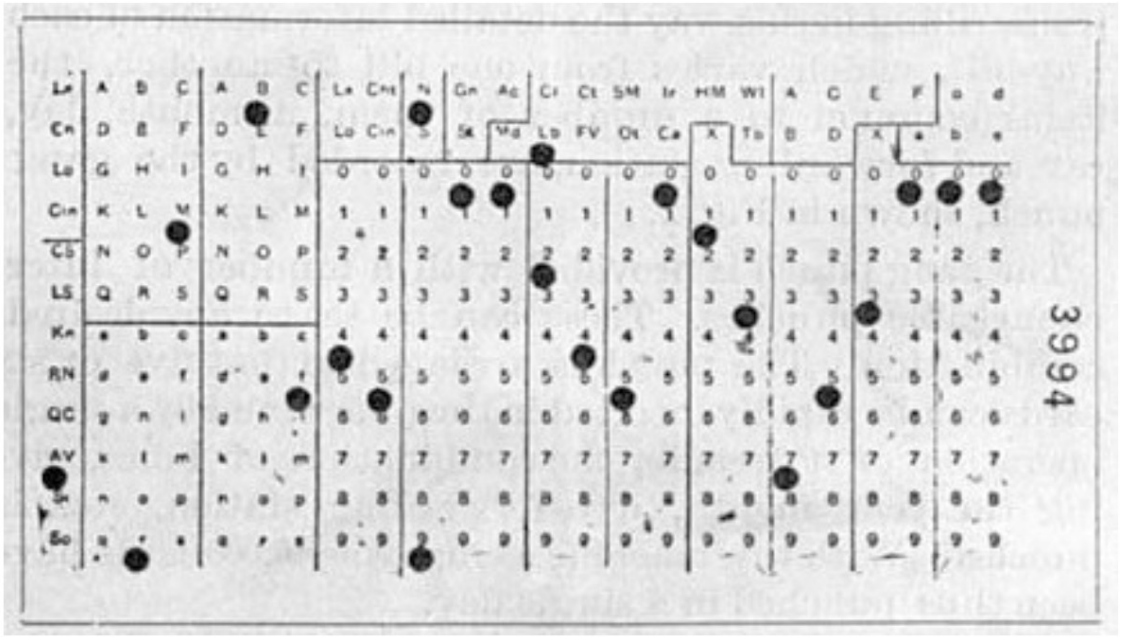
1674 Leibniz konstruiert eine Rechenmaschine mit Staffelwalzen für die vier Grundrechenarten und befasst sich mit dem dualen Zahlensystem



2 - Einführung

Geschichte der Informatik

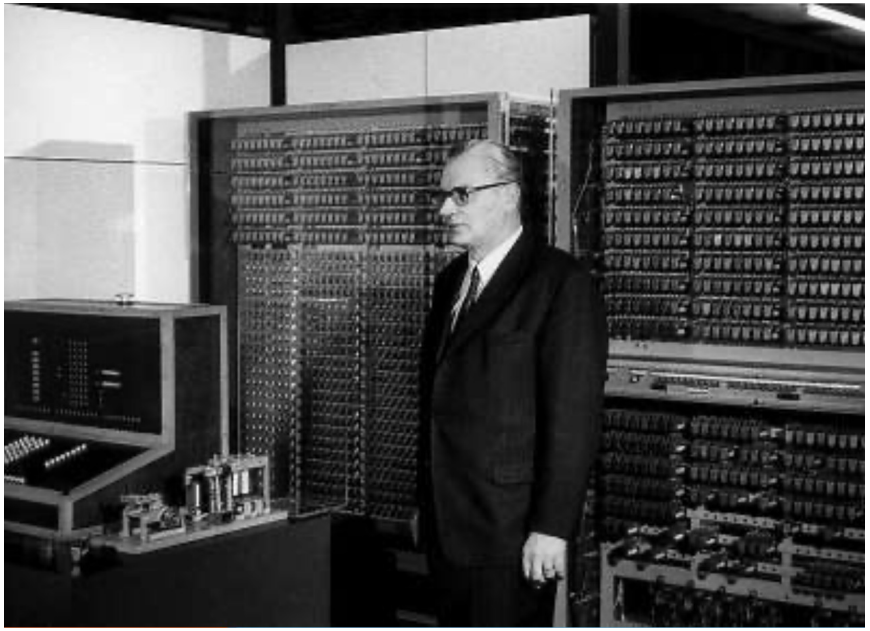
- 1886 Hermann Hollerith erfindet die Lochkarte

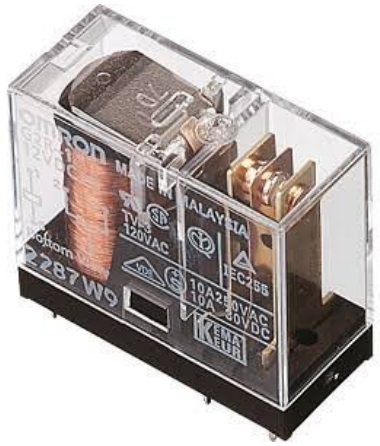


2 - Einführung

Geschichte der Informatik

- 1941 Konrad Zuse entwickelt den ersten programmgesteuerten, elektrischen Rechenautomaten, den Z3. Er ist mit Relaistechnologie aufgebaut und verwendet das duale Zahlensystem

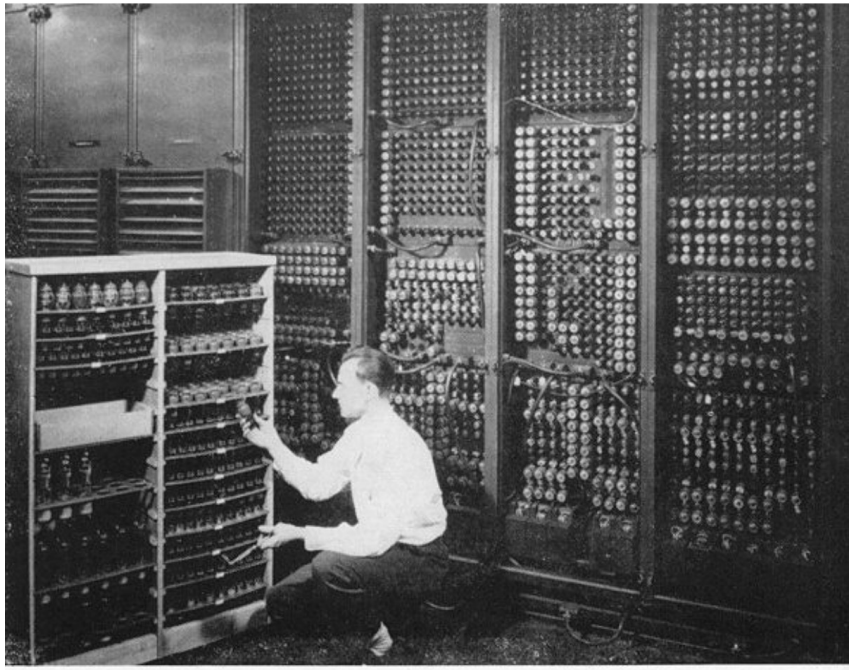




2 - Einführung

Geschichte der Informatik

- 1946 J.P. Eckert und J.W. Mauchly erbauen Eniac, den ersten voll elektronischen Rechner (19.000 Elektronenröhren). Multiplikationszeit: 3 ms.
- 1946 John von Neumann schlägt das Konzept des gespeicherten Programms vor



Replacing a bad tube meant checking among ENIAC's 19,000 possibilities.

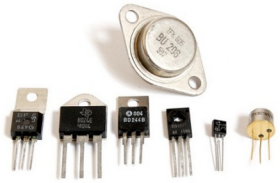
Elektronenröhre

2 - Einführung

Geschichte der Informatik

1955 TRADIC war der erste voll transistorisierte Computer

Transistoren



Elektronenröhre

2 - Einführung

Geschichte der Informatik

Ca. 1960 Erster„Minicomputer“ in Schrankgröße PDP-1 von DEC

Microchip: Intel 8080

2 - Einführung

Geschichte der Informatik

1981 IBM 5150 Personal Computer

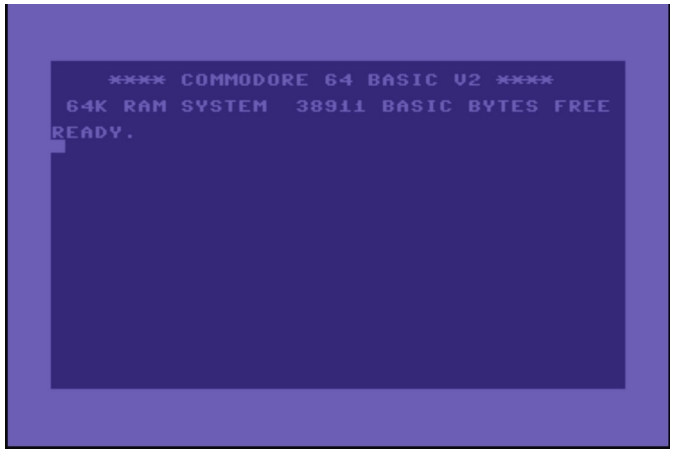
IBM-5150 von 1981

5 ¼ Zoll-Floppy Disk,
160kByte Speicher

2 - Einführung

Geschichte der Informatik

1982 Commodore (C64) Home-/ Spielecomputer



1982 – 1994
Prozessor
MOS Technology 6510
@ 1,023 MHz (NTSC-Version)
@ 0,985 MHz (PAL-Version)
Arbeitsspeicher: 64 KB
Datenträger 170-KB-Disketten, Tonkassetten (Datasette), Steckmodule
Betriebssystem: Commodore Basic V2

Emulator:
https://virtualconsoles.com/online-emulators/c64/

2 - Einführung

Geschichte der Informatik

1983 Apple IIe

Emulator:
https://www.scullinsteel.com/apple2/#karateka

2 - Einführung

Geschichte der Informatik

Emulator:
https://jamesfriend.com.au/pce-js/

2 - Einführung

Geschichte der Informatik

1 MB Speicherkapazität für runde 190 000 Mark

Für eine funktionsfähige Einstiegskonfiguration muss der Anwender bei dem Modell [...] 3090- 180 (32 MB, 16 Kanäle) etwa 6,1 Millionen Mark bezahlen [...]

Quelle: Computerwoche vom 21.02.1986 , https://www.computerwoche.de/a/3090-ibm-macht-druck-ueber-modelle-und-preise,1163347

3 - Hardware

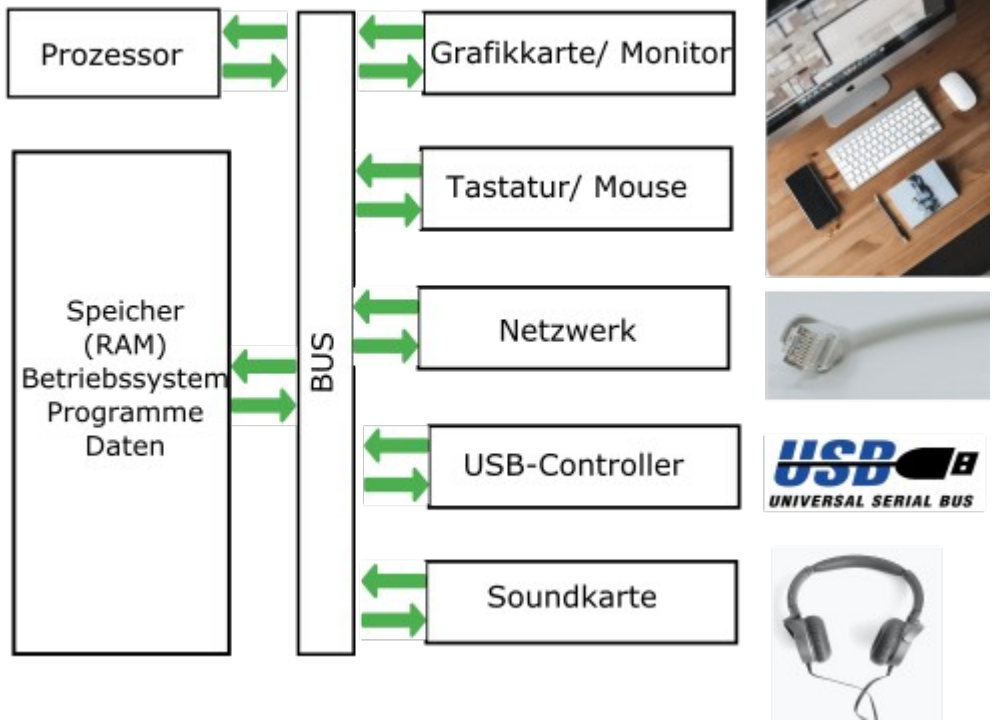
3 - Hardware

Kapitel 3 - Hardware

- Komponenten eines PC-Systems
- Busse
- Mainboard
- Prozessor
- Speicher (RAM, Flash, Harddisk, CD, DVD)

3 - Hardware

Komponenten eines PC-Systems

3 - Hardware

Bussysteme

- Steuerbus
- Adressbus
- Datenbus

Serieller Bus

1 0 0 0 1 0 1 0 0

langsam
Auch für längere Strecken
extern (z. B. USB)

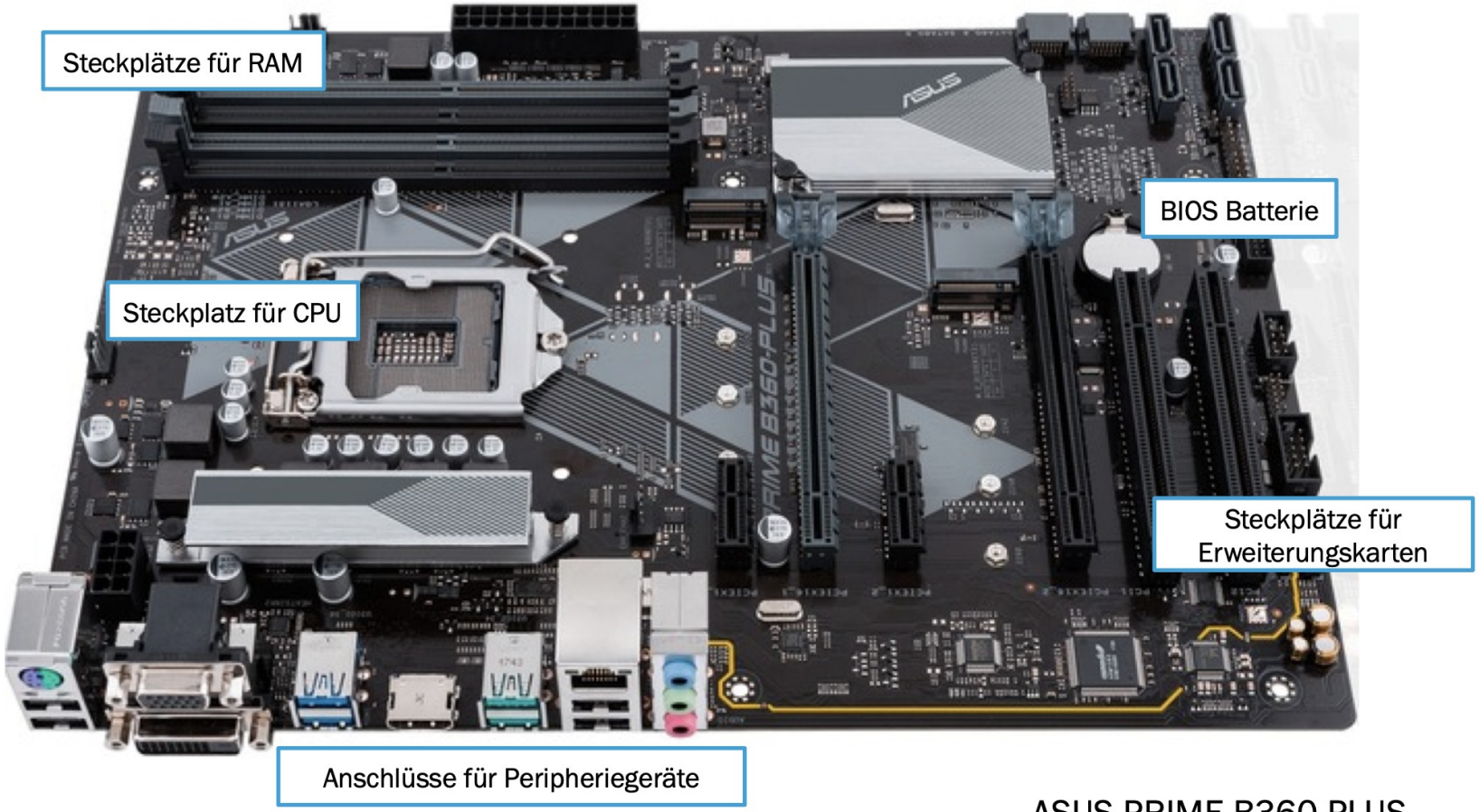
Paralleler Bus

1 0 0 1 0 1 0 0

schnell
für kurze Distanzen
rechnerintern

3 - Hardware

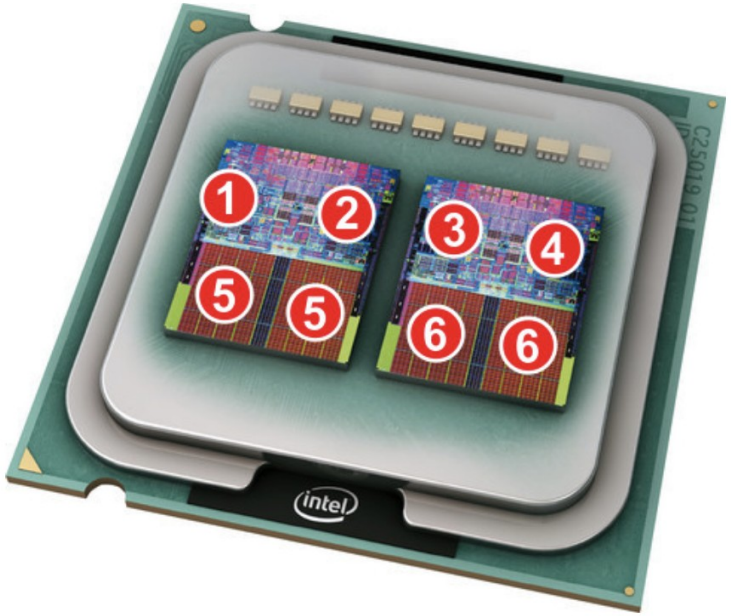
Motherboard



ASUS PRIME B360-PLUS

3 - Hardware

Der Prozessor



Aufbau eines Prozessors mit vier Kernen
am Beispiel der Baureihe „Core 2 Quad“ von Intel

1. Erster Prozessorkern (Core 1)
2. Zweiter Prozessorkern (Core 2)
3. Dritter Prozessorkern (Core 3)
4. Vierter Prozessorkern (Core 4)

5. Erster erweiterter Zwischenspeicher (Advanced Smart Cache)
6. Zweiter erweiterter Zwischenspeicher (Advanced Smart Cache)

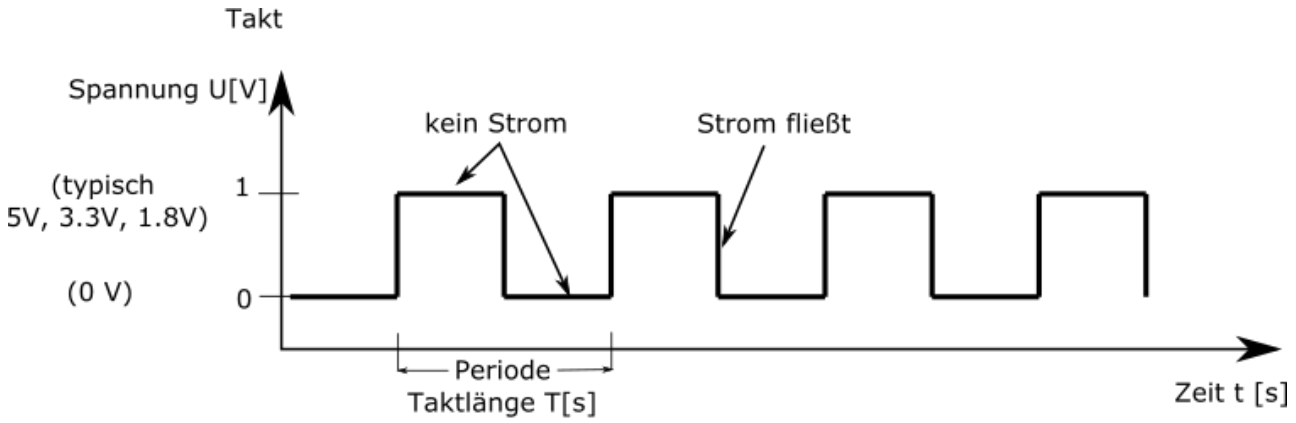
Bildquelle: https://www.computerbild.de/artikel/cbs-Ratgeber-Kurse-PC-Neun-wichtige-Fragen-zu-Prozessoren-1863303.html

3 - Hardware

Prozessor: Leistungsmerkmale

8 Kerne
3.6 GHz Taktfrequenz
16 MB SmartCache

16 Kerne
3.5 GHz Taktfrequenz
32 MB L3-Cache



Taktfrequenz f = Anzahl der Vorgänge/Zeit
f = 1/T bzw. T=1/f

f[Hz] 1Hz = 1/s

Merke: je höher die Taktfrequenz, desto mehr Strom fließt und desto heißer wird der Prozessor!

- Guter Lüfter erforderlich
- Niedrige Spannung
- Stromsparmodus: heruntertakten des Prozessors

3 - Hardware

Prozessor: physikalische Grenzen

- Prozessor mit Taktrate von 3 Ghz
- ...entspricht einer Periode von 0.333 ns
- in dieser Zeit kann ein Signal maximal

$$0.333\ \ 10^{(-9)}\,s\ \ 0.3\ \ 10^{9}\,\frac{m}{s}\ \approx\ 10\,cm$$

zurücklegen

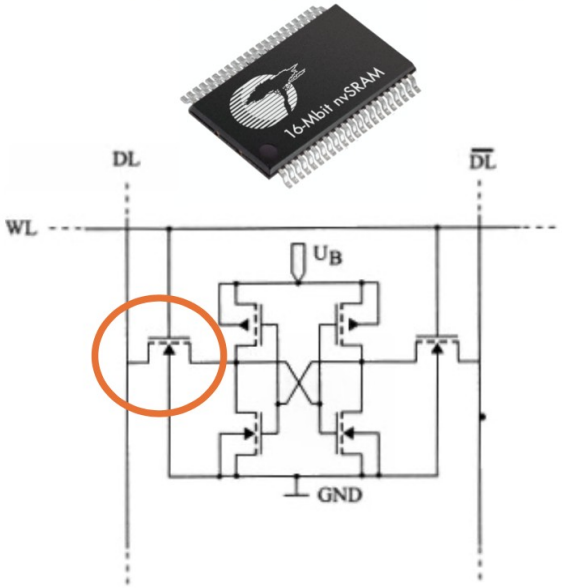
- Bei Verzehnfachung ist die Größenordnung des Chipdurchmesser erreicht!
- höhere Taktraten führen schnell zu Laufzeitproblemen der Signale, darum wird parallel auf mehreren Prozessorkernen gerechnet

3 - Hardware

Random Access Memory (RAM)

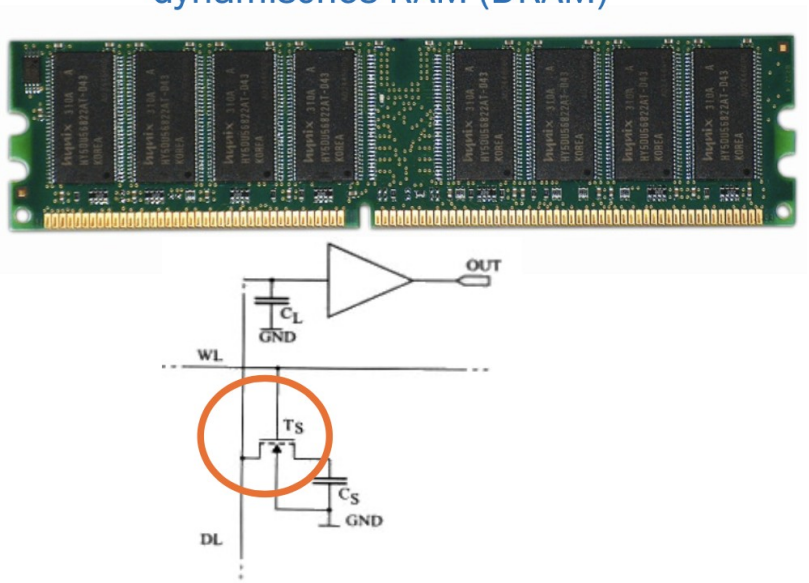
- Flüchtiger Speicher

statisches RAM (SRAM)



schneller Speicher
mit vergleichsweise kleiner Datenkapazität (MB)
in Prozessoren als Cache

dynamisches RAM (DRAM)



große Speichermengen (GB)
mittlerer Datenrate (100–1600 Mbit/s)
Als Arbeitsspeicher

3 - Hardware

Dynamisches Ram

Dynamic RAM (DRAM)
- DRAM speichert die Daten in einer Speicherzelle
- Auffrischung der Speicherzelle alle paar Millisekunden

Synchronous DRAM (SDRAM)
- Selbes Prinzip wie DRAM, aber
- Synchronisierter Datentransfer zwischen Speicher und CPU
- SDRAM ermöglicht der CPU Daten zu verarbeiten, während ein weiterer Prozess in der Warteschlange ist

Double Data Rate SDRAM (DDR SDRAM)
- DDR SDRAM neue Form von SDRAM mit erhöhtem Speichertakt bis zu 200 MHz oder mehr (theoretischer Wert).

DDR2, DDR3, DDR4....
- Weiterentwicklung des DDR-SDRAM um noch schnelleren Zugriff zu ermöglichen

3 - Hardware

Festplattenlaufwerk (HDD)

- Plattenstapel: 2-6 Platten
- Rotationsgeschwindigkeit: 4800-15000 U/min (rpm)
- Positionierzeit (Seek Time): ca. 8ms, Spurwechsel: ca. 2ms Datenzugriffszeit: ca. 12 ms

Quelle: https://de.wikipedia.org/wiki/Festplattenlaufwerk

3 - Hardware

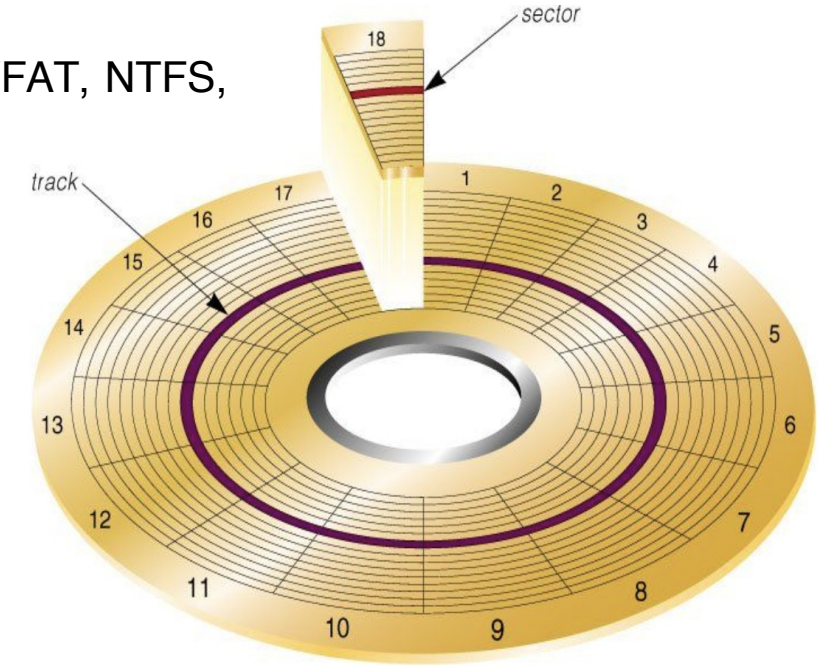
Festplattenlaufwerk (HDD)

- MBR (Master Boot Record) dort steht das Startprogramm (Bootloader) und die Partitionstabelle

- Dateisysteme sorgen dafür, dass die Dateien und Ordner auf der Festplatte, einem Flashspeicher oder einer CD/ DVD gefunden werden, die Zugriffsrechte gespeichert sind und keine Daten verloren gehen. Beispiele:

  - Windows: FAT, FAT16, FAT32, VFAT, NTFS,
  - macOS: HFS und HFS+,
  - Linux: Ext2, Ext3, Ext4,
  - CDs, DVDs, Blu-rays: ISO 9660

- Sektorgröße: 512 oder 1024 Byte (Low-Level-Formatierung)
  Ist kleinste adressierbare Speichereinheit einer Festplatte



3 - Hardware

Solid State Disk (SSD)

- Flash-Technologie
- Selbes Prinzip wie bei USB-Sticks
- Halbleiterbauelemente
- Keine beweglichen Teile, daher robust und schnell

Bildquelle: https://de.wikipedia.org/wiki/Solid-State-Drive

3 - Hardware

Solid State Disk (SSD) vs. Festplatte (HDD)

Vorteile HDD:
- Höhere maximale Speicherkapazität
- Günstiger (Anschaffungspreis)

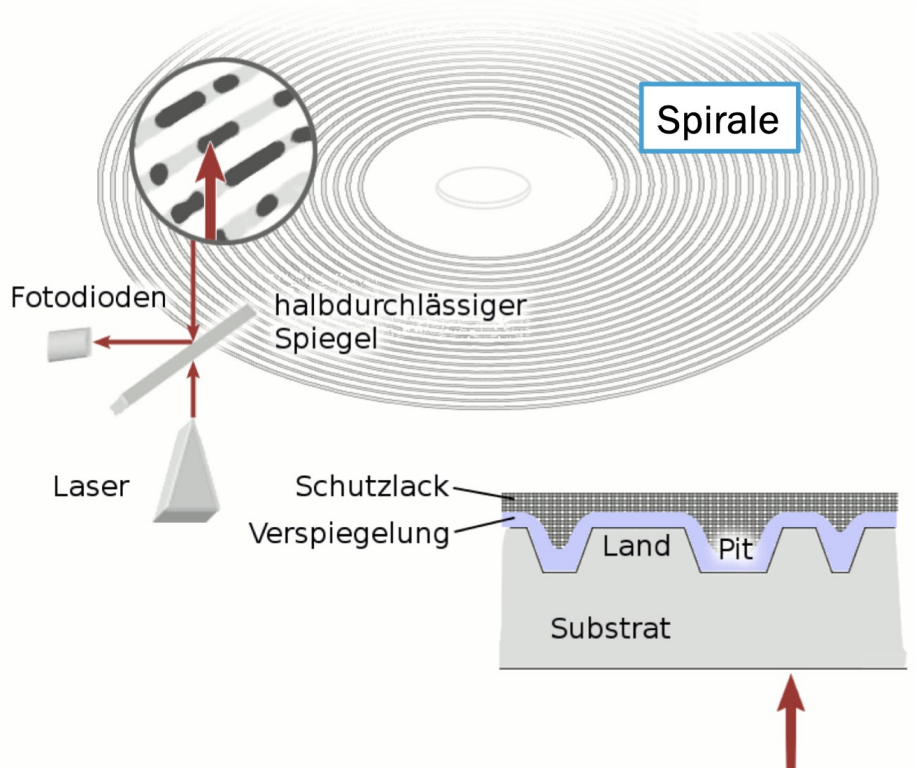
Vorteile SSD:
- hohe Transferraten
- kurze Zugriffszeiten, vor allem beim Lesen
- niedrige Leistungsaufnahme
- geräuschloser Betrieb
- Robustheit

Bildquelle: https://unsplash.com/s/photos/harddisk
https://commons.wikimedia.org/wiki/File:Samsung_MZ-N5E500_and_Samsung_MZ-M5E250_20170314.jpg

3 - Hardware

Optische Speichermedien

- Können mit Laserlicht ausgelesen und beschrieben werden



Speicherkapazität
- CD (650 MByte)
- DVD (4,7 GByte)
- Blue-ray Disk (25 GByte)

3 - Hardware

Optische Speichermedien

- Der Übergang zwischen Pit und Land entspricht einer 1
- kein Übergang entspricht einer 0
- EFM (Eight-to-Fourteen-Modulation)
- Zwischen jeder 1 sind mindestens zwei und maximal zehn Nullen

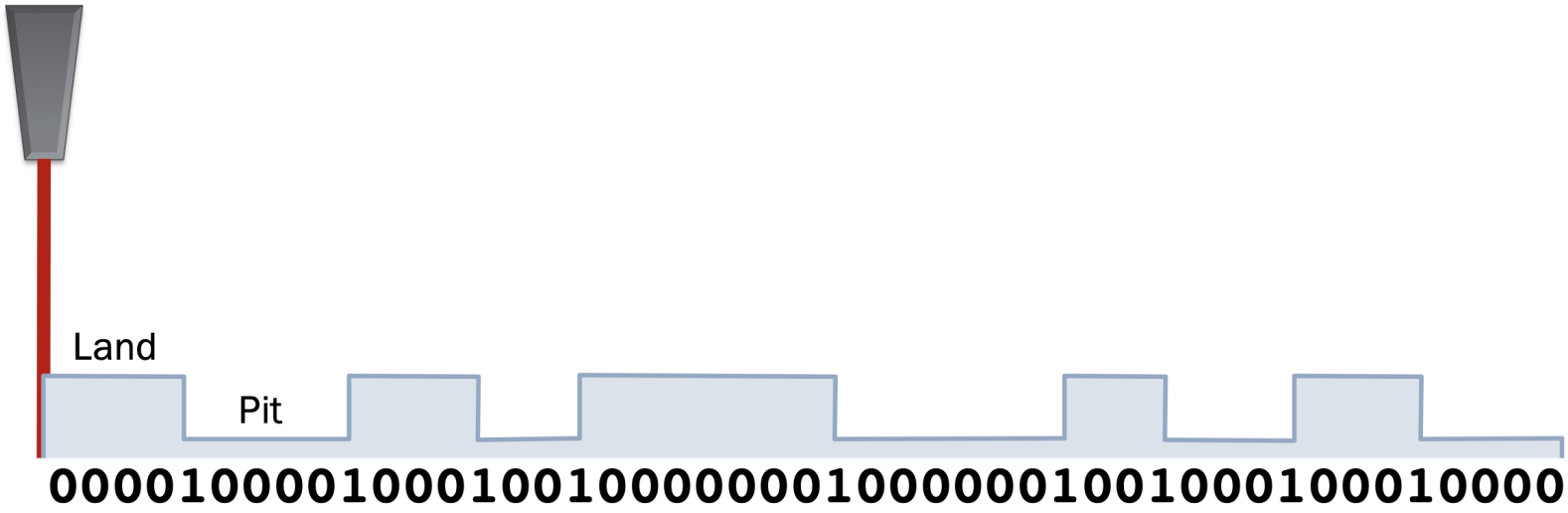


Bild: http://referate.mezdata.de/sj2003/cd_thomas-ley/index.php

4 - Zahlensysteme und Kodierungen

4 - Zahlensysteme und Kodierung

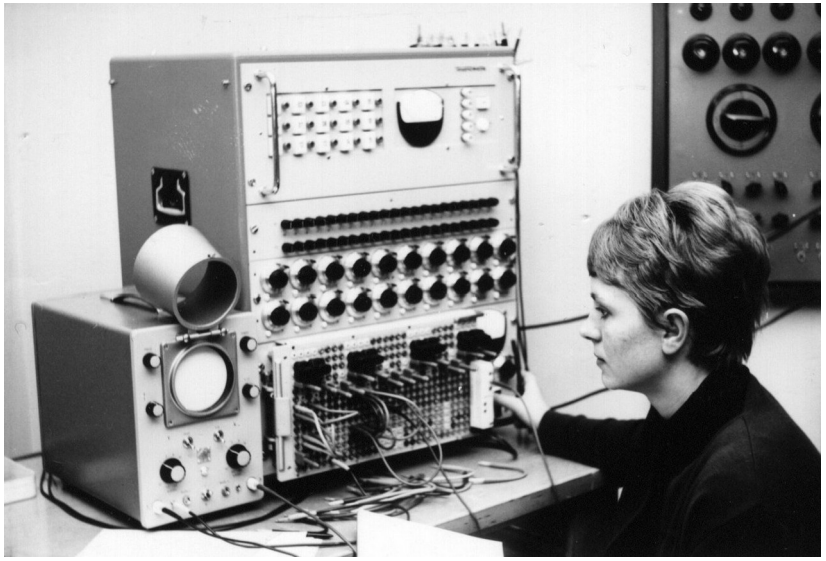
Kapitel 4 – Zahlensysteme und Kodierung

- Warum 0 und 1?
- Bit
- Binärsystem
- Umwandlung Binär - Dezimal – Hexadezimal
- Bytereihenfolge
- Datentypen
- Dateien/Dateiformate
- Komprimierung

4 - Zahlensysteme und Kodierung

Warum 0 und 1?

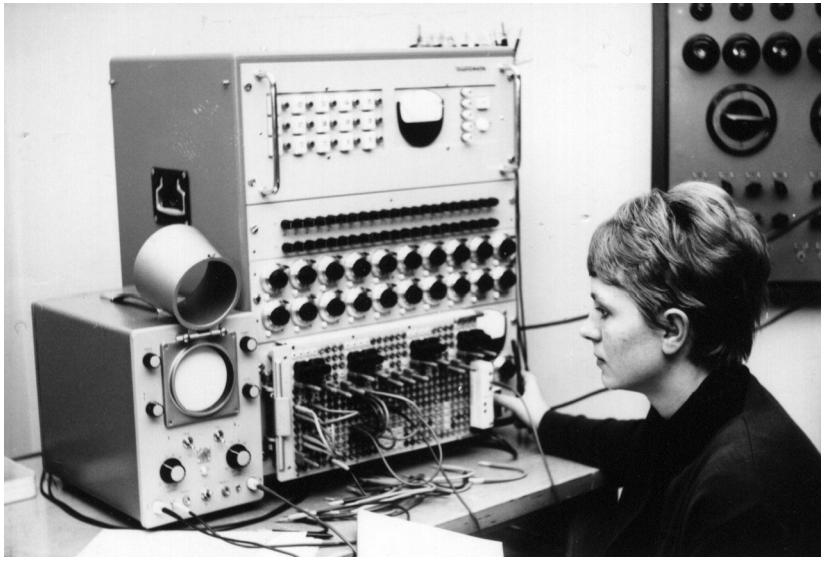
- Es gab früher auch Analogrechner, mit denen man z.B. Differentialgleichungen lösen konnte.
- Prinzipiell könnte jede Zahl mit einer bestimmten Spannung dargestellt werden.
- Was aber, wenn die Zahl winzig klein ist und man viele Nachkommastellen zur Berechnung braucht...



4 - Zahlensysteme und Kodierung

Warum 0 und 1?

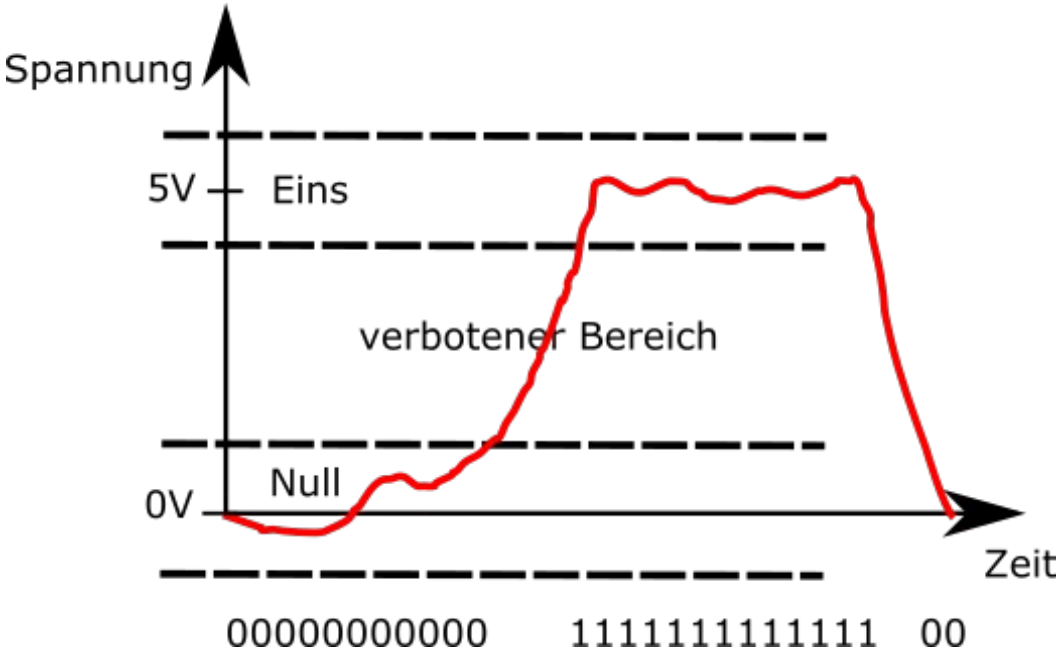
- ...oder die Zahl über den darstellbaren Bereich hinaus geht?
- zudem hat man bei einer Spannung eine Vielzahl von Störfaktoren, die das Ergebnis verfälschen können.



4 - Zahlensysteme und Kodierung

Warum 0 und 1?

- Was, wenn man die Eins als 5V darstellt und die Null als 0V?
- Dann kann man sogar einen Tolleranzbereich darum definieren, sodass 4.9V auch eine Eins bedeutet und 0.1V eine Null.



4 - Zahlensysteme und Kodierung

Bit

- Ein Bit ist in der Informatik die kleinste Informationseinheit
- Ein Bit kann nur zwei Zustände annehmen nämlich 0 und 1
- Bit = Binary Digit

Interpretation:

0	1
Nein	Ja
Falsch	Wahr
Offen	Geschlossen
Low	High

4 - Zahlensysteme und Kodierung

Byte

- Rechner arbeiten immer mit Blöcken von Bits. Die kleinste gebräuchliche Einheit sind 8 Bits.

1 Byte = 8 Bit

Ein Byte hat den Wertebereich:

Binär	00000000 – 11111111
Dezimal	0 – 255

4 - Zahlensysteme und Kodierung

Was gibt es für Datentypen?

• Logische Werte:	boolean (true = 1, false = 0)	✓
• Positive ganze Zahlen:	unsigned integer (0, 1, 2, 3, 4…)	???
• Ganze Zahlen:	integer (...-3, -2, -1, 0, 1, 2, 3…)	
• Gleitkommazahlen:	floatingpoint (+/- $x \bullet 2^{exp}$)	
• Einzelne Zeichen	character („a“, „b“, „c“,...„#“,...„*“„&“…)	
• Zeichenketten	Strings („Hello World“...)	

4 - Zahlensysteme und Kodierung

Binärsystem

- Aber wie kann man jetzt größere Zahlen darstellen?

$4 = 2^2$	$2 = 2^1$	$1 = 2^0$	Zahl
0	0	0	0•4+0•2+0•1 = 0
0	0	1	0•4+0•2+1•1 = 1
0	1	0	0•4+1•2+0•1 = 2
0	1	1	0•4+1•2+1•1 = 3
1	0	0	1•4+0•2+0•1 = 4
1	0	1	1•4+0•2+1•1 = 5
1	1	0	1•4+1•2+0•1 = 6
1	1	1	1•4+1•2+1•1 = 7

- … so kann man mit n Bits die Zahlen von 0 bis 2^{n-1} (also 2^n Zahlen) darstellen!

Beispiel: n=10 → 2^{10} = 1024 Zahlen, also die Zahlen von 0 bis 1023

Umrechner: https://www.arndt-bruenner.de/mathe/scripts/Zahlensysteme.htm

4 - Zahlensysteme und Kodierung

Dezimal in Binär Umwandlung

Umwandlung der Zahl 1234 (dez):

```
1234/2 = 617 Rest 0   (LSB = Least Significant Bit)
617 /2 = 308 Rest 1
308 /2 = 154 Rest 0
154 /2 =  77 Rest 0
77  /2 =  38 Rest 1
38  /2 =  19 Rest 0
19  /2 =   9 Rest 1
9   /2 =   4 Rest 1
4   /2 =   2 Rest 0
2   /2 =   1 Rest 0
1   /2 =   0 Rest 1   (MSB = Most Significant Bit)
```

=> 1234 (dez) = 10011010010 (bin)

4 - Zahlensysteme und Kodierung

Binär in Dezimal Umwandlung

Umwandlung der Zahl 10011010010 (bin):

$0 \cdot 2^0 +$

$1 \cdot 2^1 +$

$0 \cdot 2^2 +$

$0 \cdot 2^3 +$

$1 \cdot 2^4 +$

$0 \cdot 2^5 +$

$1 \cdot 2^6 +$

$1 \cdot 2^7 +$

$0 \cdot 2^8 +$

$0 \cdot 2^9 +$

$1 \cdot 2^{10} = 2 + 16 + 64 + 128 + 1024 = 1234$

=> 10011010010 (bin) = 1234 (dez)

4 - Zahlensysteme und Kodierung

Hexadezimalsystem

- Ziffern: 0 1 2 3 4 5 6 7 8 9 A B C D E F (entspricht 0..15 dezimal)
- So wie das Binärsystem die Basis 2 hat und das Dezimalsystem die Basis 10 hat, so hat das Hexadezimalsystem die Basis 16.

$$10010\ (bin) = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0$$

$$12345\ (dez) = 1 \cdot 10^4 + 2 \cdot 10^3 + 3 \cdot 10^2 + 4 \cdot 10^1 + 5 \cdot 10^0$$

$$1AB57\ (hex) = 1 \cdot 16^4 + 10 \cdot 16^3 + 11 \cdot 16^2 + 5 \cdot 16^1 + 7 \cdot 16^0$$

A B Basis

Da $16 = 2^4$ ist es leicht 4 Bit immer zu einer Hexadezimalziffer zusammen zu fassen:

```
0000 = 0   0100 = 4   1000 = 8   1100 = C
0001 = 1   0101 = 5   1001 = 9   1101 = D      1110 0100 1001 1111 (bin) = E49F (hex)
0010 = 2   0110 = 6   1010 = A   1110 = E
0011 = 3   0111 = 7   1011 = B   1111 = F
```

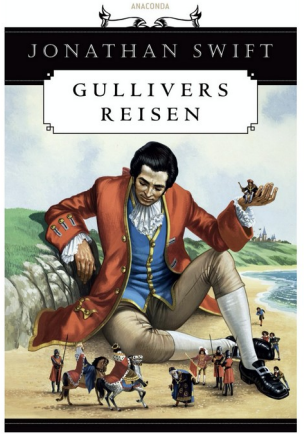
4 - Zahlensysteme und Kodierung

Bytereihenfolge

- Wenn man nun eine Zahl 04030201(hex) in den Speicher schreibt, dann stehen die Bytes 04, 03, 02 und 01 im Speicher, aber in welcher Reihenfolge?
- Das hängt vom Prozessor und vom Betriebssystem ab!

- Man spricht im englischen von Byte order oder endianess
- Man nennt die Reihenfolge
  - Big Endian, wenn die höchstwertigste Ziffer an erster Stelle steht 04 03 02 01
  - Little Endian, wenn die kleinstwertigste Ziffer an erster Stelle steht 01 02 03 04

- Bezeichnung geht zurück auf den Roman Gullivers Reisen (Big Enders, Little Enders)
- Wurde 1980 vom Informatiker Danny Cohen im Bezug auf Bytereihenfolgen verwendet



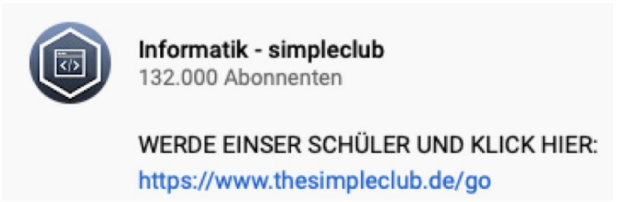
4 - Zahlensysteme und Kodierung

Darstellung im Computer

Hier noch zwei Videos von Simple Club, die ihr euch anschauen solltet:

Simple Club: Ganze Zahlen
https://youtu.be/XG-rVDpm9A4

Simple Club: Das Zweierkomplement
https://youtu.be/utqzSGXd4X4



4 - Zahlensysteme und Kodierung

Datentypen für ganze Zahlen

Typ	Java	C	Wertebereich
short	16 Bit	16 Bit	-32768 bis +32767
unsigned short	-	16 Bit	0 bis 65535
int	32 Bit	16 Bit 32 Bit 64 Bit	-32768 bis +32767 -2^{31} bis $2^{31} - 1$ -2^{63} bis $2^{63} - 1$
unsigned int	-	16 Bit 32 Bit 64 Bit	0 bis 65535 0 bis $2^{32} - 1$ 0 bis bis $2^{64} - 1$
long	64 Bit	32 Bit 64 Bit	-2^{31} bis $2^{31} - 1$ -2^{63} bis $2^{63} - 1$
unsigned long	-	32 Bit 64 Bit	0 bis $2^{32} - 1$ 0 bis bis $2^{64} - 1$

4 - Zahlensysteme und Kodierung

Datentypen

• Logische Werte:	boolean (true = 1, false = 0)	✓
• Positive ganze Zahlen:	unsigned integer (0, 1, 2, 3, 4…)	✓
• Ganze Zahlen:	integer (...-3, -2, -1, 0, 1, 2, 3…)	(negative Werte?)
• Gleitkommazahlen:	floatingpoint (+/- $x \bullet 2^{exp}$)	
• Einzelne Zeichen	character („a“, „b“, „c“,...„#“,...„*“„&“…)	
• Zeichenketten	Strings („Hello World“...)	

4 - Zahlensysteme und Kodierung

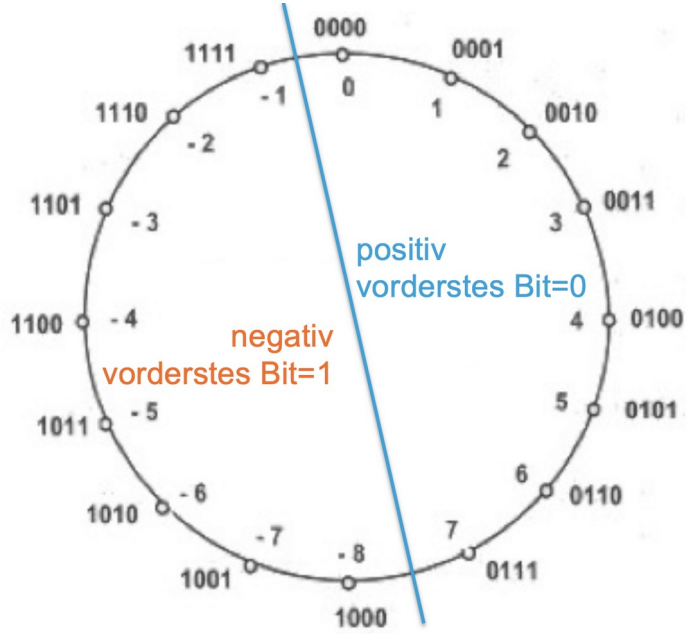
Negative ganze Zahlen

- Negative Zahlen werden im Einer- oder im Zweierkomplement kodiert.
- Beim *Einerkomplement* invertiert man einfach alle Bits. Das führt allerdings dazu, dass aus einer 0 (dez), sprich 00000000 (bin) eine 11111111 (bin) wird, obwohl die Null ja eigentlich mit der minus Null identisch sein sollte. Das muss dann auch bei Berechnungen korrigiert werden.

4 - Zahlensysteme und Kodierung

Negative ganze Zahlen

- Beim *Zweierkomplement* löst man das Problem, indem man alle Bits invertiert und dann eine 1 darauf addiert. So wird aus 00000000 (bin) → 11111111+00000001 (bin) → (1)00000000 (bin), wobei das 9. Bit in der 8 Bitdarstellung rausfällt. So bleibt die Null eine Null.



4 - Zahlensysteme und Kodierung

Wie rechnet der Computer?

- Die binären Rechenregeln sind ähnlich wie die für Dezimalzahlen.
  Beispiel Addition/ Subtraktion:

```
 0 + 0 = 0, carry flag = 0          (carry flag ist das Übertragsbit,
 0 + 1 = 1, carry flag = 0           welches auswirkungen auf die
 1 + 0 = 1, carry flag = 0           nachfolgende Stelle hat)
 1 + 1 = 0, carry flag = 1

  00110110   = 32+16+4+2 = 54(dez)       00110110 = 54(dez)
 +01000101   = 64+4+1    = 69(dez)      +11111110 = -2(dez) (Zweierkomplement)
C_____1____                            C1111111____
  01111011   =            123(dez)    (1)00110100 = 52(dez)
```

4 - Zahlensysteme und Kodierung

Datentypen

• Logische Werte:	boolean (true = 1, false = 0)	✓
• Positive ganze Zahlen:	unsigned integer (0, 1, 2, 3, 4…)	✓
• Ganze Zahlen:	integer (...-3, -2, -1, 0, 1, 2, 3…)	✓
• Gleitkommazahlen:	floatingpoint (+/- $x \cdot 2^{exp}$)	(Wie geht das?)
• Einzelne Zeichen	character („a“, „b“, „c“,...„#“,...„*“„&“…)	
• Zeichenketten	Strings („Hello World“...)	

4 - Zahlensysteme und Kodierung

Gleitkommazahlen

- Wie stellt man Zahlen wie Pi= 3.1415927 im Computer dar?

```
Vor dem Komma: 3(dez) = 11 (bin)
Nach dem Komma: 0,1415927 • 2 = 0,2831854  (vor dem Komma 0)
                0,2831854 • 2 = 0,5663708  (vor dem Komma 0)
                0,5663708 • 2 = 1,1327416  (vor dem Komma 1)
                0,1327416 • 2 = 0,2654832  (vor dem Komma 0)
                …
3,1415927(dez) = 11.0010…(bin) = 1.10010…(bin) • 2^1
                   Mantisse = (1.)10010…, Exponent = 1(dez)
Exponent in Excess-127 Schreibweise: 1+127 → 128(dez) → 10000000(bin)
                  Vorz.  Exp.          Mantisse
Floatingpoint Zahl: 0 10000000 10010010000111111011011
                 1 Bit Vorzeichen
                 8 Bit Exponent (Excess 127 Schreibweise)
                23 Bit Mantisse
```

IEEE754 Format: https://www.h-schmidt.net/FloatConverter/IEEE754de.html

4 - Zahlensysteme und Kodierung

Datentypen

• Logische Werte:	boolean (true = 1, false = 0)	✓
• Positive ganze Zahlen:	unsigned integer (0, 1, 2, 3, 4…)	✓
• Ganze Zahlen:	integer (...-3, -2, -1, 0, 1, 2, 3…)	✓
• Gleitkommazahlen:	floatingpoint (+/- $x \cdot 2^{exp}$)	✓
• Einzelne Zeichen	character („a“, „b“, „c“,...„#“,...„*“„&“…)	???
• Zeichenketten	Strings („Hello World“...)	

4 - Zahlensysteme und Kodierung

ASCII Code

- Wie werden aber nun einzelne Zeichen kodiert?

American Standard Code for Information Interchange

ASCII control characters

DEC	HEX	Simbolo ASCII	
00	00h	NULL	(carácter nulo)
01	01h	SOH	(inicio encabezado)
02	02h	STX	(inicio texto)
03	03h	ETX	(fin de texto)
04	04h	EOT	(fin transmisión)
05	05h	ENQ	(enquiry)
06	06h	ACK	(acknowledgement)
07	07h	BEL	(timbre)
08	08h	BS	(retroceso)
09	09h	HT	(tab horizontal)
10	0Ah	LF	(salto de linea)
11	0Bh	VT	(tab vertical)
12	0Ch	FF	(form feed)
13	0Dh	CR	(retorno de carro)
14	0Eh	SO	(shift Out)
15	0Fh	SI	(shift In)
16	10h	DLE	(data link escape)
17	11h	DC1	(device control 1)
18	12h	DC2	(device control 2)
19	13h	DC3	(device control 3)
20	14h	DC4	(device control 4)
21	15h	NAK	(negative acknowle.)
22	16h	SYN	(synchronous idle)
23	17h	ETB	(end of trans. block)
24	18h	CAN	(cancel)
25	19h	EM	(end of medium)
26	1Ah	SUB	(substitute)
27	1Bh	ESC	(escape)
28	1Ch	FS	(file separator)
29	1Dh	GS	(group separator)
30	1Eh	RS	(record separator)
31	1Fh	US	(unit separator)
127	20h	DEL	(delete)

ASCII printable characters

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
32	20h	espacio	64	40h	@	96	60h	`
33	21h	!	65	41h	A	97	61h	a
34	22h	"	66	42h	B	98	62h	b
35	23h	#	67	43h	C	99	63h	c
36	24h	$	68	44h	D	100	64h	d
37	25h	%	69	45h	E	101	65h	e
38	26h	&	70	46h	F	102	66h	f
39	27h	'	71	47h	G	103	67h	g
40	28h	(	72	48h	H	104	68h	h
41	29h	)	73	49h	I	105	69h	i
42	2Ah	*	74	4Ah	J	106	6Ah	j
43	2Bh	+	75	4Bh	K	107	6Bh	k
44	2Ch	,	76	4Ch	L	108	6Ch	l
45	2Dh	-	77	4Dh	M	109	6Dh	m
46	2Eh	.	78	4Eh	N	110	6Eh	n
47	2Fh	/	79	4Fh	O	111	6Fh	o
48	30h	0	80	50h	P	112	70h	p
49	31h	1	81	51h	Q	113	71h	q
50	32h	2	82	52h	R	114	72h	r
51	33h	3	83	53h	S	115	73h	s
52	34h	4	84	54h	T	116	74h	t
53	35h	5	85	55h	U	117	75h	u
54	36h	6	86	56h	V	118	76h	v
55	37h	7	87	57h	W	119	77h	w
56	38h	8	88	58h	X	120	78h	x
57	39h	9	89	59h	Y	121	79h	y
58	3Ah	:	90	5Ah	Z	122	7Ah	z
59	3Bh	;	91	5Bh	[	123	7Bh	{
60	3Ch	<	92	5Ch	\	124	7Ch	|
61	3Dh	=	93	5Dh	]	125	7Dh	}
62	3Eh	>	94	5Eh	^	126	7Eh	~
63	3Fh	?	95	5Fh	_			

theASCIIcode.com.ar

Extended ASCII characters

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
128	80h	Ç	160	A0h	á	192	C0h	└	224	E0h	Ó
129	81h	ü	161	A1h	í	193	C1h	┴	225	E1h	ß
130	82h	é	162	A2h	ó	194	C2h	┬	226	E2h	Ô
131	83h	â	163	A3h	ú	195	C3h	├	227	E3h	Ò
132	84h	ä	164	A4h	ñ	196	C4h	─	228	E4h	õ
133	85h	à	165	A5h	Ñ	197	C5h	┼	229	E5h	Õ
134	86h	å	166	A6h	ª	198	C6h	ã	230	E6h	µ
135	87h	ç	167	A7h	º	199	C7h	Ã	231	E7h	þ
136	88h	ê	168	A8h	¿	200	C8h	╚	232	E8h	Þ
137	89h	ë	169	A9h	®	201	C9h	╔	233	E9h	Ú
138	8Ah	è	170	AAh	¬	202	CAh	╩	234	EAh	Û
139	8Bh	ï	171	ABh	½	203	CBh	╦	235	EBh	Ù
140	8Ch	î	172	ACh	¼	204	CCh	╠	236	ECh	ý
141	8Dh	ì	173	ADh	¡	205	CDh	═	237	EDh	Ý
142	8Eh	Ä	174	AEh	«	206	CEh	╬	238	EEh	¯
143	8Fh	Å	175	AFh	»	207	CFh	¤	239	EFh	´
144	90h	É	176	B0h	░	208	D0h	ð	240	F0h	
145	91h	æ	177	B1h	▒	209	D1h	Ð	241	F1h	±
146	92h	Æ	178	B2h	▓	210	D2h	Ê	242	F2h	‗
147	93h	ô	179	B3h	│	211	D3h	Ë	243	F3h	¾
148	94h	ò	180	B4h	┤	212	D4h	È	244	F4h	¶
149	95h	ò	181	B5h	Á	213	D5h	ı	245	F5h	§
150	96h	û	182	B6h	Â	214	D6h	Í	246	F6h	÷
151	97h	ù	183	B7h	À	215	D7h	Î	247	F7h	¸
152	98h	ÿ	184	B8h	©	216	D8h	Ï	248	F8h	°
153	99h	Ö	185	B9h	╣	217	D9h	┘	249	F9h	¨
154	9Ah	Ü	186	BAh	║	218	DAh	┌	250	FAh	·
155	9Bh	ø	187	BBh	╗	219	DBh	█	251	FBh	¹
156	9Ch	£	188	BCh	╝	220	DCh	▄	252	FCh	³
157	9Dh	Ø	189	BDh	¢	221	DDh	¦	253	FDh	²
158	9Eh	×	190	BEh	¥	222	DEh	Ì	254	FEh	■
159	9Fh	ƒ	191	BFh	┐	223	DFh	▀	255	FFh	

Quelle: wikimedia.org

4 - Zahlensysteme und Kodierung

Unicode

- Was ist aber z.B. mit chinesischen, griechischen oder arabischen Buchstaben?
- Dafür gibt es den internationalen Unicode Standard (bis 4 Byte)
- UTF bedeutet „Unicode Transform Format“ und enthält auch ASCII
- UTF8 (1-4 Bytes) am häufigsten verwendet, im Internet, in fast allen Betriebssystemen
- Im UTF16 (2-4 Byte) z.B. Zeichenkodierung der Programmiersprachen Java, .NET
- Der Unicode Standard wird nach und nach erweitert
  (siehe auch: https://de.wikipedia.org/wiki/Unicode)

4 - Zahlensysteme und Kodierung

Datentypen

• Logische Werte:	boolean (true = 1, false = 0)	✓
• Positive ganze Zahlen:	unsigned integer (0, 1, 2, 3, 4…)	✓
• Ganze Zahlen:	integer (...-3, -2, -1, 0, 1, 2, 3…)	✓
• Gleitkommazahlen:	floatingpoint (+/- $x \cdot 2^{exp}$)	✓
• Einzelne Zeichen	character („a“, „b“, „c“,...„#“,...„*“„&“…)	✓
• Zeichenketten	Strings („Hello World“…)	???

4 - Zahlensysteme und Kodierung

Zeichenketten

- Bei Zeichenketten stehen die einzelnen Zeichen im ASCII oder Unicode Format hintereinander im Speicher.
- In älteren Programmiersprachen wie C wird eine Zeichenkette (engl. String) mit einem NULL-Zeichen (auch ‘/0‘ geschrieben) abgeschlossen.
- In Java und .net werden dagegen die Zeichen der Zeichenkette und die Länge der Zeichenkette abgespeichert.

4 - Zahlensysteme und Kodierung

Datentypen

• Logische Werte:	boolean (true = 1, false = 0)	✓
• Positive ganze Zahlen:	unsigned integer (0, 1, 2, 3, 4…)	✓
• Ganze Zahlen:	integer (...-3, -2, -1, 0, 1, 2, 3…)	✓
• Gleitkommazahlen:	floatingpoint (+/- $x \bullet 2^{exp}$)	✓
• Einzelne Zeichen	character („a“, „b“, „c“,...„#“,...„*“„&“…)	✓
• Zeichenketten	Strings („Hello World“…)	✓

4 - Zahlensysteme und Kodierung

Dateien

• Jede Information mit der der Rechner umgeht lässt sich als eine Folge von Bytes repräsentieren und kann als Datei beliebiger Länge gespeichert werden

4 - Zahlensysteme und Kodierung

Dateispeichergrößen

- Dateigrößen werden mit der Anzahl der enthaltenen Bytes angegeben
  Eine 1 KByte Datei enthält demnach 1000 Byte
  ... manchmal aber auch einen Faktor von 1024 (2^{10}).

Name	Symbol	Wert
Kilo	k	$10^3 = 1\ 000$
Mega	M	$10^6 = (1000)^2 = 1\ 000\ 000$
Giga	G	$10^9 = (1000)^3 = 1\ 000\ 000\ 000$
Tera	T	$10^{12} = (1000)^4 = 1\ 000\ 000\ 000\ 000$
Peta	P	$10^{15} = (1000)^5 = 1\ 000\ 000\ 000\ 000\ 000$
Exa	E	$10^{18} = (1000)^6 = 1\ 000\ 000\ 000\ 000\ 000\ 000$

4 - Zahlensysteme und Kodierung

Binäre Präfixe (nach IEC 60027-2, seit 1990)

Name	Symbol	Wert
kibi	ki	2^{10} =1 024
mebi	Mi	2^{20} =$(1024)^2$=1 048 576
gibi	Gi	2^{30} =$(1024)^3$=1 073 741 824
tebi	Ti	2^{40} =$(1024)^4$=1 099 511 627 776
pebi	Pi	2^{50} =$(1024)^5$=1 125 899 906 842 624
exbi	Ei	2^{60} =$(1024)^6$=1 152 921 504 606 846 976

Noch mal zum Vergleich die Präfixe mit Zehnerpotenz

Name	Symbol	Wert
Kilo	k	10^3 = 1 000
Mega	M	10^6 = $(1000)^2$=1 000 000
Giga	G	10^9 = $(1000)^3$=1 000 000 000
Tera	T	10^{12} = $(1000)^4$=1 000 000 000 000
Peta	P	10^{15} = $(1000)^5$=1 000 000 000 000 000
Exa	E	10^{18} = $(1000)^6$=1 000 000 000 000 000 000

IEC: International Electrotechnical Commission

4 - Zahlensysteme und Kodierung

Was braucht denn ungefähr wieviel Speicher?

WAS?	Codierung	Speicherbedarf
Text Datei	1-2 Byte pro Buchstabe	Wenige kByte
Audio Datei (Song 2 min)	komprimiert unkomprimiert	1-3 MByte 10-20 MByte
Bild Datei (Foto 12 Megapixel)	komprimiert unkomprimiert	3 MByte 20 MByte
Videodatei (Spielfilm)	komprimiert	2-3 GByte

4 - Zahlensysteme und Kodierung

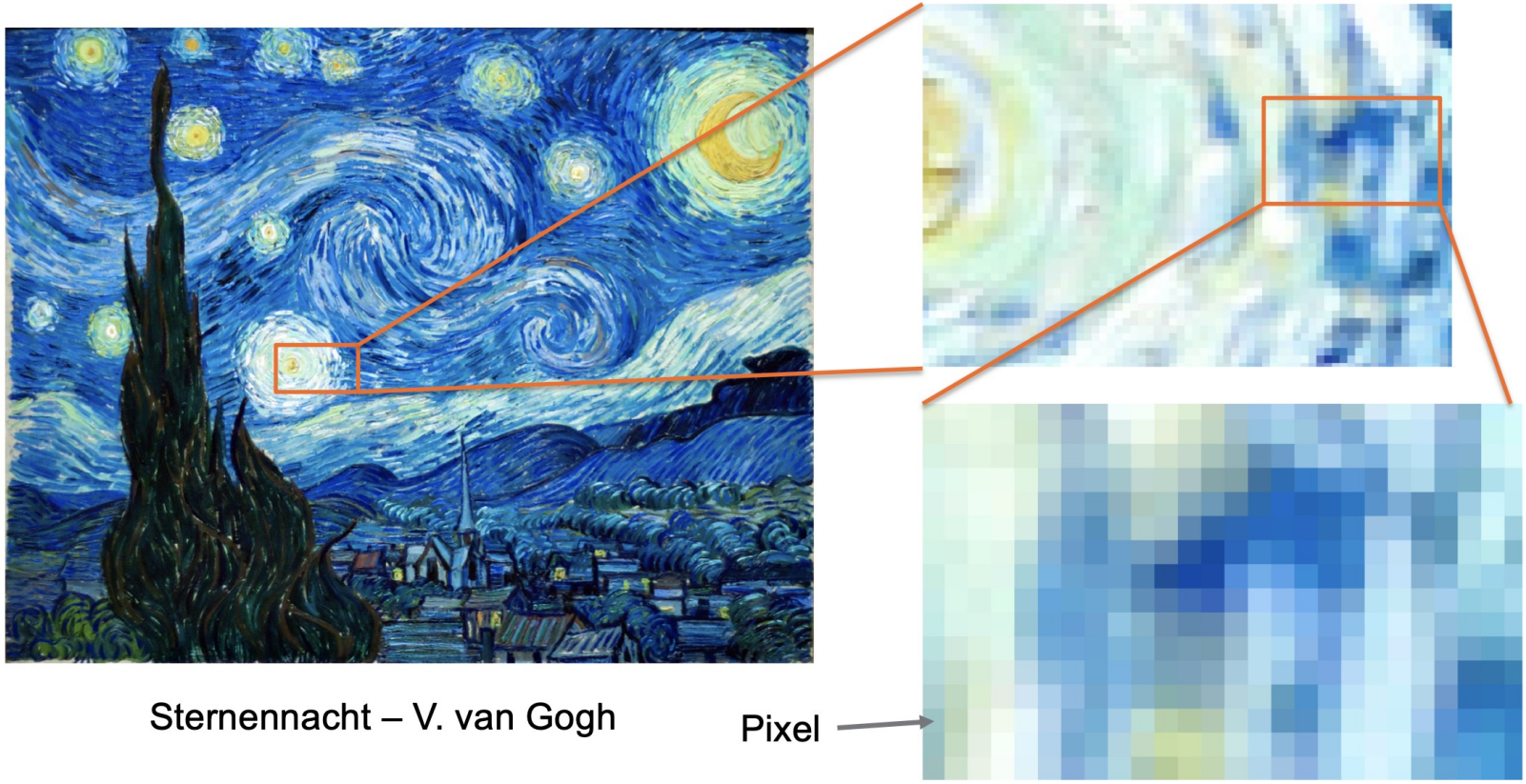
Binäre und Textfiles

- Unter Binärfiles versteht man, dass jedes der Bytes des Files alle Werte zwischen 0 und 255 annehmen kann
  Beispiele: .doc, .xls, . ppt, .gif., wav, .mov, .eps, .avi….

- Textfiles sind Files, die nur die üblichen ASCII Zeichen benutzen und die Information in Buchstaben, Zahlen und Sonderzeichen der ASCII Tabelle kodieren
  Beispiele:
  .txt, .rtf, .csv, .ppm, .html, .c, .cpp, .h, . java, .js, .php ...

4 - Zahlensysteme und Kodierung

Bilder als Rastergrafiken



4 - Zahlensysteme und Kodierung

Bilder als Rastergrafiken

- Es gibt zahlreiche Dateiformate für Bilder wie jpg, bmp, png, gif, ppm…
- Viele bieten die Möglichkeit zur Bildkompression mit komplexen Algorithmen
- Die einfachsten textbasierten sind die Portable Anymap Formate

  - Portable Bitmap (PBM)
  - Portable Graymap (PGM)
  - Portable Pixmap (PPM)

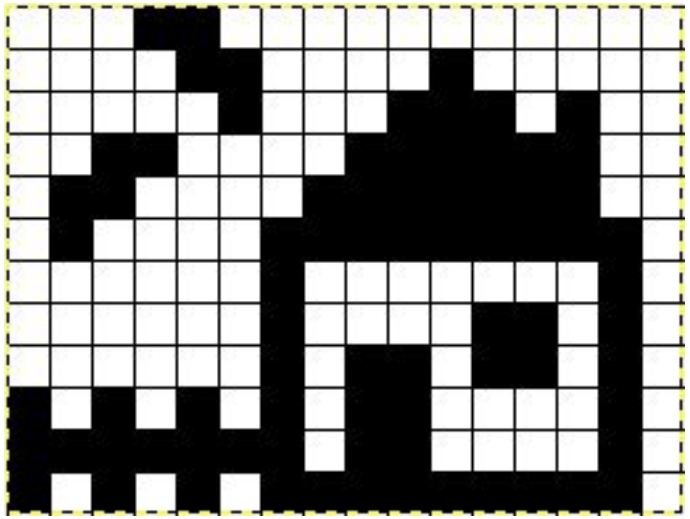


Bild Quelle:https://lehrerfortbildung-bw.de/u_matnatech/informatik/gym/bp2016/fb1/1_daten_code/2_kopier/8_grafik/

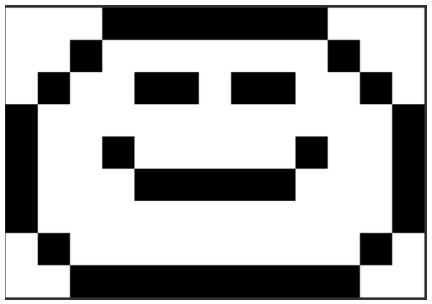
4 - Zahlensysteme und Kodierung

Bilder als Rastergrafiken

Portable Bitmap (PBM)

Öffne einen Texteditor und kopiere das unten stehende Beispiel in den Editor.
Speichere das File als **bild.pbm** ab.
Zum anschauen kannst du es z.B. mit Gimp (https://www.gimp.org/) öffnen

```
P1            #Bitmap
13 9          #Breite des Bildes, Leerstelle, Höhe des Bildes
0 0 0 1 1 1 1 1 1 1 0 0 0
0 0 1 0 0 0 0 0 0 0 1 0 0
0 1 0 0 1 1 0 1 1 0 0 1 0
1 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 1 0 0 1
1 0 0 0 1 1 1 1 1 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 1
0 1 0 0 0 0 0 0 0 0 0 1 0
0 0 1 1 1 1 1 1 1 1 1 0 0
```



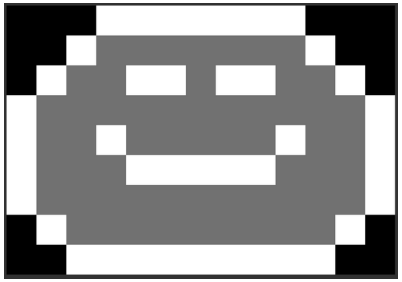
4 - Zahlensysteme und Kodierung

Bilder als Rastergrafiken

Portable Graymap (PGM)

Öffne einen Texteditor und kopiere das unten stehende Beispiel in den Editor.
Speichere das File als **bild.pgm** ab.
Zum anschauen kannst du es z.B. mit Gimp (https://www.gimp.org/) öffnen

```
P2              #Graymap
13 9            #Breite des Bildes, Leerstelle, Höhe des Bildes
9               #Helligkeitswerte bis 9
0 0 0 9 9 9 9 9 9 9 0 0 0
0 0 9 4 4 4 4 4 4 4 9 0 0
0 9 4 4 9 9 4 9 9 4 4 9 0
9 4 4 4 4 4 4 4 4 4 4 4 9
9 4 4 9 4 4 4 4 4 9 4 4 9
9 4 4 4 9 9 9 9 9 4 4 4 9
9 4 4 4 4 4 4 4 4 4 4 4 9
0 9 4 4 4 4 4 4 4 4 4 9 0
0 0 9 9 9 9 9 9 9 9 9 0 0
```

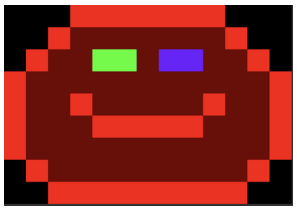


4 - Zahlensysteme und Kodierung

Bilder als Rastergrafiken

Portable Pixmap (PPM)

Öffne einen Texteditor und kopiere das unten stehende Beispiel in den Editor.
Speichere das File als **bild.ppm** ab.
Zum anschauen kannst du es z.B. mit Gimp (https://www.gimp.org/) öffnen



```
P3            #Portable Pixmap
13 9          #Breite des Bildes, Leerstelle, Höhe des Bildes
9             #Helligkeitswerte bis 9
0 0 0 0 0 0 0 0 0 9 0 0 9 0 0 9 0 0 9 0 0 9 0 0 9 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 9 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 9 0 0 0 0 0 0 0 0
0 0 0 9 0 0 4 0 0 4 0 0 0 9 0 0 9 0 4 0 0 4 0 9 4 0 9 4 0 0 4 0 0 9 0 0 0 0 0
9 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 9 0 0
9 0 0 4 0 0 4 0 0 9 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 9 0 0 4 0 0 4 0 0 9 0 0
9 0 0 4 0 0 4 0 0 4 0 0 9 0 0 9 0 0 9 0 0 9 0 0 9 0 0 4 0 0 4 0 0 4 0 0 9 0 0
9 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 9 0 0
0 0 0 9 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 4 0 0 9 0 0 0 0 0
0 0 0 0 0 0 9 0 0 9 0 0 9 0 0 9 0 0 9 0 0 9 0 0 9 0 0 9 0 0 9 0 0 0 0 0 0 0 0
```

rot grün blau rot grün blau...

4 - Zahlensysteme und Kodierung

Bilder als Vektorgrafiken

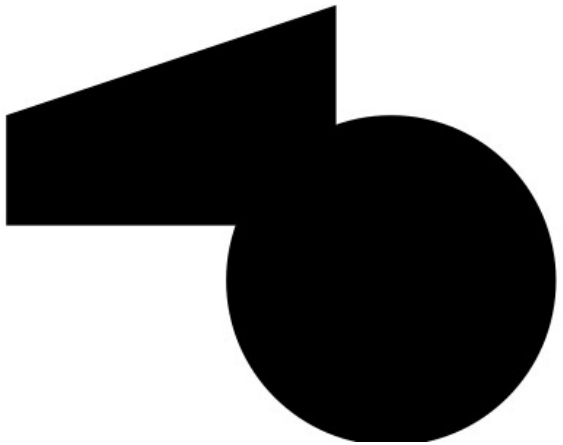
Scalable Vector Graphics

Vektorgrafiken lassen sich gegenüber Pixelgrafiken beliebig skalieren!

Öffne einen Texteditor und kopiere das unten stehende Beispiel in den Editor.
Speichere das File als **bild.svg** ab.
Zum anschauen kannst du es z.B. mit inkscape (https://www.inkscape.org/) öffnen

```
<?xml version="1.0" encoding="UTF-8"?>
<svg xmlns="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    version="1.1" baseProfile="full"
    width="800mm" height="600mm"
    viewBox="-400 -300 800 600">
    <title>Titel der Datei</title>
    <desc>Beschreibung/Textalternative zum Inhalt.</desc>
<!--Inhalt der Datei -->
    <circle cx="225" cy="225" r="75" />
    <polyline points="50,150 50,200 200,200 200,100" />

</svg>
```



4 - Zahlensysteme und Kodierung

Textverarbeitung (Word, Libre Office)

Rich Text File format

Öffne einen Texteditor und kopiere das unten stehende Beispiel in den Editor.
Speichere das File als **text.rtf** ab.
Zum anschauen kannst du es z.B. mit Wordpad, Word oder Libre Office öffnen.

```
{\rtf1\ansi\deff0
{\colortbl;\red0\green0\blue0;\red255\green0\blue0;}
This line is the default color\line
\cf2
This line is red\line
\cf1
This line is the default color
}
```

This line is the default color
This line is red
This line is the default color

4 - Zahlensysteme und Kodierung

Tabellenkalkulation (Excel, Libre Office Spreadsheet)

Comma-separated values (CSV)

Öffne einen Texteditor und kopiere das unten stehende Beispiel [Quelle: Wikipedia] in den Editor.
Speichere das File als **spreadsheet.csv** ab.
Zum anschauen kannst du es z.B. mit Excel oder Libre Office öffnen.

```
Stunde,Montag,Dienstag,Mittwoch,Donnerstag,Freitag
1,Mathematik,Deutsch,Englisch,Erdkunde,Politik
2,Sport,Deutsch,Englisch,Sport,Geschichte
3,Sport,"Religion (ev., kath.)",Kunst,,Kunst
```

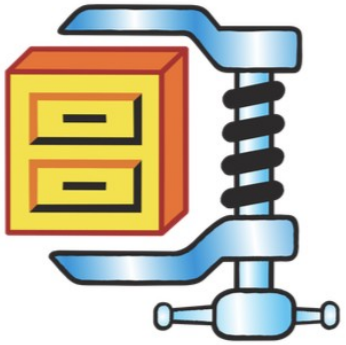
Stunde	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
1	Mathematik	Deutsch	Englisch	Erdkunde	Politik
2	Sport	Deutsch	Englisch	Sport	Geschichte
3	Sport	Religion (ev., kath.)	Kunst		Kunst

4 - Zahlensysteme und Kodierung

Komprimierung

Die meisten Dateiformate sind aber binär und oft noch komprimiert um Speicherplatz zu sparen. Aber wie funktioniert das?

Komprimierung



Verlustfreie Komprimierung (lossless)

- RLE
- Huffman Coding
- LZW

Verlustbehaftete Komprimierung (lossy)

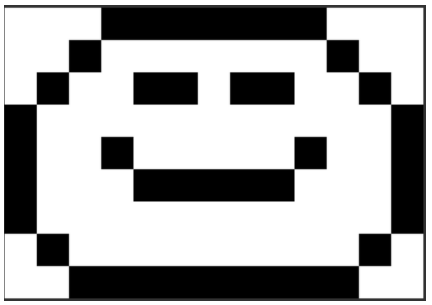
- MPEG
- MP3
- JPG

4 - Zahlensysteme und Kodierung

Komprimierung (verlustfrei)

RLE (Run Length Encoding)
- aufeinanderfolgende Symbole werden über ihre Anzahl kodiert
- das ist einfach, schnell

```
Beispiel: Bild mit 13•9 = 117 Zahlen
0 0 0 1 1 1 1 1 1 1 0 0 0
0 0 1 0 0 0 0 0 0 0 1 0 0
0 1 0 0 1 1 0 1 1 0 0 1 0
1 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 1 0 0 1
1 0 0 0 1 1 1 1 1 0 0 0 1
1 0 0 0 0 0 0 0 0 0 0 0 1
0 1 0 0 0 0 0 0 0 0 0 1 0
0 0 1 1 1 1 1 1 1 1 1 0 0
```



```
Kodiert in 74 Zahlen:
0 3 1 7 0 5 1 1 0 7 1 1 0 3 1 1 0 2 1 2 0 1 1 2 0 2 1 1 0 1 1 1 0 11
1 2 0 2 1 1 0 5 1 1 0 2 1 2 0 3 1 5 0 3 1 2 0 11 1 1 0 1 1 1 0 9 1 1
0 3 1 9 0 2
```

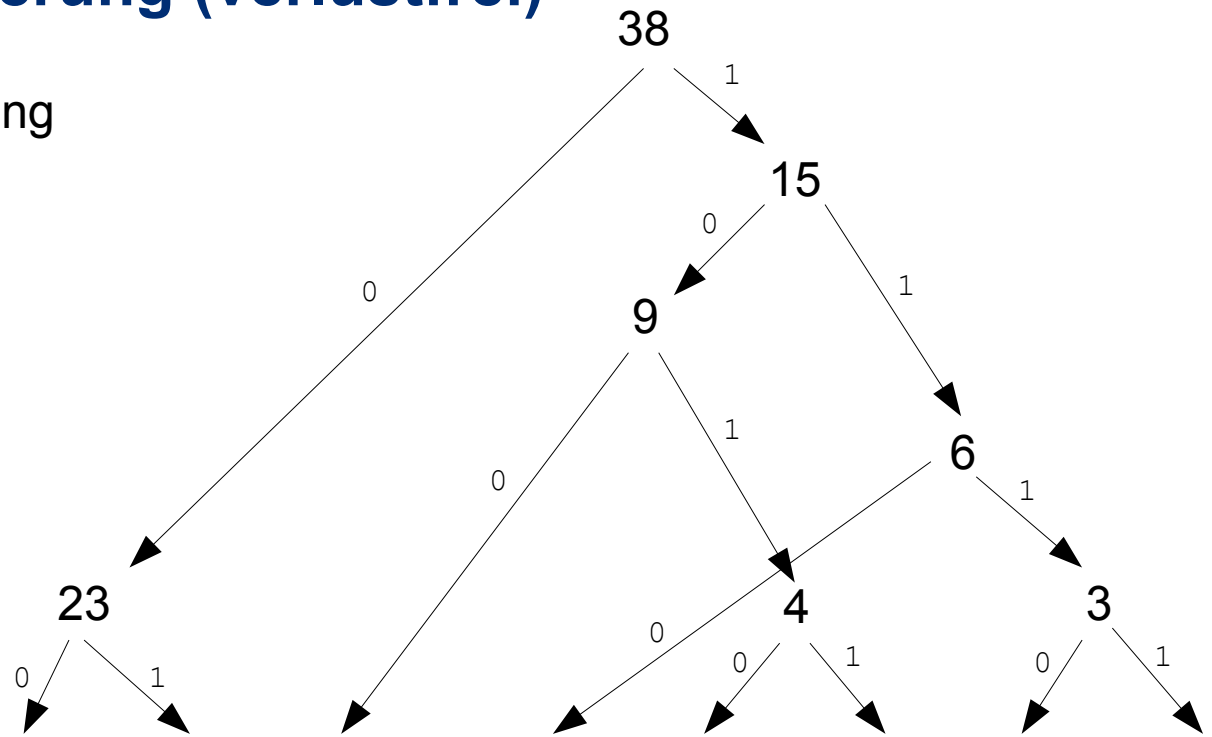
start-wert

```
Da Schwarz und Weiß sich abwechseln braucht man nur die Hälfte (37+1 Zahlen):
0 3 7 5 1 7 1 3 1 2 2 1 2 2 1 1 1 11
2 2 1 5 1 2 2 3 5 3 2 11 1 1 1 9 1
3 9 2
```

4 - Zahlensysteme und Kodierung

Komprimierung (verlustfrei)

Huffman Coding



Zahl	1	2	3	5	7	9	11	0
Häufig-keit	13	10	5	3	2	2	2	1
Kodierung	00	01	100	110	1010	1011	1110	1111

4 - Zahlensysteme und Kodierung

Komprimierung (verlustfrei)

Huffman Coding

```
Beispiel: RLE kodiertes Bild mit 38*8 bit = 304 bit
0 3 7 5 1 7 1 3 1 2 2 1 2 2 1 1 1 11
2 2 1 5 1 2 2 3 5 3 2 11 1 1 1 9 1
3 9 2

=>
11111001010100001010001000001010001010000001110          47 Bit
01010010000010110011010001111000000101100               +43 Bit
100101101                                                + 9 Bit
                                                          _______
                                                304 Bit → 99 Bit
```

Zahl	1	2	3	5	7	9	11	0
Häufig-keit	13	10	5	3	2	2	2	1
Kodierung	00	01	100	110	1010	1011	1110	1111

Siehe auch: https://www.youtube.com/watch?v=JsTptu56GM8

4 - Zahlensysteme und Kodierung

Komprimierung (verlustfrei)

LZW Codierung (Lempel, Ziv, Welch)

Prinzip:
- Wörterbuch mit 12 Bit Index wird mit neuen Bytesequenzen gefüllt
- Bytesequenzen die sich wiederholen können so zunehmend verkürzt werden

```
Beispiel: ababc
Ursprungskodierung: a=1, b=2, c=3
lese a
lese b
schreibe ab=4 ins Wörterbuch
nachfolgendes ab wird mit 4 kodiert
lese c
→ komprimiert: 1243
Das Wörterbuch muss nicht übertragen werden, weil es bei der
Dekompression dynamisch mit generiert werden kann!
```

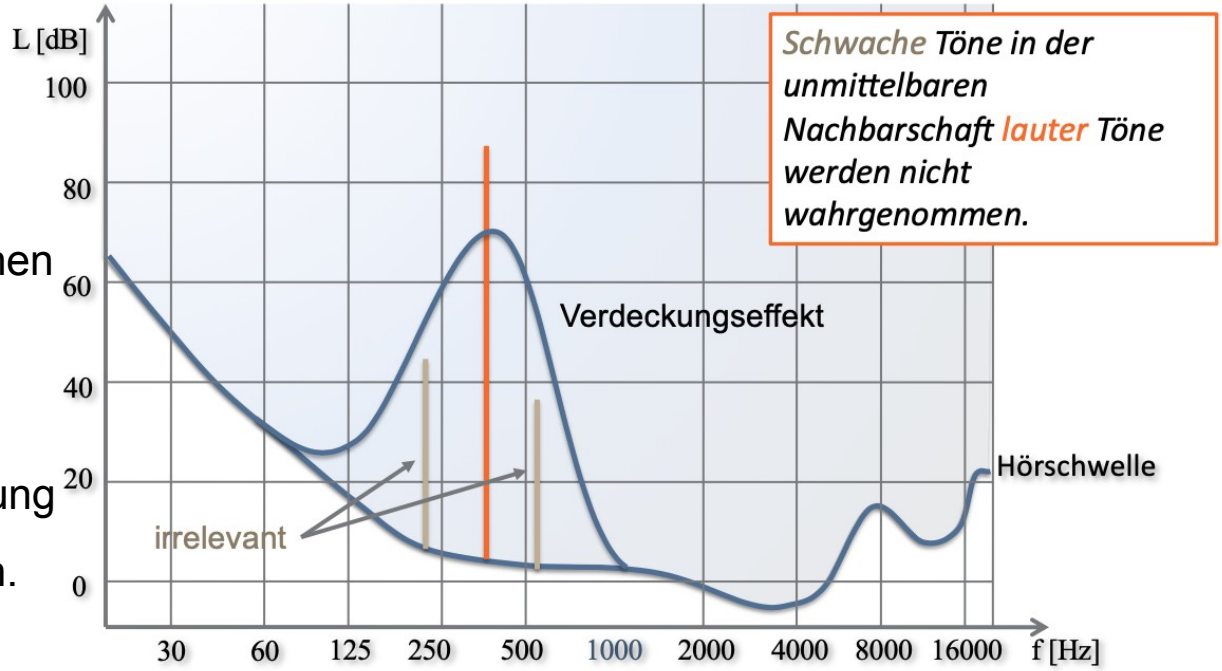
Eingesetzt bei
- ZIP-Komprimierung von (beliebigen) Dateien, Grafikformaten (z.B. GIF)
- gängigstes Verfahren für eine „Allzweck-Komprimierung“

4 - Zahlensysteme und Kodierung

Komprimierung (verlustbehaftet)

Audio (MP3 oder MPEG Audio Layer 3)

Prinzip: Es werden die Informationen weggelassen, die das menschliche Gehör nicht wahrnimmt. Stereo Signale werden mit Differenz-signalen kodiert und zum Schluss werden die Informationen mit dem Huffman Algorithmus komprimiert.
Dadurch wird eine Komprimierung etwa um den Faktor 10 möglich.



4 - Zahlensysteme und Kodierung

Komprimierung (verlustbehaftet)

Pixelgrafik (JPG)

Prinzip: Umwandlung in 4 Schritten

1 – RGB (rot, grün, blau) → YUV (Helligkeit, U-Blaudifferenz, V-Rotdifferenz)

2 – 8x8 Pixelblöcke mittels Diskreter Cosinustransformation → 64 Koeffizienten

3 – Koeffizienten werden je nach Frequenz unterschiedlich gerundet (komprimiert)

4 – Die resultierenden komprimierten Koeffizienten werden nun noch einmal verlustfrei komprimiert. Dabei wird der RLE und Huffman Algorithmus in Kombination genutzt

4 - Zahlensysteme und Kodierung

Komprimierung (verlustbehaftet)

Pixelgrafik (JPG)

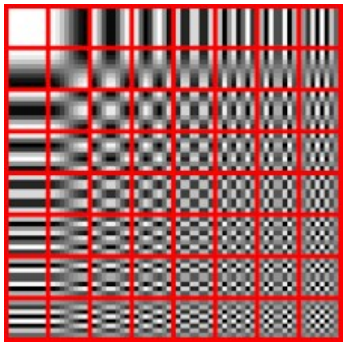
Prinzip: Umwandlung in 4 Schriten

1 – RGB (rot, grün, blau) → YUV (Helligkeit, U-Blaudifferenz, V-Rotdifferenz)

2 – 8x8 Pixelblöcke mittels Diskreter Cosinustransformation → 64 Koeffizienten

3 – Koeffizienten werden je nach Frequenz unterschiedlich gerundet (komprimiert)

4 – Die resultierenden komprimierten Koeffizienten werden nun noch einmal verlustfrei komprimiert. Dabei wird der RLE und Huffman Algorithmus in Kombination genutzt



Bildquelle: Wikipedia

5 - Betriebssysteme

Kapitel 5 - Betriebssysteme

- Überblick
- Architektur von Betriebssystemen
- Prozesse und Threads
- Speicherverwaltung
- Scheduling/Multitasking
- Dateisystem
- Terminal

5 - Betriebssysteme

Überblick: Wofür braucht man ein Betriebssystem

- Es sorgt für die ordentliche Konfiguration der Hardware
- Es bietet i.Allg. Treiber um bestimmte Hardware-Komponenten anzusprechen
- Es beschränkt Hardwarezugriffe im USER-Mode
- Es implementiert das Memory-Management
- Es bietet oftmals eine Grafische Oberfläche oder eine Konsole um die Kommunikation mit dem Nutzer zu erlauben
- Es managed das Multitasking von Prozessen*
- Es bietet Funktionen für Ein-/Ausgabe z.B. um auf das Filesystem zuzugreifen
- Es verwaltet mehrere Nutzer*

* Diese Funktionalität ist nicht in allen Betriebssystemen enthalten

5 - Betriebssysteme

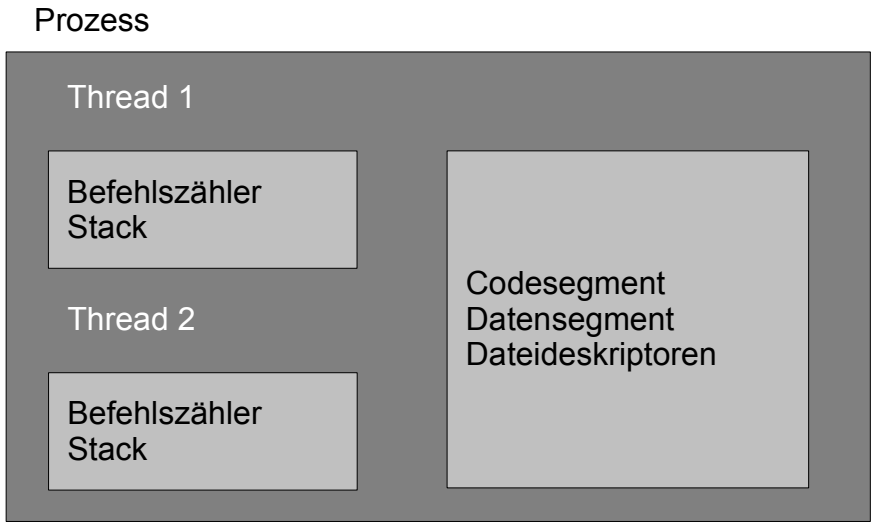
Architektur von Betriebssystemen

- Monolithische Architektur: alle wesentlichen Komponenten in einem homogenen Code, effizient aber wenig flexibel anpassbar
- Kern-Schale-Architektur: wichtigsten Komponenten im Kern (z.B. ist bei Linux die Prozessverwaltung im Kern (engl. Kernel) und der Kommando-Interpreter in der Schale (Shell))
- Hierarchische Schichten (Mehrschichtenmodell): jede Schicht bietet der darüberliegenden Schicht Dienstleistungen, Schichten austauschbar
- Mikrokern (micro kernel): bildet nur eine Art Infrastruktur mit minimalem Funktionsumfang, z.B. bei Client-Server-Modellen

5 - Betriebssysteme

Prozesse und Threads

- Prozesse sind Programme. Im Multitasking Betriebssystem können mehrere Prozesse (engl. task) parallel (gleichzeitig) laufen.
- Threads sind parallel ablaufende Routinen oder Unterprogramme. Sie laufen meist innerhalb eines Prozesses und teilen sich Codesegment, Datensegment, Dateideskriptoren



Speicherverwaltung

- Ein üblicher Computer hat im Prozessor eine Memory Management Unit (MMU)
- Diese ermöglicht es Programme in einem virtuellen Adressraum zu legen
- Das bedeutet, dass die reale Adresse an der ein Program liegt eine andere ist, als die virtuelle Adresse, so dass absolute Sprünge (an eine Adresse), Sprünge an die virtuelle Adresse sind
- Zusätzlich gibt es meist einen DMA (Direct Memory Access), mit Hilfe deren es möglich ist Speicher zu kopieren, ohne Rechenressourcen zu blockieren.

Scheduling/ Multitasking

- Multitasking heisst, dass man vom Prozessor mehrere Prozesse parallel oder quasiparallel ausführen lassen kann
- Das funktioniert z.B. für einen Prozessor mit einem Rechenkern folgendermassen:

  - Ein Prozess wird ausgeführt

  - Nach einer vorgegebenen Zeit wird dieser von einem Timer-Interrupt unterbrochen

  - Die Register und der Program Counter (PC) werden gesichert

  - Die Register und der PC vom nächsten geplanten Prozess werden geladen

  - Der nächste Prozess wird ausgeführt, bis der nächste Timer-Interrupt kommt

5 - Betriebssysteme

Dateisystem

- Ein Filesystem ermöglicht das anlegen und beschreiben von Ordnern und Dateien
- Dafür hat jedes Filesystem eine File Allocation Table, eine art Tabelle, in der steht, wie der Ordner oder das File heisst, wie groß das File ist und an welcher Adresse der Festplatte/SSD/SD-CARD etc. das File beginnt.
- Meist wird ein Order wie eine Datei behandelt, in denen Referenzen auf die enthaltenen Dateien und Ordner stehen.
- Oftmals werden in der File Allocation Table auch der Nutzer, der die Datei angelegt hat und die Rechte einer Datei festgelegt

Das Terminal

- Früher gab es keine grafischen Benutzeroberflächen. Jeder Befehl wurde einzeln eingegeben und ausgeführt.
- Da diese Befehlseingabe über die Kommandozeile sehr effektiv sein kann, wurde in den Grafischen Benutzeroberflächen meist ein Terminal Fenster zur Befehlseingabe hinzugefügt.
- Unter Windows kann man im Startmenü einfach **cmd** eingeben
- Unter Linux findet man das Terminal im Startmenü oder mittels shortcut
- Unter MacOS kann man **command-space** drücken und **terminal** eingeben

5 - Betriebssysteme

Das Terminal

Was?	Windows	Linux MacOSx
Inhaltsverzeichnis der HDD anzeigen	dir	ls -l
Datei kopieren	copy quelle ziel	cp quelle ziel
Datei löschen	del datei	rm datei
Verzeichnis erstellen	mkdir dirName	md dirName
Programm starten	progName.exe	./progName

6 - Rechnernetze

Kapitel 6 - Rechnernetze

- Komponenten eines Netzwerkes
- Funktionsweise eines Netzwerkes
- Domain Name Service (DNS)
- TCP/IP Model
- Open System Interconnection (OSI) Model

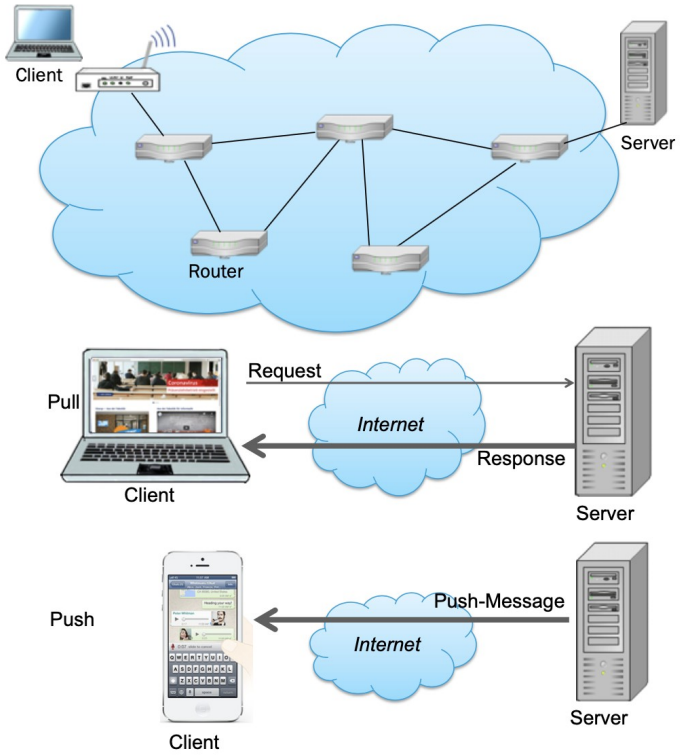
6 - Rechnernetze

Komponenten eines Netzwerkes

Hosts
- Clients bekommt Informationen und fordert welche an
- Server stellt Informationen (auf Anforderung) bereit

Router
- Verbindungs- und Weiterleitungsknoten

Kommunikationsarten
- Pull Kommunikation (Request → Response)
- Push Kommunikation (Ereignisgesteuert)



Funktionsweise eines Netzwerkes

LAN ist die Abkürzung für Local Area Network (bis zum eigenen Router)
WAN ist die Abkürzung für Wide Area Network (bis zum Internet Service Provider)

- Jede Netzwerkkarte hat eine MAC Adresse, die sich nicht ändern lässt.
- In einem Netzwerk bekommt jeder Computer dynamisch eine IP Adresse zugewiesen (IPv4 entspricht 4 Byte, z.B. 192.168.0.255)
- Wenn man mehrere Computer vernetzt und eine Nachricht an einen anderen Computer senden möchte, dann sendet man die Nachricht an alle Computer im Netzwerk. Die Nachricht selbst enthält die IP Adresse des Empfängers. Alle Computer mit anderen IP Adressen ignorieren die Nachricht.
- Bei vielen Computern und vielen zu übertragenen Daten kommt es jedoch häufig zu Kollisionen. Wenn die Leitung belegt ist, muss der Computer warten. Um erneute Kollisionen zu vermeiden, wird die Wartezeit zufällig variiert. Ausserdem wird, wenn die Leitung erneut belegt ist, die Wartezeit exponentiell erhöht (z.B. immer verdoppelt)

Funktionsweise eines Netzwerkes (IP-Adressen)

- Zur Identifizierung eines Gerätes im Netzwerk gibt es IP-Adressen
- Die IP-Adressen können in 4 Bytes (IPv4) oder 8 Worten (IPv6) kodiert sein
- IPv4 Beispieladresse: 192.168.10.1 (4 Bytes)
- Eine Subnetzmaske 255.255.240.0 definiert, wieviele Bits zur Identifizierung des Subnetzes gebraucht werden (alle, an denen eine 1 steht, hier 20 Bits) und wieviele für die Geräte im Netz zur Verfügung stehen (hier 12 Bits).

- IPv6 Beispieladresse:

```
2001 : 0db8 : 3c4d : 0015 : 0000 : 0000 : 1a2f : 1a2b
  Standortpräfix    :NetzID:       Schnittstellen ID
```

- In der Schnittstellen ID ist die MAC Adresse der Netzwerkkarte kodiert. Die Privacy Extension sorgt dafür, dass die MAC Adresse nicht im Klartext zu sehen ist, sondern sich ihre Kodierung sogar in definierten Zeitabständen ändern kann.

Funktionsweise eines Netzwerkes

- Um den Informationsfluss weiter zu verbessern, werden die Daten in Pakete zerteilt und am Ende wieder zusammengefügt
- Zudem wird das Netzwerk mittels Router in Teilnetze geteilt. Der Router sorgt dafür, dass nicht immer alle anderen Computer die Nachrichten bekommt, sondern dass die Nachricht über einen freien Weg vom Sender zum Empfänger geroutet wird.
- Falls es Routingfehler gibt und Nachrichten z.B. zwischen zwei Routern hin und her laufen, dann kann das detektiert werden, denn der sogenannte Hop-Count zählt die Anzahl der Zwischenstationen. Ist dieser Hop-Count größer als das sogenannte Hop-Limit, stimmt etwas mit dem Routing nicht.

6 - Rechnernetze

Funktionsweise eines Netzwerkes

```
Jorns-MacBook-Pro-2:~ fischer$ traceroute google.com
traceroute to google.com (216.58.207.46), 64 hops max, 52 byte packets
 1  192.168.0.1 (192.168.0.1)  3.848 ms  3.601 ms  3.786 ms
 2  * * *
 3  84.116.190.89 (84.116.190.89)  29.818 ms  14.659 ms  15.137 ms
 4  de-fra04d-rc1-ae-60-0.aorta.net (84.116.191.221)  16.455 ms  18.562 ms  15.283 ms
 5  de-fra11b-ri1-ae-5-0.aorta.net (84.116.134.217)  21.296 ms  13.589 ms  13.640 ms
 6  213.46.179.106.aorta.net (213.46.179.106)  13.424 ms  15.090 ms  16.299 ms
 7  * * *
 8  108.170.252.1 (108.170.252.1)  23.292 ms
    108.170.235.250 (108.170.235.250)  18.211 ms
    216.239.40.56 (216.239.40.56)  24.244 ms
 9  72.14.233.47 (72.14.233.47)  23.062 ms
    72.14.232.51 (72.14.232.51)  14.250 ms
    72.14.233.47 (72.14.233.47)  16.477 ms
10  72.14.239.167 (72.14.239.167)  14.243 ms
    209.85.242.79 (209.85.242.79)  14.386 ms
    108.170.236.249 (108.170.236.249)  27.522 ms
11  fra16s24-in-f14.1e100.net (216.58.207.46)  21.546 ms
    209.85.240.112 (209.85.240.112)  15.040 ms
    fra16s24-in-f14.1e100.net (216.58.207.46)  15.504 ms
```

6 - Rechnernetze

Domain Name Services (DNS)

- Adressen wie www.wikipedia.org, werden über einen DNS in IP-Adressen und Portnummer umgewandelt.
- Der Nameserver befindet sich typischerweise beim Internet Service Provider
- Die Domänen, die dort gespeichert sind gliedern sich in
  - Top Level Domains (.org, .gov, .net, .de, .edu, …),
  - Second Level Domains (google.com, dftba.com)
  - Sub Domains (drive.google.com, images.google.com, store.dftba.com)

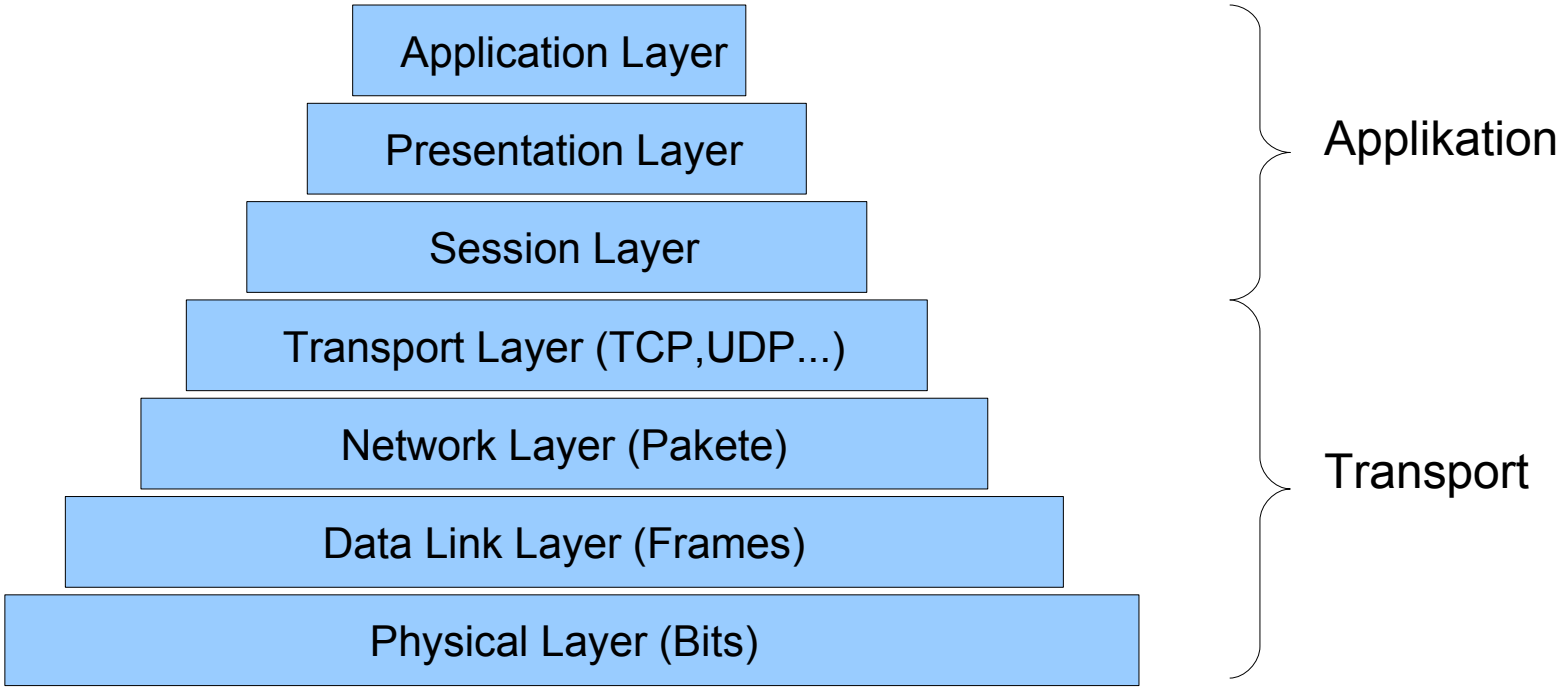
  in einem riesigen Baum mit 300 Millionen Domainnamen alleine in den Second Level Domains

TCP/IP Model

- Das **Internet Protokol (IP)** sorgt dafür, dass die Daten an den richtigen Computer gelangen
- Das **User Datagram Protocol (UDP)** kann in dem IP eingebettet werden und sorgt dafür, dass die Daten an den richtigen Port und somit ans richtige Programm kommen.
- UDP enthält auch eine Checksumme der Daten, die eine sichere Übertragung der Daten ermöglicht.
- Das **Transmission Control Protocol (TCP)** wird wie UDP in den Datenteil des Internet Protokol geschrieben. Die Kombination wird dann TCP/IP genannt.
- TCP enthält neben der Portnummer und der Checksumme auch die Paketnummer, damit die Pakete in der richtigen Reihenfolge zusammengebaut werden können.
- Ausserdem wartet der Sender auf das Acknowledgment des Empfängers, dass das Paket richtig angekommen ist. Falls das nicht der Fall ist, sendet der Sender das Packet nach einer festgelegten Zeit noch einmal.

6 - Rechnernetze

Open System Interconnection (OSI)-Model



6 - Rechnernetze

Open System Interconnection (OSI)-Model

Application Layer:	Stellt Funktionen für die Anwendung zur Verfügung
Presentation Layer:	Umwandlung der Daten in unabhängiges Format
Session Layer:	Steuerung der Verbindungen und des Datenaustausches
Transport Layer:	Zuordnung der Datenpakete zu einer Anwendung
Network Layer:	Routing der Datenpakete zum nächsten Knoten
Data Link Layer:	Segmentierung der Pakete in Frames
Physical Layer:	Umwandlung Bits in Signal

7 – Mathematik der Algorithmen

7 – Mathematik der Algorithmen

Kapitel 7 – Mathematik der Algorithmen

- Aussagenlogik
- Prädikatenlogik
- Graphen als Datenstruktur
- O-Notation

7 – Mathematik der Algorithmen

Aussagenlogik

- Boolsche Ausdrücke bestehen aus boolschen Variablen, die entweder WAHR oder FALSCH annehmen können und mit Junktoren wie UND, ODER, NICHT und ENTWEDER-ODER verknüpft werden.
  Je nach Wert der Variablen ist ein Ausdruck, wenn man ihn auswertet, WAHR oder FASCH.

  z.B.

a	b	$a \wedge b \vee a$
true	true	true
true	false	true
false	true	false
true	true	true

7 – Mathematik der Algorithmen

Aussagenlogik

- Assoziativgesetz

$$(A \vee B) \vee C \Leftrightarrow A \vee (B \vee C)$$
$$(A \wedge B) \wedge C \Leftrightarrow A \wedge (B \wedge C)$$

- Kommutativgesetz

$$(A \vee B) \Leftrightarrow (B \vee A)$$
$$(A \wedge B) \Leftrightarrow (B \wedge A)$$

- Distributivgesetz

$$A \vee (B \wedge C) \Leftrightarrow (A \vee B) \wedge (A \vee C)$$
$$A \wedge (B \vee C) \Leftrightarrow (A \wedge B) \vee (A \wedge C)$$

- Absorptionsgesetz

$$A \wedge (A \vee B) \Leftrightarrow A$$
$$A \vee (A \wedge B) \Leftrightarrow A$$

- Idempotenzgesetz

$$A \wedge A \Leftrightarrow A$$
$$A \vee A \Leftrightarrow A$$

7 – Mathematik der Algorithmen

Aussagenlogik

- KV Diagramm zur Vereinfachung von logischen Ausdrücken

gegeben $f(A,B,C,D) = \overline{A}\,\overline{B}\,\overline{C}\,D + A\,B\,\overline{C}\,\overline{D} + \overline{A}\,C\,\overline{D} + \overline{A}\,\overline{C}\,\overline{D} + B\,\overline{C}\,\overline{D}$

	$\overline{C}\,\overline{D}$	$\overline{C}\,D$	$C\,D$	$C\,\overline{D}$
$\overline{A}\,\overline{B}$	1	0	0	1
$\overline{A}\,B$	1	0	0	1
$A\,B$	1	0	0	0
$A\,\overline{B}$	1	1	0	0

7 – Mathematik der Algorithmen

Aussagenlogik

- KV Diagramm zur Vereinfachung von logischen Ausdrücken

gegeben $f(A,B,C,D) = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + A\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,C\,\overline{D} + \overline{A}\,C\,D + \overline{B}\,\overline{C}\,\overline{D}$

	$\overline{C}\,\overline{D}$	$\overline{C}\,D$	$C\,D$	$C\,\overline{D}$
$\overline{A}\,\overline{B}$	1	0	0	1
$\overline{A}\,B$	1	0	0	1
$A\,B$	1	0	0	0
$A\,\overline{B}$	1	1	0	0

$= \overline{C}\,\overline{D} + \overline{A}\,\overline{D} + A\,\overline{B}\,\overline{C}$

7 – Mathematik der Algorithmen

Prädikatenlogik

- Die Prädikatenlogik ist eine Erweiterung der Aussagenlogik und untersucht zusätzlich Aussagen und Aussagenverbindungen.
  Mit Hilfe der Prädikatenlogik kann man Aussagen über eine unendliche Menge von Zahlen treffen.

  Dabei helfen Quantoren wie der
  - Allquantor (für alle x gilt): $\forall x$
  - Existenzquantor (es gibt mindestens ein x für das gilt): $\exists x$

z.B.	$(\forall n \in \mathbb{N})(J^n)^n = J^{n^2}$	allgemeine Umformung
	$(\forall n \in \mathbb{N}) n^2 \geq n$	Relationen, die für alle n erfüllt sind
	$(\exists x \in \mathbb{R}) x^2 \geq x$	Relationen, für die es mindestens eine Zahl gibt, die die Relation erfüllt.

7 – Mathematik der Algorithmen

Graphen

- Ein Graph besteht aus einer Menge von Knoten und Kanten.
  Man unterscheidet gerichtete und ungerichtete Graphen:

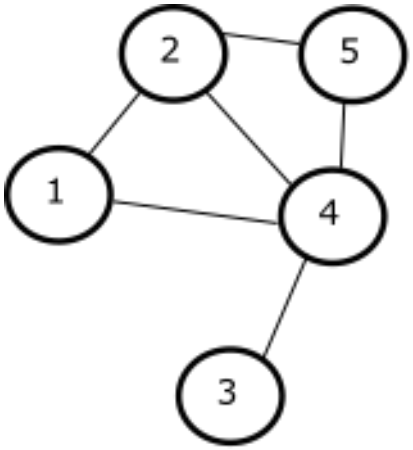


Abb.1: ungerichteter Graph

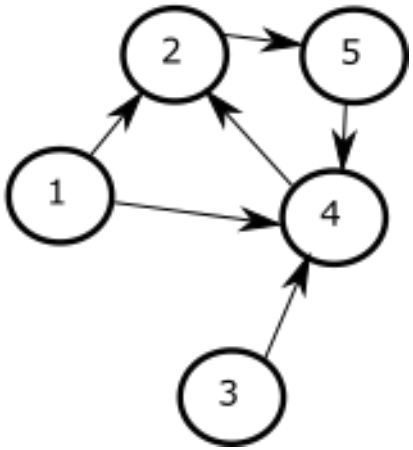


Abb.2: gerichteter Graph

7 – Mathematik der Algorithmen

Graphen

- Ein Graph, der mindestens einen Zyklus hat heisst zyklisch.
- Graphen ohne Zyklen werden azyklisch oder Wald genannt.
- Ein Zyklus (auch Kreis genannt) heiß trivial, wenn er weniger als 3 Knoten enthält.
- Als Taillenweite eines Graphen bezeichnet man die Länge eines kürzesten nichttrivialen Kreises[1]

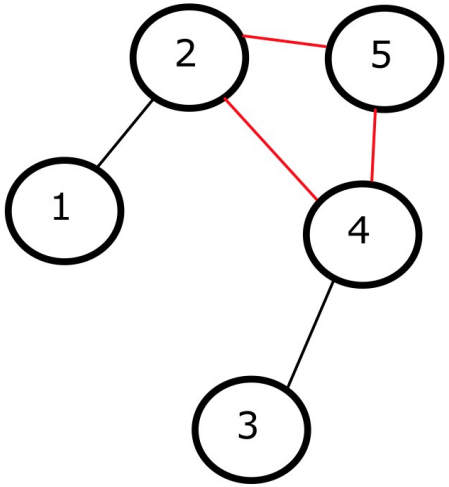


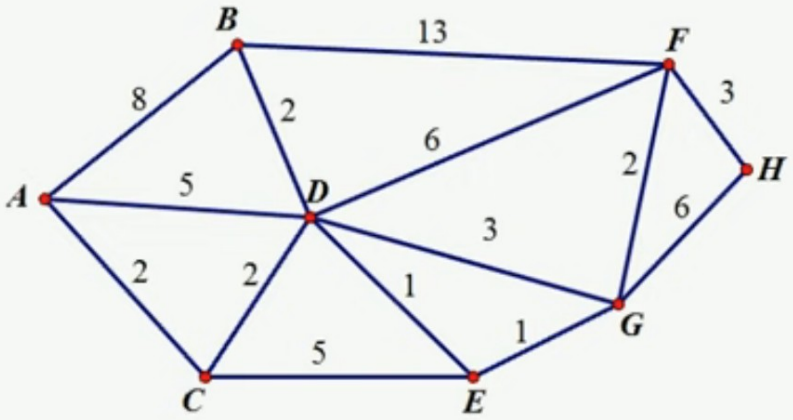
Abb.2: Zyklischer Graph mit Taillenweite 3

1 https://de.wikipedia.org/wiki/Graphentheorie.

7 – Mathematik der Algorithmen

Graphen

Beispiel: Dijkstra Algorithmus



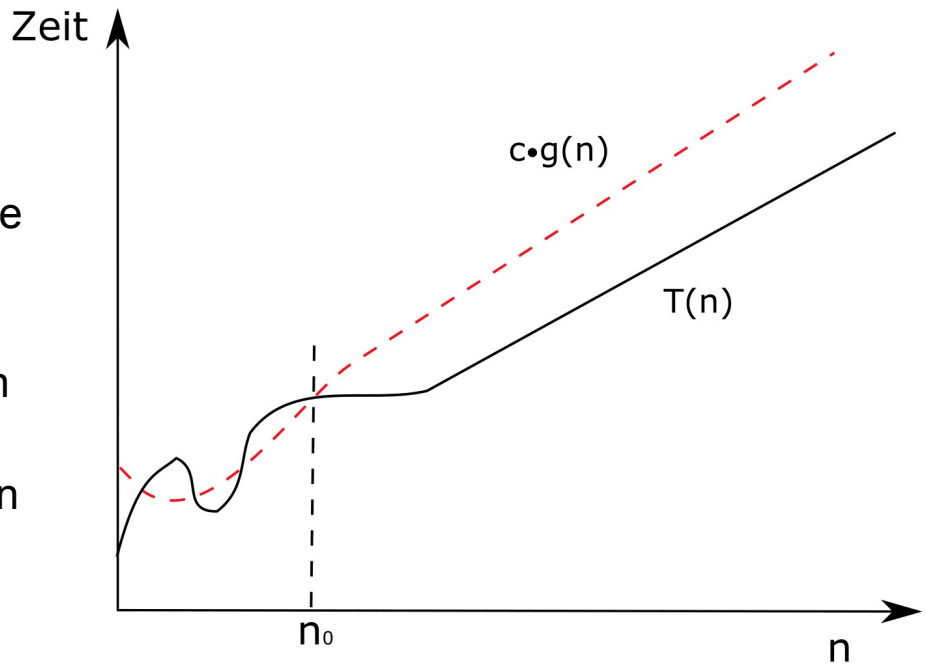
V	A	B	C	D	E	F	G	H
A	0_A	8_A	2_A	5_A	∞	∞	∞	∞
C		8_A	2_A	4_C	7_C	∞	∞	∞
D		6_D		4_C	5_D	10_D	7_D	∞
E		6_D			5_D	10_D	6_E	∞
B		6_D				10_D	6_E	∞
G						8_G	6_E	12_G
F						8_G		11_F
H								11_F

https://www.youtube.com/watch?v=5GT5hYzjNoo

7 – Mathematik der Algorithmen

Groß O-Notation

- Gibt ein Maß für die Anzahl der Elementarschritte in Abhängigkeit der Problemgröße an
- Dabei gibt n die Problemgröße an z.B. die Anzahl der zu sortierenden Elemente oder die Anzahl der Städte einer Rundreise
- T(n) ist die Anzahl der Schritte (Zeit), die der Algorithmus wirklich benötigt
- c•g(n) ist eine obere Schranke, die für alle $n>n_0$ gilt.
- Als Beispiel haben 2 ineinander geschachtelte Schleifen die jeweils von 0 bis n zählen einen Aufwandt von etwa $g(n)\sim n^2$
- Man schreibt dann, dass ein Programm einen Rechenaufwandt von O(g(n)) hat.
  (Im Beispiel mit den verschachtelten Schleifen bedeutet das ein Aufwand von $O(n^2)$)



8 – Python

Kapitel 8 – Python

- Einführung
- Variablen und Ausdrücke
- Listen, Tupel, Dictionaries
- Bedingte Verzweigungen (Schleifen, if-Abfragen)
- Funktionen
- Klassen, Vererbung
- Bibliotheken

8 – Python

Einführung

- Python ist eine Programmiersprache, die vor allem in wissenschaftlichen Bereichen, im Speziellen im Maschinellen Lernen, enorm an Zuwachs gewonnen hat.
- Auch im Bereich der Systemadministration ist Python stark.
- Python ist verhältnismäßig einfach zu programmieren und gut lesbar.
- Der Code ist recht kompakt und die Bibliotheken, die meist in optimiertem C/C++ Code geschrieben sind, sind meist schnell und oftmals Umfangreich.
- Python ermöglicht nicht nur prozedurale und objektorientierte Programmierung, sondern beinhaltet mit Lambdas auch Ansätze zur funktionalen Programmierung

8 – Python

Variablen und Ausdrücke

- Python Variablen sind zunächst typenlos. Seit Python 3.6 können Variablen aber auch typisiert werden.
- Variablen müssen iniialisiert werden, bevor sie in einem Ausdruck ausgewertet werden.
- intern gibt es die Datentypen: int (ganze Zahlen), float (Fließkommazahlen), (komplexe Zahlen) complex, (boolscher Wert) bool, (Zeichenkette) str, tupel, list, sets, dict und bytearray
- Python entscheidet bei der Zuweisung, welcher Datentyp genommen wird.
- Mit type(var) oder isinstance(var, typ) kann man den Datentyp ermitteln.
- Ähnlich wie bei Java werden bei einfachen Datentypen (int, float, complex, bool, str, tupel) die Inhalte bei Zuweisung kopiert (immutable), während bei komplexeren Datentypen das zugewiesene Objekt nur referenziert wird (mutable). Will man hier auch die Inhalte kopieren, so kann man die copy Funktion nutzen!

8 – Python

Variablen und Ausdrücke

Beispiele:

```
s = 'Dies ist ein'

i = 3

f = 3.1415927

s += ' String!'      #  mit + kann man Strings konkatenieren

i = f//2             #  // heisst Ganzzahldivision

f = f ** 2           #  ** 2 heisst hoch 2
```

8 – Python

Listen

Eine Liste ist besteht aus einer Menge von Elementen beliebigen Datentyps

```
lst = [3, 9, 5, 20, 7]

lst2 = ["Ted", "Fred", "Eni", "Joe", "Nick"]

print(lst2[0])    # ergibt "Ted"

print(lst2[-1])   # ergibt "Nick" (-1 bedeutet das letzte Element)

lst2[0:3]         # ergibt "Ted", "Fred", "Eni"

del lst2[1]       # löscht "Fred"

lst3 = lst + lst2 # hängt die Listen aneinander

lst.sort()        # sortiert die Liste!

lst.insert(105,1) # fügt die 105 an Stelle 1 in der Liste hinzu
```

8 – Python

Listen

Ein weiteres Beispiel:

```
name      = input("Wie heisst du? ")
list_name = list(name)
age       = int(input("Wie alt bist du? "))
list_age  = [age]
user      = list_name + list_age
```

8 – Python

Tupel

Während man Listen verändern kann, bleiben Tupel in Python konstant.

```
day = ("Montag", "Dienstag", "Mittwoch","Donnerstag", "Freitag",
       "Samstag", "Sonntag")

print(day[1])  # gibt "Dienstag" aus
```

So können auch Listen von gruppierten Tupeln erstellt werden:

```
birthdays=[("Mozart", 1756),("Bach", 1685), (Haendel, 1685)]

print(birthday[0][1]) # ergibt 1756 denn es ist das Nullte
                      # Listenelement und davon der 2. Eintrag
```

8 – Python

Dictionaries

Dictionaries sind Praktisch, da die Daten mit Schlüsseln assoziiert werden: **Schlüssel:Wert**

```
phonebook={"Norah":"0176-2242622", "Emma":"0161-552668", "Eric":"0176-246732"}
print(phonebook[Eric])  # gibt 0176-246732 aus
```

Es können einfach Einträge hinzugefügt werden:

```
phonebook["Philip"] = "0176-3342622"
```

Genauso einfach ist es Einträge zu löschen:

```
del phonebook["Philip"]
```

8 – Python

Bedingte Verzweigungen

- Eine Eigenart von Python ist, dass man Programmblöcke von Befehlen als Block kennzeichnet, indem man alle darin enthaltenen Befehle gleich weit einrückt.

Beispiele für bedingte Verzweigungen:

```
if a>b:
    print(a)
else:
    print(b)
```

Die for-Schleife:

```
for i in range(start, end, stepsize):
    print(i)
```

die While Schleife:

```
while i<j:
    print(i,j)
```

8 – Python

Funktionen

- Funktionen werden in Python mit def vor einem Funktionsnamen definiert:

```
def my_function():
     print("Jetzt bin ich in der Funktion my_function")
```

- Hinter dem Funktionsnamen kommen Klammern, in den Parameter übergeben werden:

```
def square(x):
    return x*x
```

- Der Block innerhalb der Funktion wird durch einrücken kenntlich gemacht:

```
def sum_from_1_to(x):
    sum = 0
    for i in range(1,x+1):
        sum += i
    return sum
```

8 – Python

Klassen, Vererbung

- Wie in Java, C++ und anderen Programmiersprachen gibt es in Python auch Klassen

```
class Konto:
    def __init__(self, betrag):
        Kontostand = betrag;
    def abheben(self):
        print("Abheben")
    def einzahlen(self):
        print("Einzahlen")

class Girokonto(Konto):    # erbt von Konto
    def abheben(self):
    print("Abheben vom Girokonto")

meinKonto = Girokonto(100) # girokonto erstellen
meinKonto.abheben()        # überschriebene Funktion aufrufen
meinKonto.einzahlen()      # Funktion von Konto aufrufen
```

--- Ergebnis ---

```
Abheben vom Girokonto
Einzahlen
```

8 – Python

Bibliotheken

- NumPy ist eine Mathematische Bibliothek, die es erlaubt mit Elementen aus der linearen Algebra, wie z.B. Vektoren und Matritzen, zu arbeiten.
- Matplotlib ist eine Bibliothek, die unterschiedliche Visualisierungen seiner Daten ermöglicht
- PyGame erlaubt das Spieleprogrammieren. Man kann es auch nutzen um Life Grafiken zu erzeugen, die sich im Laufe der Zeit ändern.
- PyQT ist für grafische Benutzeroberflächen und Grafische Ausgaben nützlich
- SciPy ist eine mathematische Bibliothek
- Tensorflow, Keras, Pytorch sind Bibliotheken zur Implementierung von Neuronalen Netzen. Meist ist es möglich diese auf Grafikkarten um ein Vielfaches zu beschleunigen.

… u.v.a.m.