Basisinformationstechnologie II

Sommersemester 2023. Zusammenfassung I.

Basierend auf Jan Wieners' Folien

Institut für Digital Humanities, Historisch-Kulturwissenschaftliche Informationsverarbeitung | Prof. Dr. Øyvind Eide | Slavina Stoyanova

Seminarthemen BIT I

Block I: Grundlagen I - Grundlagen IV

- Informatik, Information und Daten, Zahlendarstellungen, Informationsdarstellung,
  Umwandlung / Rechnen im Binärsystem

Block II: Rechnertechnologie

- Von Neumann Architektur, Rechnerkomponenten: Hardware, Boolesche Algebra, (Transistor)Schaltungen, Speicherbausteine

Block III: Betriebssysteme

- Verknüpfung Hard- und Software, Aufgaben von Betriebssystemen, Prozesse, Multitasking, Speicher- und Dateiverwaltung

Block IV: Programmiersprachen

- Arten von Programmiersprachen, VMs, Interpreter, Compiler, Programmentwicklung, UML, Datentypen, Variablen, Kontrollstrukturen

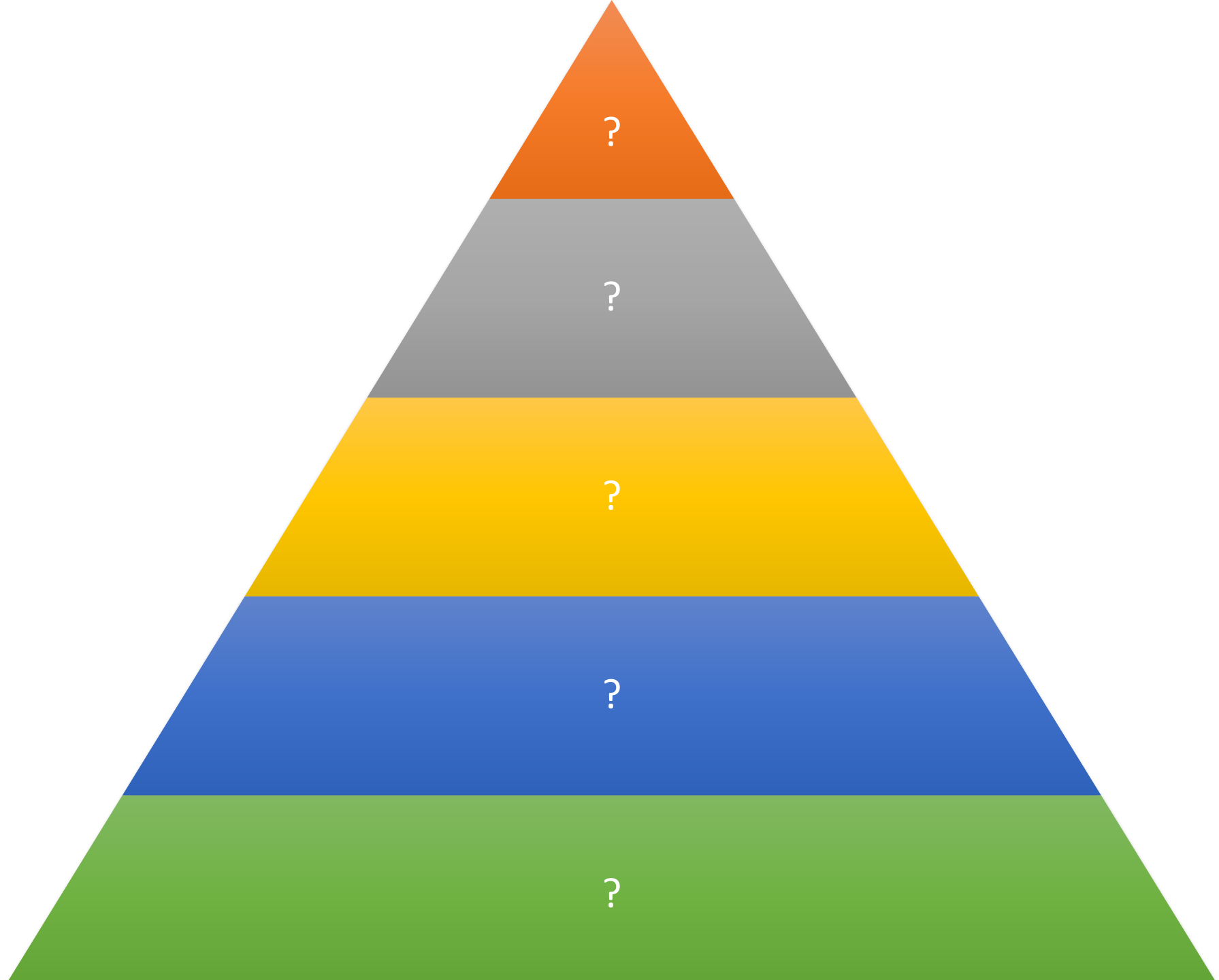
Rechnertechnologie I

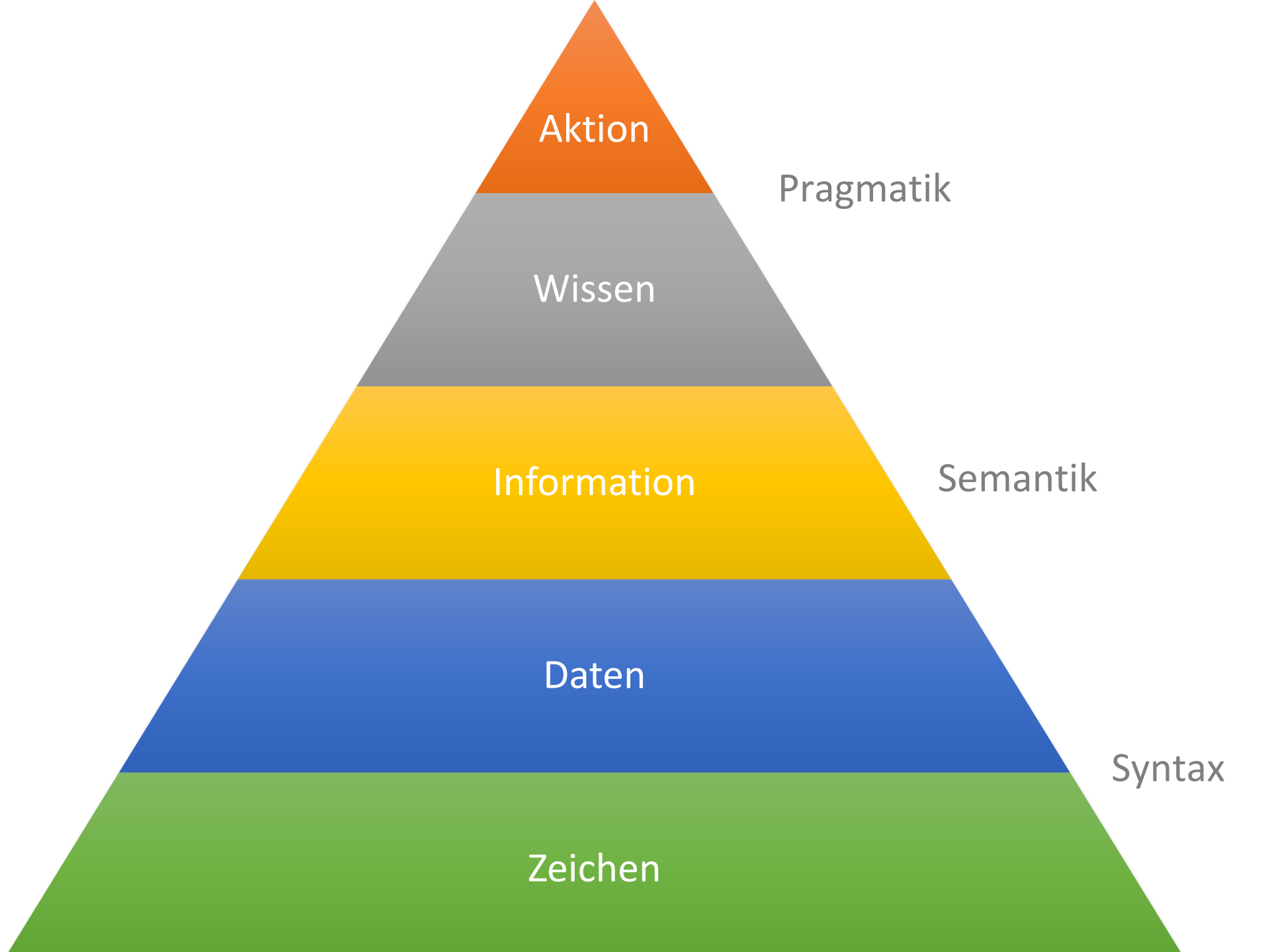
Überblick: Rechner-/Computerentwicklung

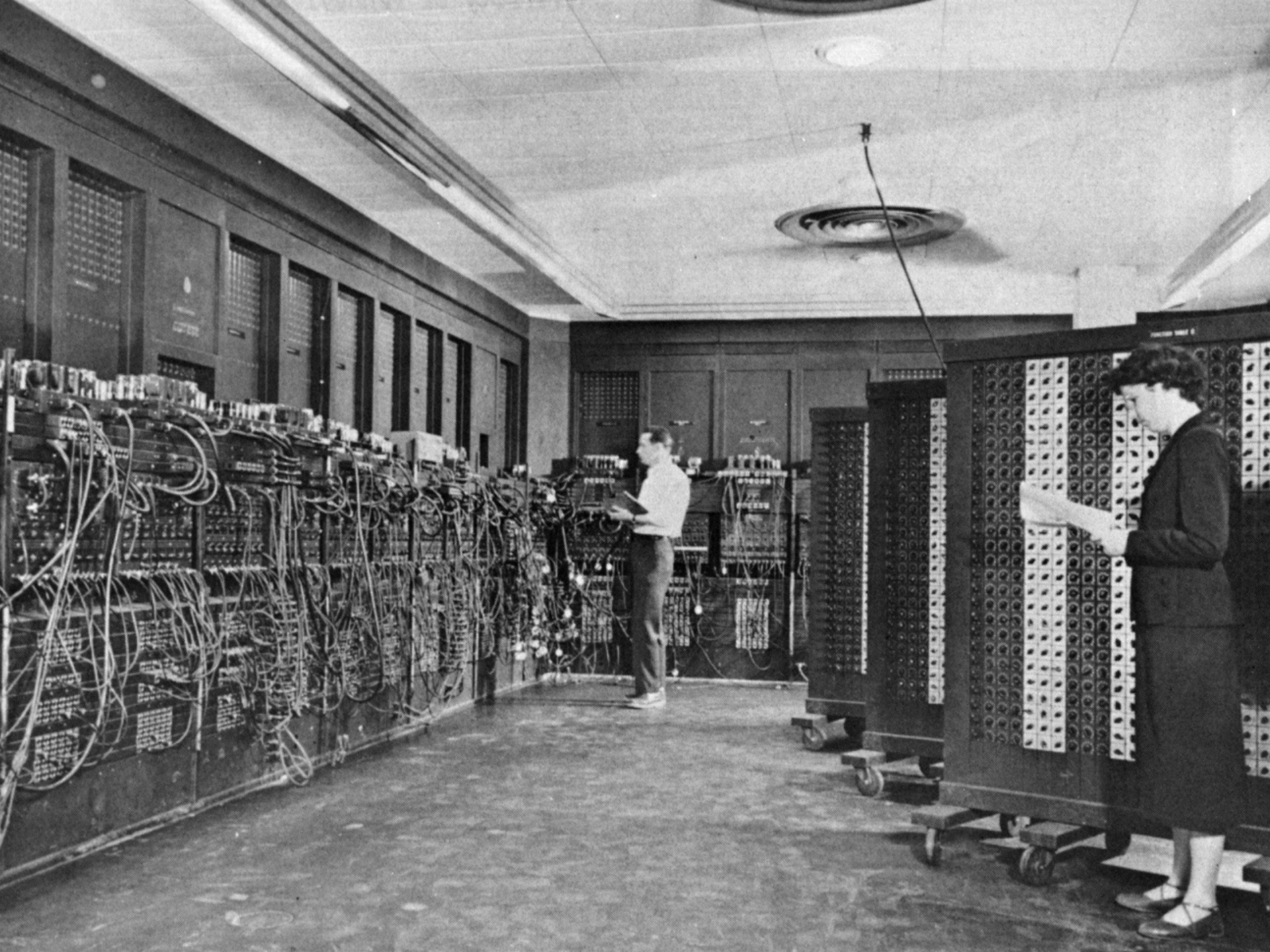
- Moore
- Leibniz
- Babbage
- Turing
  - Exkurs: Turingtest
- Weizenbaum
- von Neumann
  - Exkurs: Spieltheorie
- Die von Neumann Rechnerarchitektur
- Konzept: Universalrechner
- Cache als Hardwareelement
- Caching als Grundmechanismus

Zeitgemäße Rechnerhardware

- Motherboard, etc.




Aktion
Wissen
Information
Daten
Zeichen
Pragmatik
Semantik
Syntax



…und die Klausur?

Grundlagen I

- Grundbegriffe
  - Symbole, Semiotik !
  - Daten !
  - Informationen !

  → Wissenspyramide !
- Codierung !
- Fachdisziplinen der Informatik
  - Technische Informatik
  - Praktische Informatik
  - Theoretische Informatik
  - Angewandte Informatik
- Codierung von Information: der Binärcode !

ប្រាសាទអង្គរវត្ត

(ត្រូវបានបញ្ជូនបន្តពី Angkor Wat)

សម្រាប់ក្រុមភ្លេងរលោហៈវាយគោៈអារមរិក សូមមើល អង្គរវត្ត (ក្រុមភ្លេង) ។

អង្គរវត្ត ស្ថិតនៅក្នុងខេត្តសៀមរាប មានទីតាំងស្ថិតនៅភាគខាងជើងនៃក្រុងសៀមរាប ក្នុងស្រុកសៀមរាប មានចម្ងាយ ៧ គ.ម ពីទីរួមខេត្ត តាមផ្លូវកូម៉ែ ឬផ្លូវសាលដី ហោលៗ។ វាជាប្រជុំអគារប្រាសាទព្រហ្មញ្ញសាសនាធំបំផុតនិងវិមានសាសនាដ៏ធំបំផុតនៅក្នុងលោក។ ប្រាសាទត្រូវបានកសាងឡើងដោយសូរ្យវរ្ម័នទី២ ដែលជាស្នាដៃដ៏ធំចម្លងអស្ចារ្យនិងមានឈ្មោះល្បីល្បាញន្ទឹសុះសាយ ទៅគ្រប់ទិសទីលើពិភពលោក។ សង់ឡើងនៅដើមសតវត្សទី១២នៅយសោធរបុរៈ (សម័យបច្ចុប្បន្ន អង្គរ) អធិរាជាណាចក្រខ្មែរ ជាប្រាសាទតំណាងនិងព្រះសុសានចុងក្រោយរបស់ព្រះអង្គ។ ដែលងាកចេញពីប្រពៃណីសិវនិយមនៃព្រះរាជាមុនៗ អង្គរវត្តដែលជំនួសមកវិញ ឧទ្ទិសដល់ព្រះវិស្ណុ។ ជាប្រាសាទដែលនៅគង់វង្សល្អបំផុតនៅស្ថានីយនេះ វាគឺជាប្រាសាទតែមួយគត់ដែលបានបន្សល់ទុកនូវមជ្ឈមណ្ឌលសាសនាដ៏សំខាន់មួយចាប់តាំងពីមូលដ្ឋានគ្រឹះរបស់ខ្លួន – ដំបូងព្រហ្មញ្ញសាសនាបូជាដល់ព្រះវិស្ណុ ក្រោយមកពុទ្ធនិយម។ ប្រាសាទនេះស្ថិតនៅក្នុងរចនាបថបុរាណជាន់ខ្ពស់កំពូលនៃស្ថាបត្យកម្មខ្មែរ។ វាបានក្លាយជានិមិត្តរូបកម្ពុជា[១] ដែលរំលេចលើទង់ជាតិរបស់ប្រទេស ជារូបភាពតំណាងប្រទេសជាតិខ្មែរ និងវាក៏ជាការទាក់ទាញពួកអ្នកទេសចរណ៍សំខាន់របស់ប្រទេសនេះផងដែរ។

អង្គរវត្តរួមផ្សំនូវគ្រោងការគ្រឹះពីរនៃស្ថាបត្យកម្មប្រាសាទខ្មែរៈ ប្រាសាទភ្នំ និងប្រាសាទដែលមានថែវចុងក្រោយគេ ដោយផ្អែកលើស្ថាបត្យកម្មទ្រាវដើមដំបូង រួមមានលក្ខណៈពិសេសដូចជាជគតិ។ ប្រាសាទត្រូវបានរចនាដើម្បីតំណាងភ្នំព្រះសុមេរុ លំនៅនៃពួកទេវៈក្នុងទេវកថាព្រហ្មញ្ញៈ ក្នុងកសិណ រីគូទឹកព័ទ្ធជុំ វិញប្រហែលជា ១៦គ.ម ចំណែកកសិណពិតប្រាកដរបស់ប្រាសាទ មានបណ្តោយប្រវែង១៥០០ម x ទទឹង១៣០០ម លើ២០០ម ផ្លូវធំដើរចូលពីទិសខាងលិចទៅដល់ប្រាសាទកណ្តាលមានប្រវែង២៥០ម ឯកំពូលកណ្តាលខ្ពស់ធំជាងគេបំផុតរបស់នគរវត្ត មានកម្ពស់ ៦៥ម និងជញ្ជាំងខាងក្រៅ៣,៦ សហគមាត្រ (២,២ ម៉ៃ.)វែងគឺជាថែវមានបីជ្រុង ដែលថែវនីមួយៗឡើងខ្ពស់ទៅថែវដែលជាប់គ្នា។ នៅចំកណ្តាលនៃប្រាសាទបញ្ឈរនូវជួរចតុរង្គនៃប៉ម។ មិនដូចប្រាសាទនៅអង្គរភាគច្រើនទេ អង្គរវត្តបែរមុខទៅទិសខាងលិច ពួកអ្នកប្រាជ្ញជែកញែកថាជាចំណុចសំខាន់នៃប្រាសាទនេះ។ ប្រាសាទនេះត្រូវបានគេកោតសរសើរដោយសារភាពសម្បើមនិងរុងរឿងនៃស្ថាបត្យកម្មនេះ ចម្លាក់លៀនលឹមៗដែលលាតសន្ធឹង និងដោយសារពពួកទេវតាដែលតាក់តែងលើជញ្ជាំងប្រាសាទ។

ឈ្មោះសម័យថ្មី អង្គរវត្តមានន័យថា *ទីក្រុងវត្ត* ក្នុងភាសាខ្មែរ *អង្គរ*មានន័យថា *ទីក្រុងរី ទីក្រុងធានី* ជាទម្រង់ប្រាក្រឹតនៃពាក្យ *នគរ* ដែលមកពីពាក្យសំស្ក្រឹត *nagara* (नगर)។[២] *វត្ត*គឺជាពាក្យខ្មែរសម្រាប់ *ទេវាល័យលើដី* បានមកពីពាក្យបាលី "vatta" (वत्त)។[៣] មុនពេលនេះ ប្រាសាទនេះត្រូវបានគេស្គាល់ថាជា *បរមវិសលោក* (Parama Vishnuloka ជាភាសាសំស្ក្រឹត) បន្ទាប់ពីងារបចាមរណៈនៃស្ថាបនិកនៃប្រាសាទ។[៤]

ទំព័រដើម
បន្លាស់ប្តូរថ្មីៗ
ជំនួយ
អត្ថបទថ្មីៗ
ទំព័រចៃដន្យ

សហគមន៍
សុំធ្វើជាអ្នកអភិបាល
សុំប្តូរអត្តនាម
ផតថលសហគមន៍
ព្រឹត្តិការណ៍ថ្មីៗ

បោះពុម្ព/នាំចេញ

ប្រអប់ឧបករណ៍

ជាភាសាដទៃទៀត

Aragonés
العربية
Azərbaycanca
Беларуская
Беларуская (тарашкевіца)
Български
বাংলা
Bosanski
Català
Нохчийн
Česky
Cymraeg

Normenfamilie ISO 8859

ISO 8859-1	Latin-1, Westeuropäisch
ISO 8859-2	Latin-2, Mitteleuropäisch
ISO 8859-3	Latin-3, Südeuropäisch
ISO 8859-4	Latin-4, Baltisch
ISO 8859-5	Kyrillisch
ISO 8859-6	Arabisch
ISO 8859-7	Griechisch
ISO 8859-8	Hebräisch
ISO 8859-9	Latin-5, Türkisch
…	…
ISO 8859-16	Latin-10, Südosteuropäisch

Und noch ein Standard: UTF-8

- UTF-8: In den 1990ern eingeführt von der ISO
- UTF → Implementierung von Unicode
- UTF-8 ist eine Mehrbyte-Codierung. Das bedeutet:
  - Dass 7-Bit ASCII-Zeichen mit einem Byte codiert werden, alle anderen verwenden zwischen 2 und 6 Bytes

Die Idee:

- Häufig benutzte Zeichen werden mit einem Byte codiert, seltenere mit mehreren Bytes – das spart Speicherplatz.
- UTF-8 codierte Dateien sind kompatibel zu 7-Bit ASCII

101

„Mit der Symbolfolge 101 ist die Dezimalzahl Einhunderteins gemeint“

„Mit der Symbolfolge 101 ist die Binärzahl 101 gemeint. Der Binärzahl 101 entspricht die Dezimalzahl 5 (Fünf)“

Vier Zahlensysteme gegenübergestellt

Dezimal	0	1	2	3	4	5	6	7	8
Binär	0	1	10	11	100	101	110	111	1000
Oktal	0	1	2	3	4	5	6	7	10
Hexadezimal	0	1	2	3	4	5	6	7	8

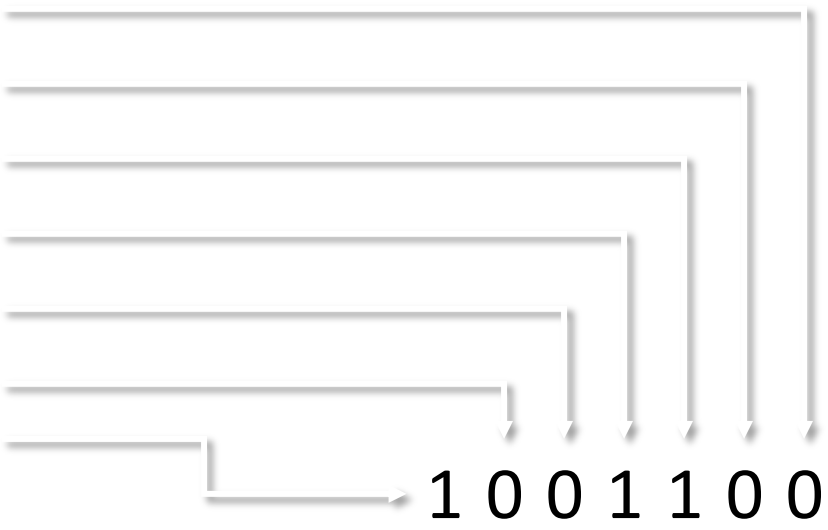
Dezimal	9	10	11	12	13	14	15	16
Binär	1001	1010	1011	1100	1101	1110	1111	10000
Oktal	11	12	13	14	15	16	17	20
Hexadezimal	9	A	B	C	D	E	F	10

Umwandlung Dezimal→Binärsystem

Eine Dezimalzahl lässt sich über die Division durch 2 und Aufschreiben der Reste in eine Binärzahl umwandeln (das ist eine Möglichkeit, häufig lässt sich das auch im Kopf lösen).

Beispiel: Die Zahl 76_{10} soll ins Binärsystem umgewandelt werden

- 76 / 2 = 38; Rest 0
- 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



…und die Klausur?

Grundlagen II

Zeichencodierungen

- ASCII / Extended ASCII !
- ISO 8859-x
- UTF-8 !

Zahlensysteme

- Hexadezimal-
- Dezimal- !
- Binärsystem !

Umwandlung vom

- Dezimal- ins Hexadezimalsystem
- Hexadezimal- ins Dezimalsystem
- Dezimal- ins Binärsystem !
- Binär- ins Dezimalsystem !

Rechnen im Binärsystem: Addition

Additionsregeln

- 0 + 0 = 0
- 0 + 1 = 1
- 1 + 0 = 1
- 1 + 1 = 0 mit 1 Übertrag → 1 0

- 1 + 1 + Übertrag = 1 + Übertrag
- 1 + 1 + Übertrag + Übertrag = 1+1+1+1 (4x1)

Eine Beispielaufgabe

```
  0101 1011
+ 0000 1101

Überträge
        1 1  1 1 1

---------------

   110 1000
```

Beispielaufgabe Multiplikation

4*7 = ?

- 100*111
  100
  100
  100

  11100=28

Binärdarstellung ganzer Zahlen

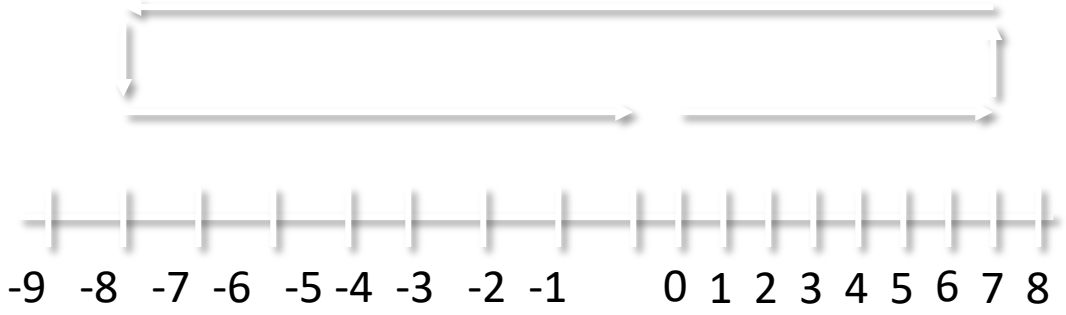
Wie $\mathbb{Z}$ darstellen?

Möglichkeit I:
MSB (Most Significant Bit, d.h.: erstes Bit, ganz links) zur Kennzeichnung verwenden
→ MSB == 0, dann positive Zahl
→ MSB == 1, dann negative Zahl

…Probleme?

0000 = +0	1000 = -0
0001 = +1	1001 = -1
0010 = +2	1010 = -2
0011 = +3	1011 = -3
0100 = +4	1100 = -4
0101 = +5	1101 = -5
0110 = +6	1110 = -6
0111 = +7	1111 = -7

Zweierkomplementdarstellung



1000 = -8	1100 = -4	0000 = 0	0100 = 4
1001 = -7	1101 = -3	0001 = 1	0101 = 5
1010 = -6	1110 = -2	0010 = 2	0110 = 6
1011 = -5	1111 = -1	0011 = 3	0111 = 7

Darstellbarer Zahlenbereich: -2^{n-1} bis $2^{n-1}-1$

Zweierkomplement: Umrechnung

Umwandlung 6 in -6:

- Schritt 0: Binärdarstellung bilden: 0110
- Schritt I: Einerkomplement bilden, d.h. Negation aller Bits
  0110 → 1001
- Schritt II: Addition von 1
  1001 + 0001 = 1010

1010 ist die Entsprechung der Dezimalzahl -6 im Binärsystem (unter Verwendung der Zweierkomplementdarstellung)

1000 = -8	1100 = -4	0000 = 0	0100 = 4
1001 = -7	1101 = -3	0001 = 1	0101 = 5
1010 = -6	1110 = -2	0010 = 2	0110 = 6
1011 = -5	1111 = -1	0011 = 3	0111 = 7

Subtraktion: Addition von Zweierkomplementzahlen

Subtraktion = Addition der zu subtrahierenden Zahl:

5 - 7 = 5 +(- 7)

Im Binärsystem:

```
  0101  → 5
+ 1001  → Zweierkomplementdarst. von 7
  ------
  1110  → Führendes Bit == 1: Negative Zahl
```

Da negative Zahl: Wieder umwandeln, um Betrag zu bestimmen:

Einerkomplement von 1110 = 0001

Zweierkomplement von 1110 = 0010 = (-)2

…und die Klausur?

Grundlagen III

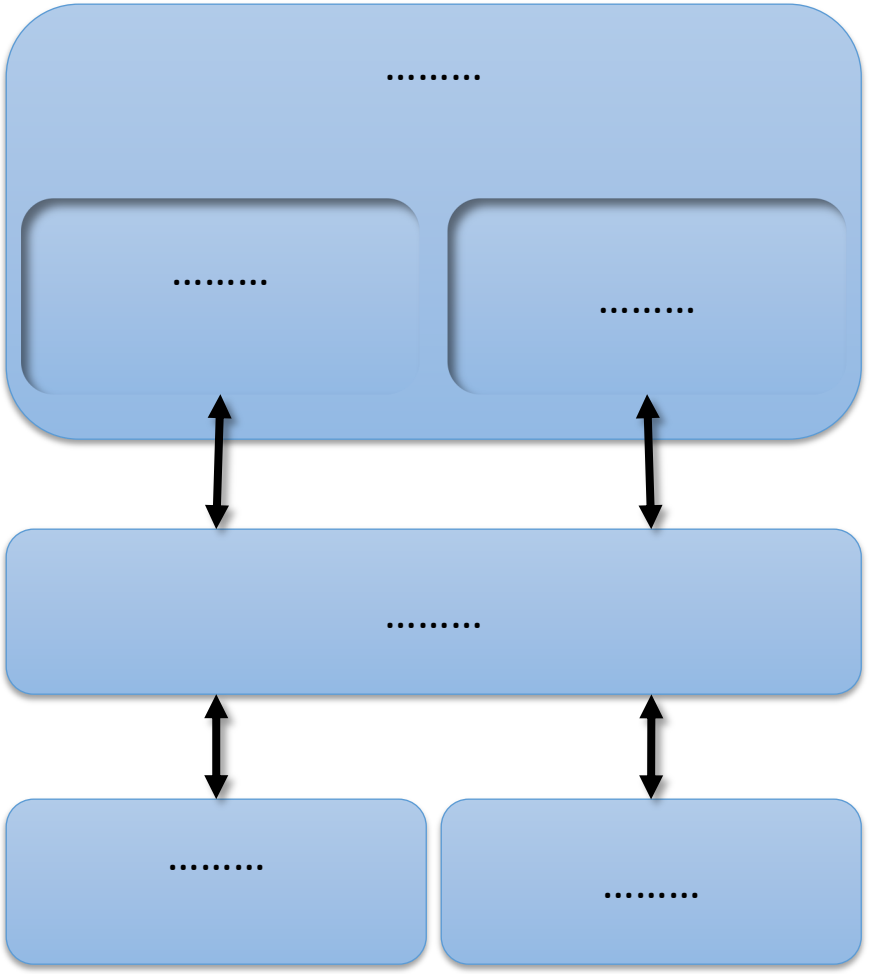
Vorzeichenlose Zahlen !

Binärsystem: Darstellung ganzer Zahlen, d.h. negativer Zahlen → Zweierkomplementdarstellung !

Rechnen im Binärsystem

- Addition !
- Subtraktion (Addition der negativen Zahl) !
- Multiplikation !
- ~~Division~~

Von-Neumann-Architektur



Rechnertechnologie I

Überblick: Rechner-/Computerentwicklung

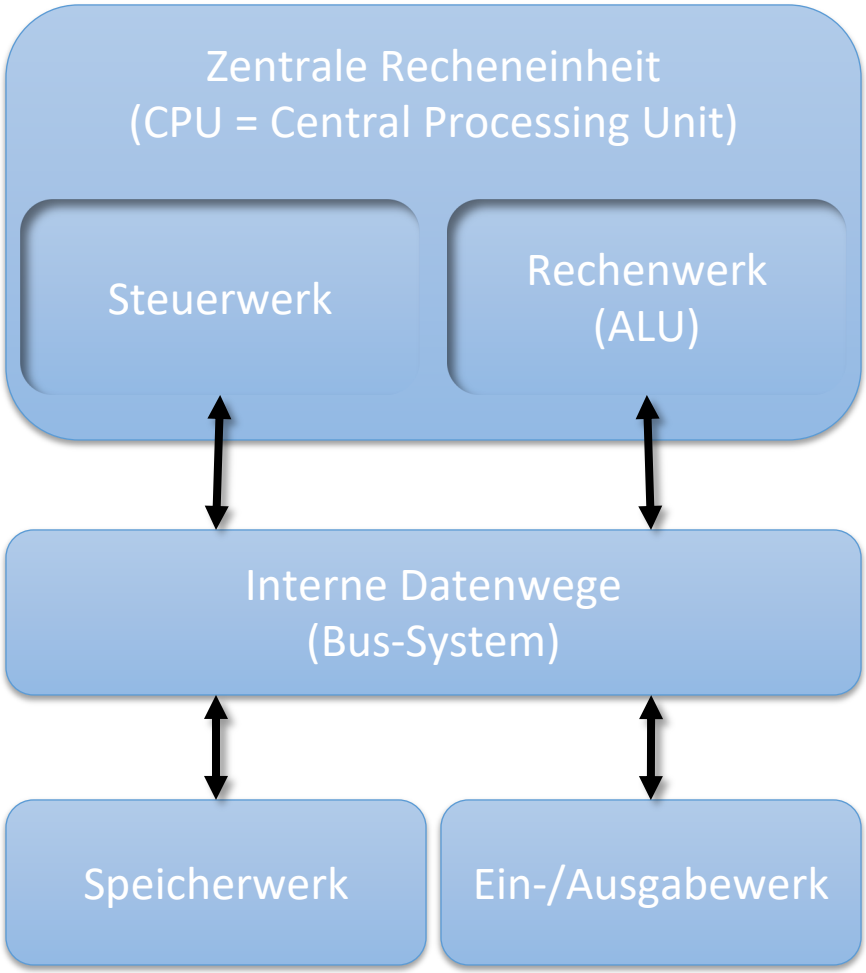
- Moore
- Leibniz
- Babbage
- Turing
  - Exkurs: Turingtest !
- Weizenbaum
- von Neumann
  - Exkurs: Spieltheorie
- Die von Neumann Rechnerarchitektur !
- Konzept: Universalrechner !
- von-Neumann-Flaschenhals **!**
- von-Neumann-Zyklus **!**
- Cache als Hardwareelement !
- Caching als Grundmechanismus !

Zeitgemäße Rechnerhardware

- Motherboard, etc.

Von-Neumann-Architektur

- **Steuerwerk**
- **Rechenwerk**
- **Interne Datenwege**
- **Arbeitsspeicher / Speicherwerk**
- **Ein-/Ausgabewerk**



Funktionsweise & Eigenschaften

- Zahlen werden im Rechner **binär** dargestellt

→ **Universalrechner**

- Programme und Daten werden in einem **gemeinsamen Speicher** abgelegt
- Befehle geben nur die **Speicheradresse** an, wo die Daten abgelegt sind, nicht die Daten selbst

Von-Neumann-Architektur

Befehlsverarbeitung → **Von-Neumann-Zyklus** in fünf Teilschritten:

- FETCH
- DECODE
- FETCH OPERANDS
- EXECUTE
- UPDATE PROGRAM COUNTER (UPC)

- **FETCH**: Laden des nächsten zu bearbeitenden Befehls in das **Befehlsregister** (bildet gemeinsam mit Steuerwerk und Rechenwerk die CPU).
- **DECODE**: Befehl wird durch Steuerwerk in Schaltinstruktionen für das Rechenwerk übersetzt.
- **FETCH OPERANDS**: Operanden holen, die durch den Befehl verändert werden sollen.
- **EXECUTE**: Rechenwerk führt die Operation aus.
- **UPC**: Erhöhung des Befehlszählers, damit der Rechner weiß, an welcher Stelle des Programms er sich gerade befindet. Geschieht parallel zu DECODE und FETCH OPERANDS.

Von-Neumann-Architektur

Vorteile

- **Bedeutende Idee**: Zunächst Laden des Programmes und der Daten in ein und denselben **Speicher**, danach Ausführung.
  - Vor von Neumanns Ansatz war das Programm hardwareseitig verschaltet / repräsentiert oder wurde über Lochstreifenkarten schrittweise eingelesen und sofort (sequentiell) verarbeitet.
- Nun möglich:
  - **Sprünge** auf vorhergehende und spätere Programmsequenzen
  - Modifikation des Programmcodes während des Programmablaufes

→ **Paradigmawechsel**: Übergang vom starren Programmablauf zur flexiblen Programmsteuerung bzw. von der Rechenmaschine zur Datenverarbeitungsmaschine

Von-Neumann-Architektur

Nachteile

- Da **Daten und Befehle** im **Speicher** gehalten werden, wird die Verbindung und Datenübertragung zwischen CPU und Speicher über den Systembus zum **Von-Neumann-Flaschenhals**:
  - Jeglicher Datenverkehr von und zur CPU wird über den internen Bus abgewickelt, dessen Transfergeschwindigkeit langsamer ist, als die Verarbeitungsgeschwindigkeit der CPU.
    Dieses Problem versucht man in modernen PC's durch die Verwendung von schnellem **Cache-Speicher** abzuschwächen, der meist in die CPU integriert ist.

→ Cache-Hit? Cache-Miss?

…und die Klausur?

Seminarthemen BIT I

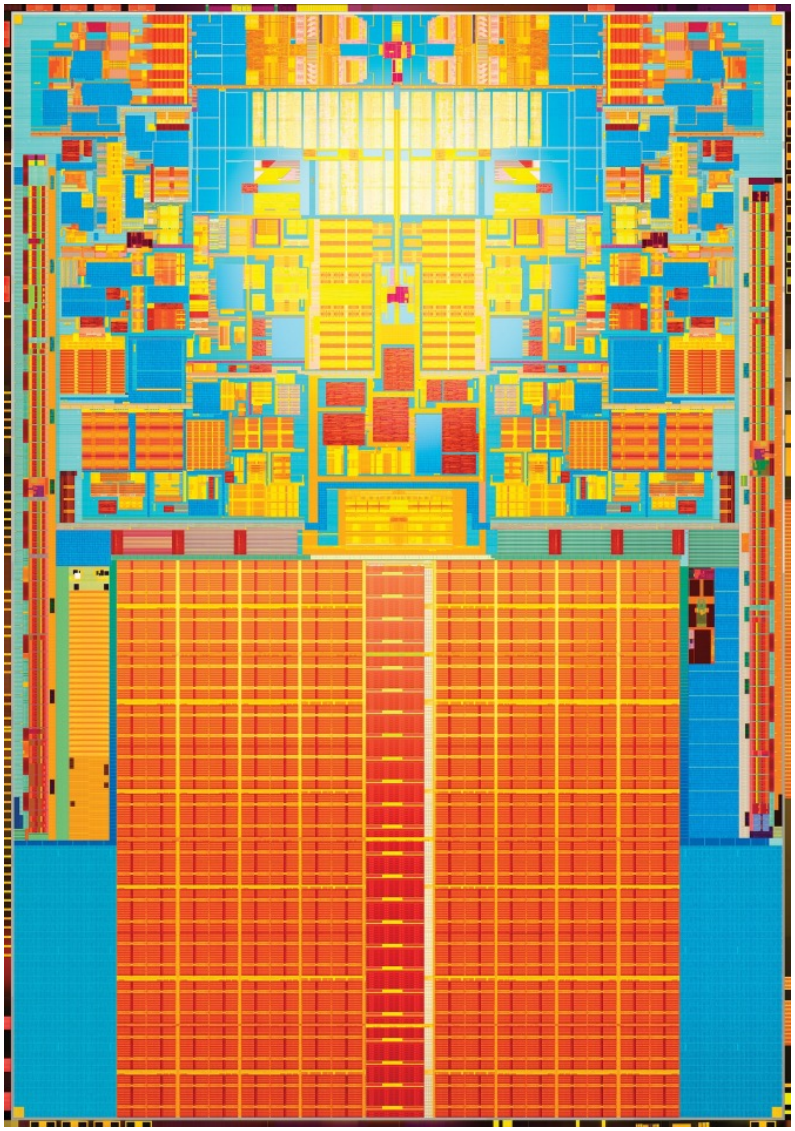
Block III: Betriebssysteme

- Verknüpfung Hard- und Software, Aufgaben von Betriebssystemen, Prozesse, Multitasking, Speicher- und Dateiverwaltung

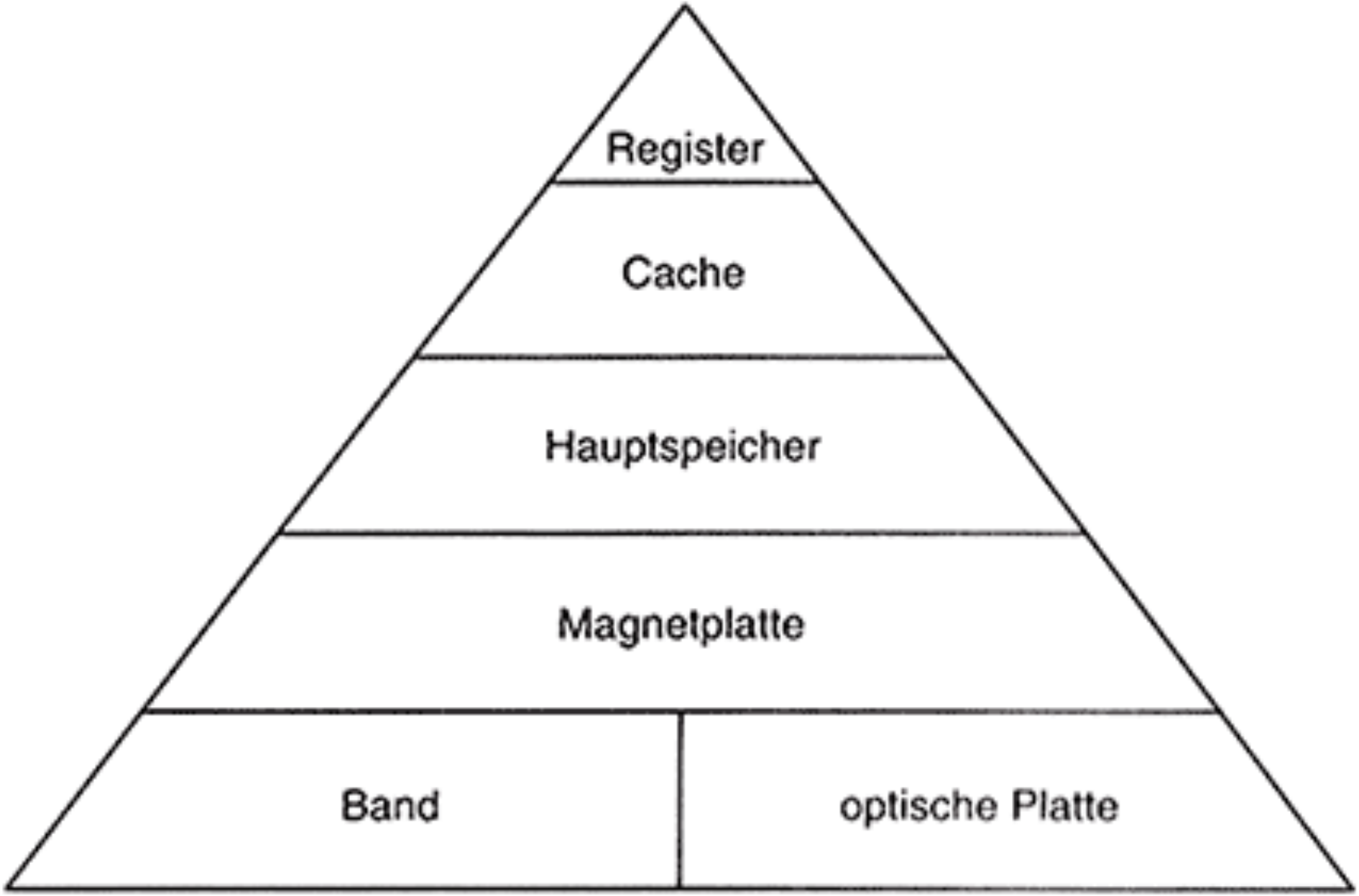
Block IV: Programmiersprachen

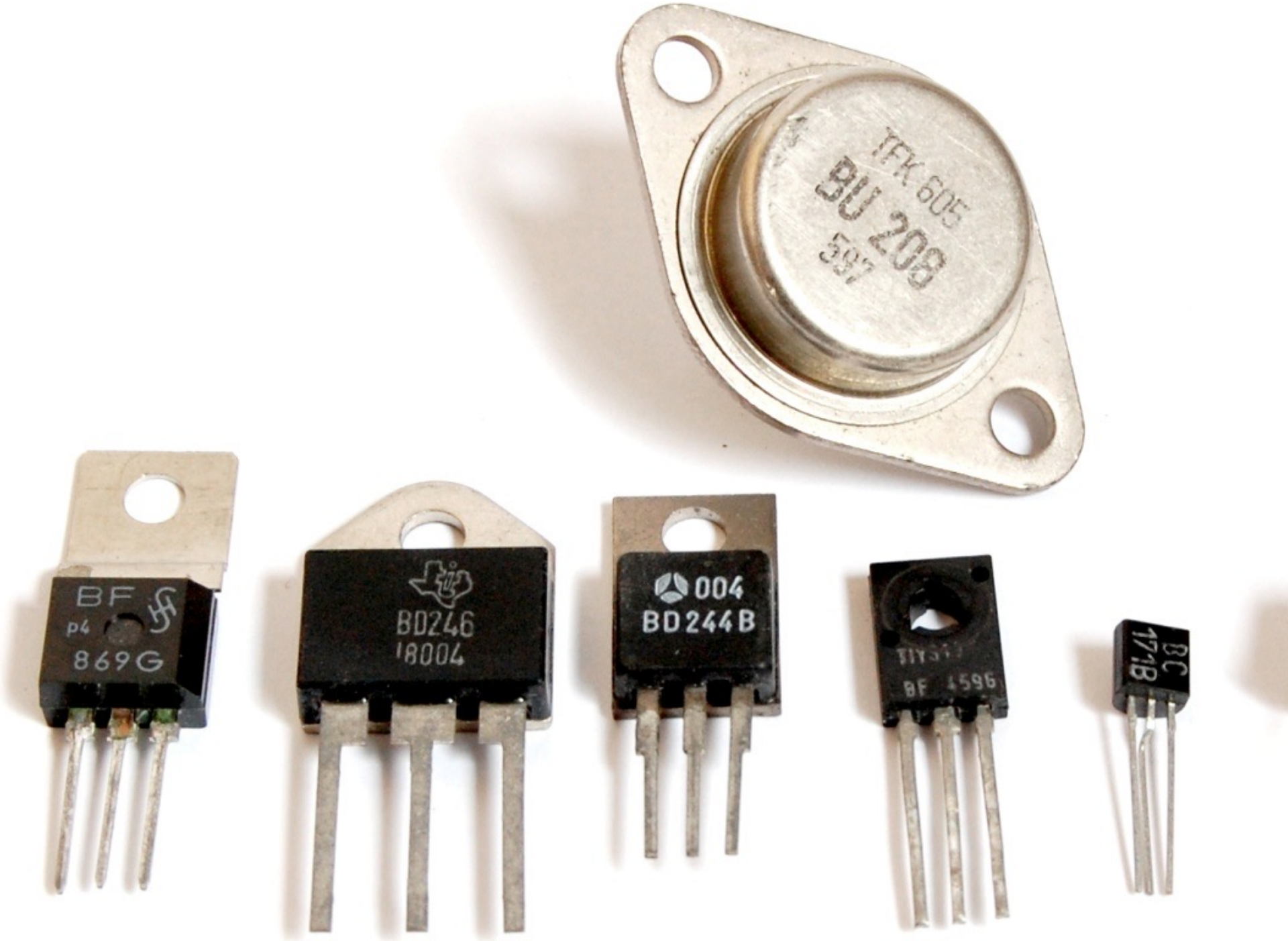
- Arten von Programmiersprachen, VMs, Interpreter, Compiler, Programmentwicklung, UML, Datentypen, Variablen, Kontrollstrukturen

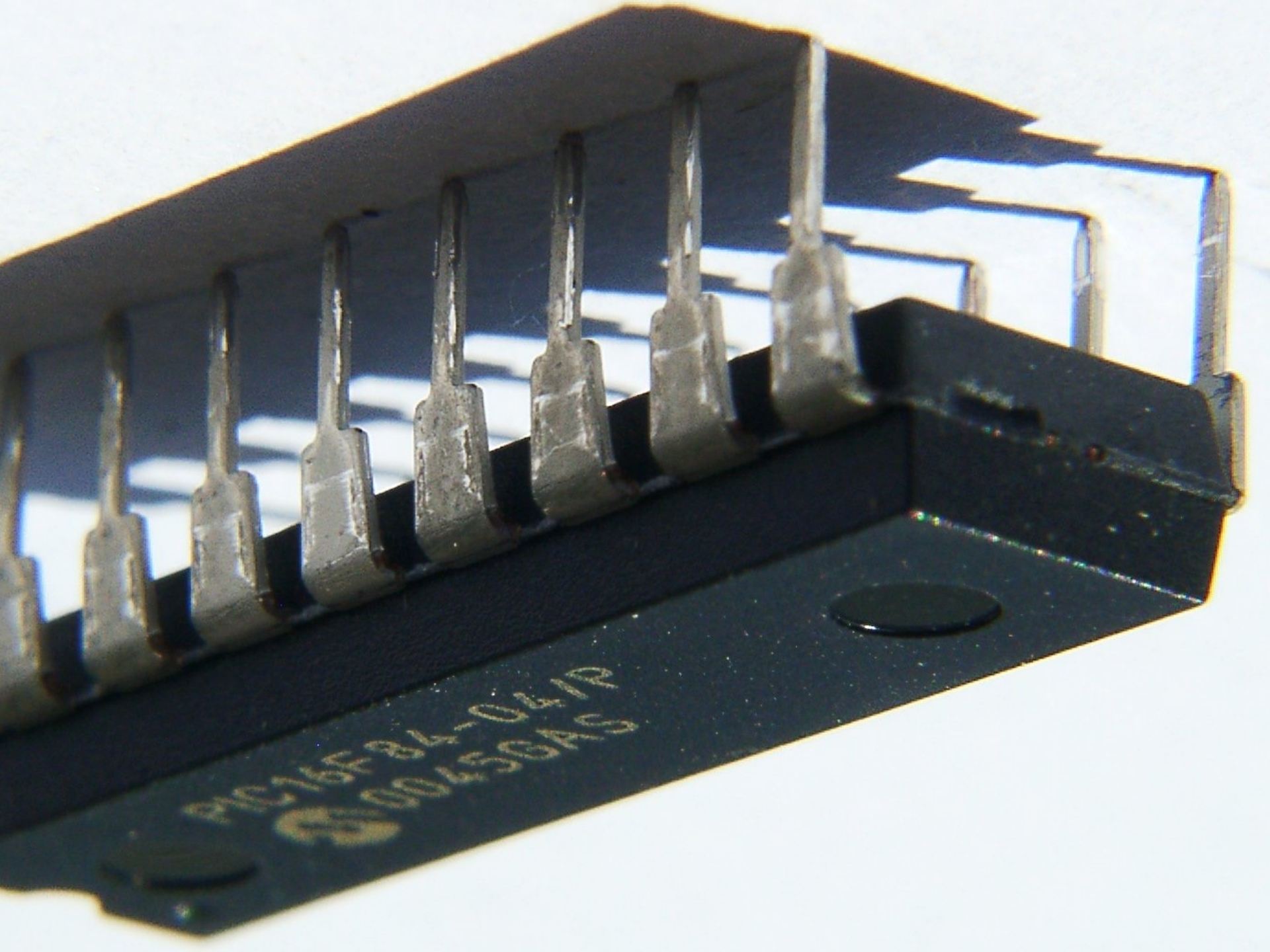
Theorie I: Binäre Logik, Gatter, Schaltungen



L1 → L2 → L3 → Ln


Register
Cache
Hauptspeicher
Magnetplatte
Band
optische Platte


TFK 605
BU 208
597
BF
p4
869G
BD246
18004
004
BD244B
BF 459
BC
141B


PIC16F84-04/P
9904SBAS

(Logik)Gatter

Vereinfacht: Blackbox mit n Eingängen und einem Ausgang

Eingänge / Ausgang: Spannungszustände, i.e. 0 Volt für 0 und 5 Volt für 1

Wahrheitstabelle

Für zwei Eingänge (A, B): $2^2=4$ Tabellenzeilen

A	B	Y
0	0	
0	1	
1	0	
1	1	

Bitte beachten: 0 und 1 sind in diesem Kontext Wahrheitswerte (0 ist FALSE, 1 ist TRUE)!

De Morgan‘sche Gesetze

Augustus De Morgan
(1806 – 1871)

Erstes Gesetz: $Z = \neg(A \wedge B) = \mathbf{\neg A \vee \neg B}$

A	B	$A \wedge B$	$\neg(A \wedge B)$	$\neg A$	$\neg B$	$\neg A \vee \neg B$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

Praxis I: Binäre Logik, Gatter, Schaltungen

Vervollständigen Sie das Symbol für die Konjunktion und geben Sie sowohl die Funktionsgleichung als auch die Wahrheitstabelle der Konjunktion wieder.

Symbol:

Funktionsgleichung:

Wahrheitstabelle:

Bestimmen Sie die vollständige Wahrheitstabelle für die folgende Funktionsgleichung mit den drei Variablen A, B und C:

$Y = ((A \vee B) \wedge (C \vee B)) \wedge \neg A$

$$Y = ((A \vee B) \wedge (C \vee B)) \wedge \neg A$$

A	B	C	D	E	F	Y
0	0	0	0	0	0	0
0	0	1	0	1	0	0
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	0	0	0
1	0	1	1	1	1	0
1	1	0	1	1	1	0
1	1	1	1	1	1	0

$$Y = ((A \vee B) \wedge (C \vee B)) \wedge \neg A$$

D

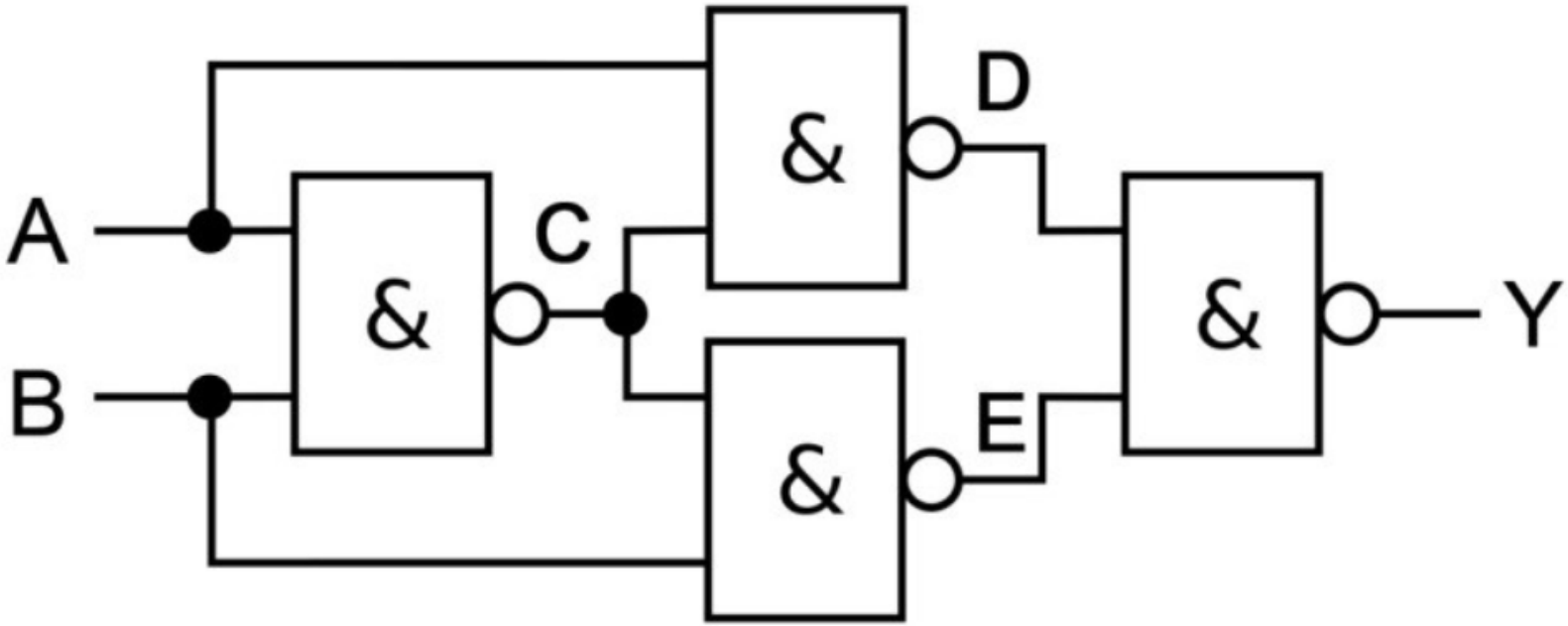
E

F

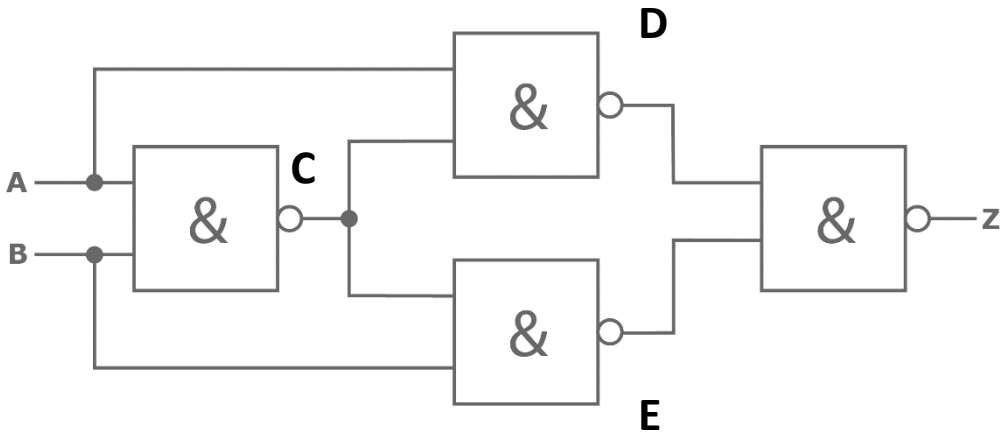
A	B	C	D	E	F	Y
0	0	0	0	0	0	0
0	0	1	0	1	0	0
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	0	0	0
1	0	1	1	1	1	0
1	1	0	1	1	1	0
1	1	1	1	1	1	0

Y = (NICHT A UND B UND NICHT C) ODER (NICHT A UND B UND C)

Erstellen Sie die Wahrheitstabelle für die im Folgenden dargestellte Schaltung. Geben Sie in Ihrer Wahrheitstabelle die Werte der Ausgänge C, D, E und Y für alle möglichen Zustände der Schaltung an. Beschreiben Sie die Schaltung auf Grundlage der Wahrheitstabelle mit einem logischen Ausdruck (Y = …).



Stellen Sie die zu der logischen Schaltung gehörende Wahrheitstabelle auf und beschreiben Sie die Schaltung mit Hilfe eines logischen Ausdrucks:



A	B	C = NICHT(A UND B)	D = NICHT (A UND C)	E = NICHT (B UND C)	Z = NICHT (D UND E)
0	0	1	1	1	0
0	1	1	1	0	**1**
1	0	1	0	1	**1**
1	1	0	1	1	0

$Z = \overline{(A \wedge B)} \vee \overline{(A \wedge B)}$

→ Exklusives ODER

Theorie II: Betriebssysteme

Themenüberblick „Betriebssysteme“

Zentrale Aufgaben:

- Dateiverwaltung
  - Abstraktionen / Schnittstellen für Zugriff auf Low-Level Funktionen (z.B. der Festplatte)
  - Blöcke / Cluster, Fragmentierung **!**

Prozessverwaltung / Ressourcenverwaltung / Zeitplanung → Scheduling

- Prozesse vs. Threads **!**
  - (Prozesszustände) **!**
- Scheduling / Multitasking **!**
  - Kooperatives Multitasking **!**
  - Präemptives Multitasking **!**

Speicherverwaltung

- Virtueller Speicher **!**
  - Swapping **!**
  - Paging **!**

Theorie III: Software-Entwicklung

Themenüberblick „Software-Entwicklung: klassisch vs. agil“

Phasen der Software-Entwicklung

- Analyse !
  - Spezifikation !
- Entwurf !
  - Algorithmus !
  - Pseudocode !
- Implementation
- (Post-Implementation)

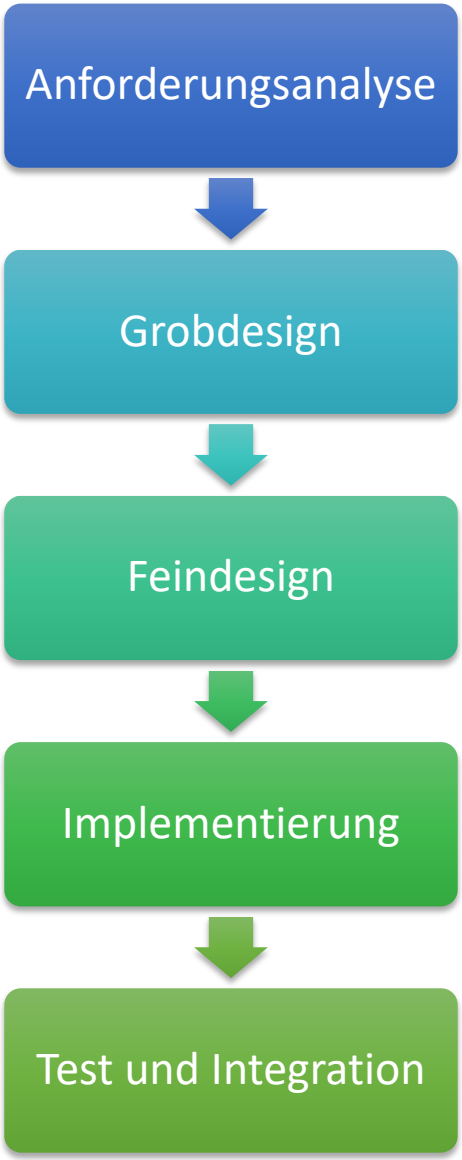
Vorgehensmodelle

- Wasserfallmodell !
- Prototypische Entwicklung !
- Iterative Entwicklung !
- Iterativ inkrementelle Entwicklung !

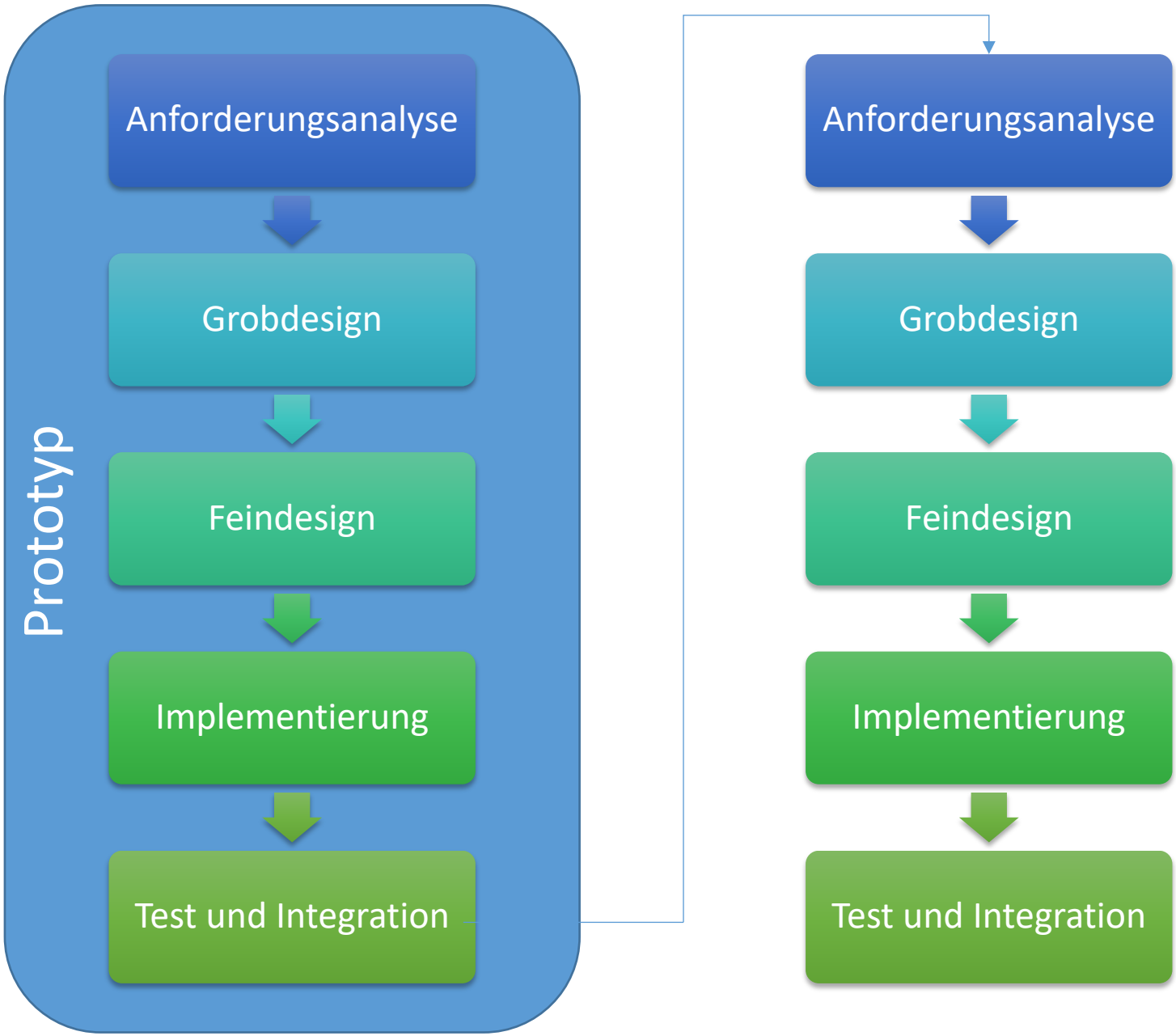
Agile Software-Entwicklung

- Intention
- Mockups !
- Werkzeuge und Methoden
- User Stories !
- User Story Mapping

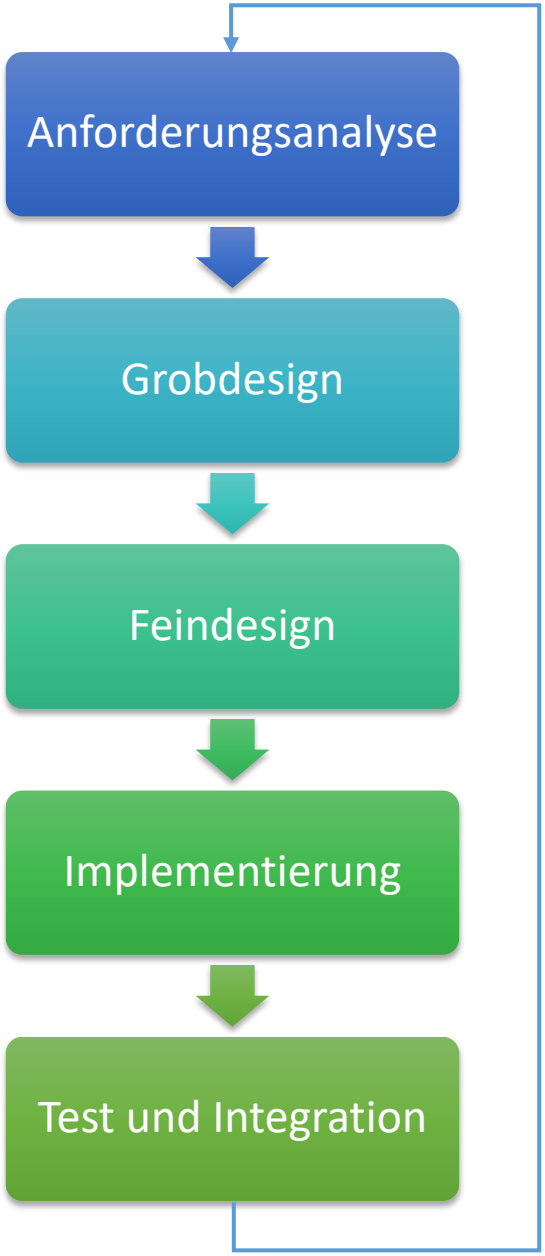
Wasserfallmodell



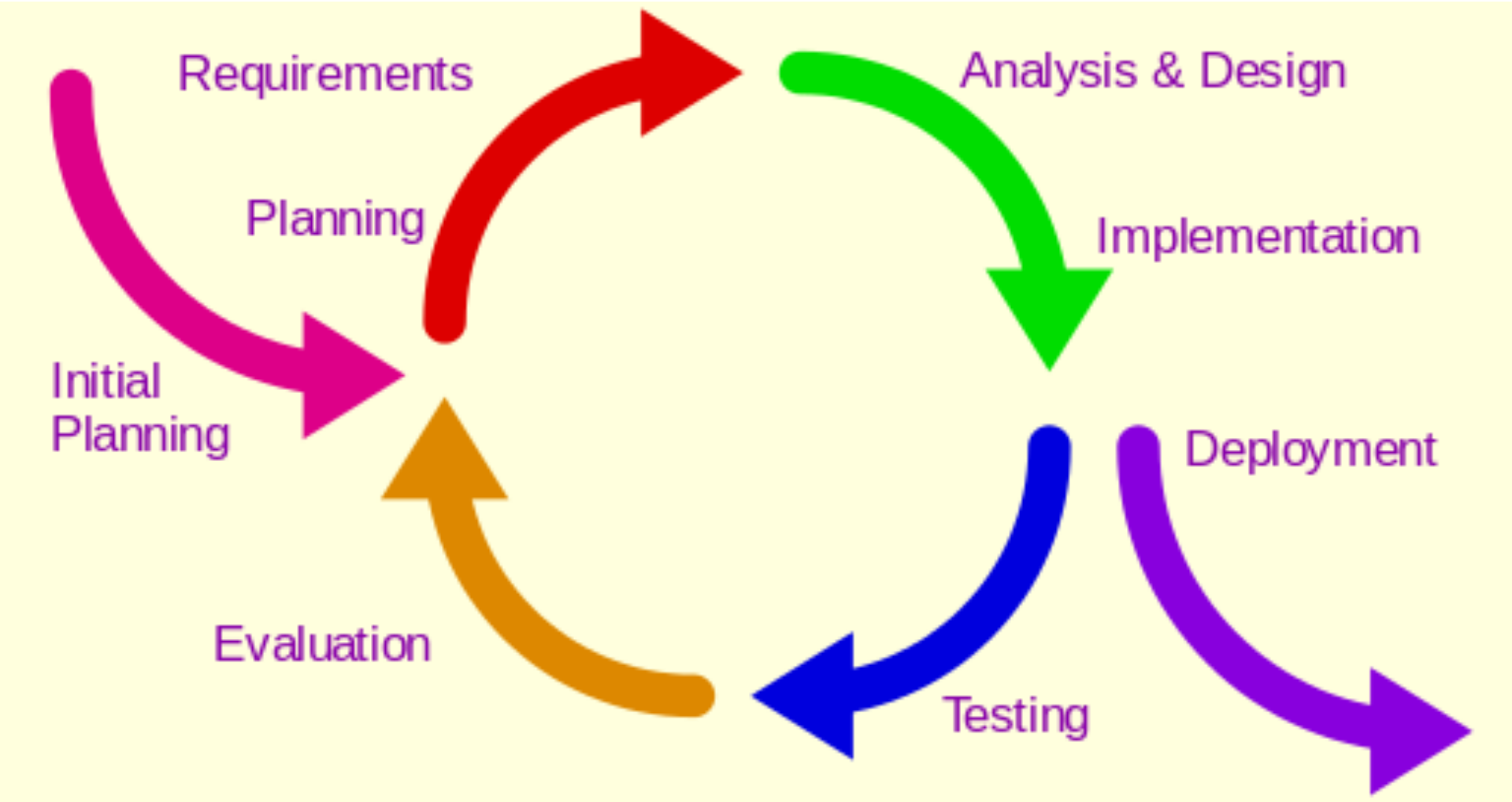
Prototypische Entwicklung



Iterative Entwicklung



Iterativ Inkrementelle Entwicklung (State of the Art)



Theorie IV: Programmiersprachen

Programmiersprachen und ihre Unterschiede

Differenzierung anhand von C++ und JavaScript:

- Compiler vs. Interpreter !
- Typisierung !: Dynamisch vs. statisch !
- Variablen: Deklaration vs. Initialisierung !
- Paradigmen: funktionale vs. Objektorientierte Programmierung
- Objektorientierung: Klassen, Kapselung, Geheimnisprinzip !
- Hardwarenahe Programmierung: C++ und Zeiger
- Gemeinsamkeiten: Auswahlanweisungen und Kontrollstrukturen !