Grundfragen der Informatik, KV

Boolesche Algebra und Aussagenlogik

Überblick

Aussagenlogik

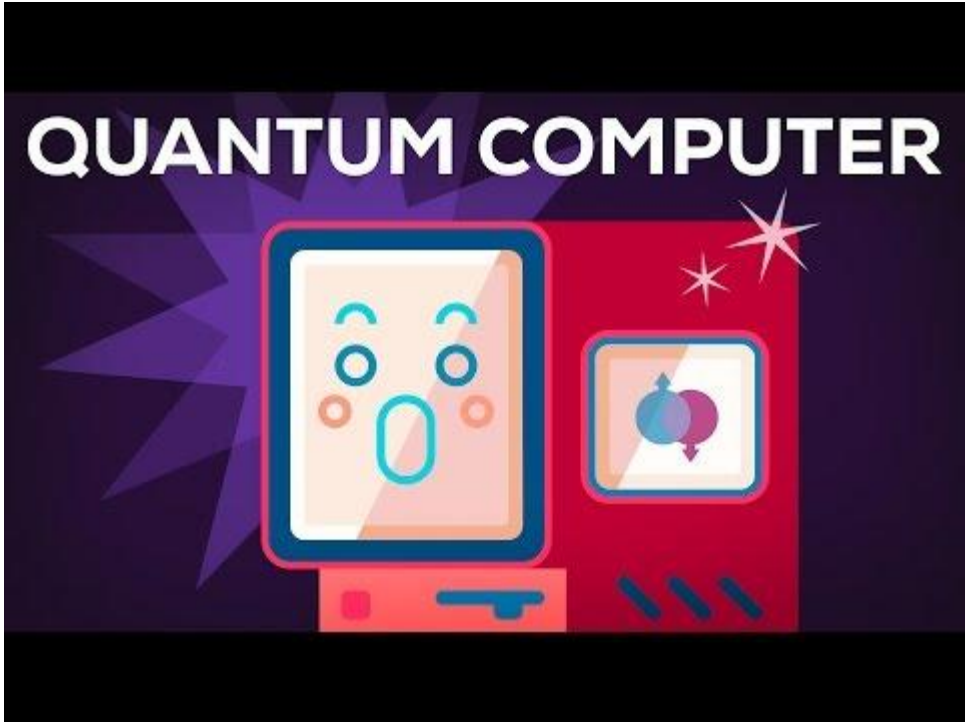
Boolesche Operatoren

Wahrheitstabellen

Logikgatter

Parsetree

Konjunktive Normalform (KNF)



(bis 02:15)

Aussagenlogik

Ist eine einfache Logik und Grundlage für weitere Logiken.
Damit lassen sich Aussagen formulieren, die wahr/falsch (1/0) sind.

Aussagen im Sinne der Logik

“Heute hat es in Graz geregnet.”	*//wahr, wenn es geregnet hat*
“Die Erde ist rund”	*//wahr*
1 + 1 = 2.	*//wahr*
“Der Papst ist Präsident der USA.”	*//falsch*

Keine Aussagen

“Geh lernen!”	*//Imperativ*
“Magst du Bier?”	*//Frage*
“Rot ist besser als Gelb!”	*//ein Gefühl (subjektiv)*
“Riech dich später!”	*//Floskeln… etc.*

Aussagen lassen sich mit Junktoren verknüpfen

- “Es regnet” **und** “die Straße ist nass” — **AND** — $p \wedge q$
- “Ich trinke Bier” **oder** “ich trinke Wein” — **OR** — $p \vee q$
- “Ich trinke gerade **nichts**.” — **NOT** — $\neg p$
- **Wenn** “ich die Prüfung schaffe, **dann** “geh ich ein Bier trinken” — **IMPLIES** — $p \rightarrow q$
- Morgen ist Freitag, **nur wenn** heute Donnerstag ist.
  (gestern war Donnerstag, weil heute Freitag ist) — **EQUALS** — $p \leftrightarrow q$
- **Entweder** “du isst das Eis”, **oder** “du bekommst nichts” — **XOR** — $p \oplus q$

  Aus dem Wahrheitsgehalt von A und B lässt sich der Wahrheitswert der zusammengesetzten Aussage schlussfolgern.

  Mit Klammern $(p \rightarrow q) \wedge q$ können Terme zusammengefasst werden

Boolesche Algebra

Menge {0,1}

Binäre Operatoren: ∧ (UND) ∨ (exklusives ODER)
Unäre Operatoren: ¬ (NEGIERUNG)

∧	0	1
0	0	0
1	0	1

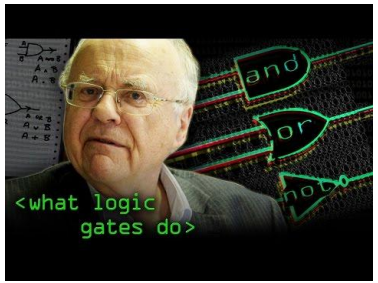
“So wie Multiplikation”
Logisches Produkt

∨	0	1
0	0	1
1	1	1

“So wie Addition”
Logisches Summe

¬	
0	1
1	0

“Konverter”



“Punkt vor Strich”

1.	Negation	NOT	¬
2.	Konjunktion	AND	∧
3.	Disjunktion	OR	v
4.	Implikation	implies	→

rechts orientiert:

$p \rightarrow q \rightarrow r$	heißt	$p \rightarrow (q \rightarrow r)$

Hands-On

Formalisiere folgende Aussagen:

(1) Diese Hütte steht nicht an einem See oder der Schatz ist nicht in der Küche.

(2) Wenn der Baum vor der Hütte eine Ulme ist, dann ist der Schatz in der Küche.

(3) Der Baum hinter der Hütte ist eine Eiche oder der Baum vor der Hütte ist keine Ulme oder der Baum hinter der Hütte ist eine Ulme.

**(4) Die Aussage ”Der Baum vor der Hütte ist keine Ulme
und der Schatz ist nicht unter dem Fahnenmast begraben“ ist nicht wahr.**

(5) Wenn der Baum hinter der Hütte eine Eiche ist, dann ist der Schatz im Keller.

Hands-On

Formalisiere folgende Aussagen:

(1) Diese Hütte steht nicht an einem See oder der Schatz ist nicht in der Küche.

(2) Wenn der Baum vor der Hütte eine Ulme ist, dann ist der Schatz in der Küche.

(3) Der Baum hinter der Hütte ist eine Eiche oder der Baum vor der Hütte ist keine Ulme oder der Baum hinter der Hütte ist eine Ulme.

(4) Die Aussage ”Der Baum vor der Hütte ist keine Ulme
und der Schatz ist nicht unter dem Fahnenmast begraben“ ist nicht wahr.

(5) Wenn der Baum hinter der Hütte eine Eiche ist, dann ist der Schatz im Keller.

Formalisierung der Schatzsuche

(1)	Diese Hütte steht nicht an einem See oder der Schatz ist nicht in der Küche.	¬See ∨ ¬Kü
(2)	Wenn der Baum vor der Hütte eine Ulme ist, dann ist der Schatz in der Küche.	BvHU ⇒ Kü
(3)	Der Baum hinter der Hütte ist eine Eiche oder der Baum vor der Hütte ist keine Ulme oder der Baum hinter der Hütte ist eine Ulme.	BhHE ∨ ¬BvHU ∨ BhHU
(4)	Die Aussage ”Der Baum vor der Hütte ist keine Ulme und der Schatz ist nicht unter dem Fahnenmast begraben“ ist nicht wahr.	¬(¬BvHU ∧ ¬Fa)
(5)	Wenn der Baum hinter der Hütte eine Eiche ist, dann ist der Schatz im Keller.	BhHE ⇒ Ke
Der Baum **hinter** der Hütte ist keine Ulme:		¬BhHU

Wie zeichne ich eine Wahrheitstabelle...

AND

p	q	p ∧ q

Diesen Term wollen wir untersuchen

Wahrheitstabellen

AND

p	q	p ∧ q
T	T	T
T	F	F
F	T	F
F	F	F

OR

p	q	p v q
T	T	T
T	F	T
F	T	T
F	F	F

XOR

p	q	p ⊕ q
T	T	F
T	F	T
F	T	T
F	F	F

NOT

p	¬p
T	F
F	T

IMPLIES

p	q	$p \rightarrow q$	$\neg p \vee q$
T	T	T	
T	F	F	
F	T	T	
F	F	T	

EQUAL

p	q	$p \leftrightarrow q$	$(p \rightarrow q) \wedge (q \rightarrow p)$
T	T	T	
T	F	F	
F	T	F	
F	F	T	

IMPLIES

p	q	$p \rightarrow q$	$\neg p \vee q$
T	T	T	
T	F	F	
F	T	T	
F	F	T	

EQUAL

p	q	$p \leftrightarrow q$	$(p \rightarrow q) \wedge (q \rightarrow p)$
T	T	T	
T	F	F	
F	T	F	
F	F	T	

Hands-On: Vervollständigen der Tabelle

IMPLIES

p	q	p → q	¬p v q
T	T	T	T
T	F	F	F
F	T	T	T
F	F	T	T

$p \rightarrow q \equiv \neg p \vee q$

EQUAL

p	q	p ↔ q	(p → q) ∧ (q → p)
T	T	T	T
T	F	F	F
F	T	F	F
F	F	T	T

p→q	q→p	
T	T	
F	T	
T	F	
T	T	

Aussagenlogische Formeln, die in jeder Interpretation wahr sind, nennt man **Tautologien = T**

p	¬p	p ∨ ¬p
T	F	T
F	T	T

Aussagenlogische Formeln, die in mindestens einer Interpretation wahr sind, nennt man **erfüllbar**

p	q	p ∧ q
T	T	T
T	F	F
F	T	F
F	F	F

Aussagenlogische Formeln, die in keiner Interpretation wahr sind, nennt man **Kontradiktionen = ⊥**

p	¬p	p ∧ ¬p
T	F	F
F	T	F

Aussagenlogische Formeln, in denen alle Interpretation wahr sind, nennt man **gültig**

p	¬p	p ∨ ¬p
T	F	T
F	T	T

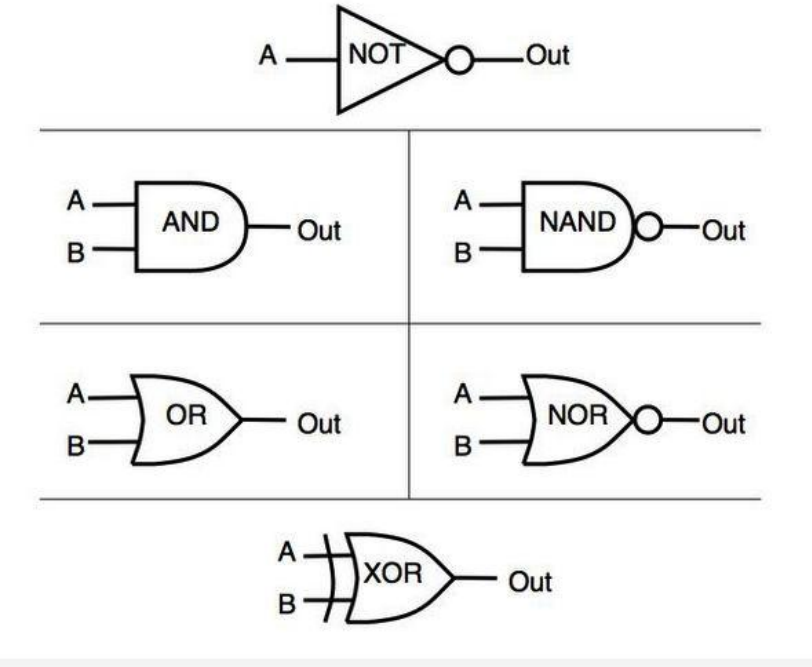
Mit XOR (exklusives ODER) wird binäre Addition umgesetzt

⊕	0	1
0	0	1
1	1	0

```
   10 = 2
⊕  11 = 3
  101 = 5
```

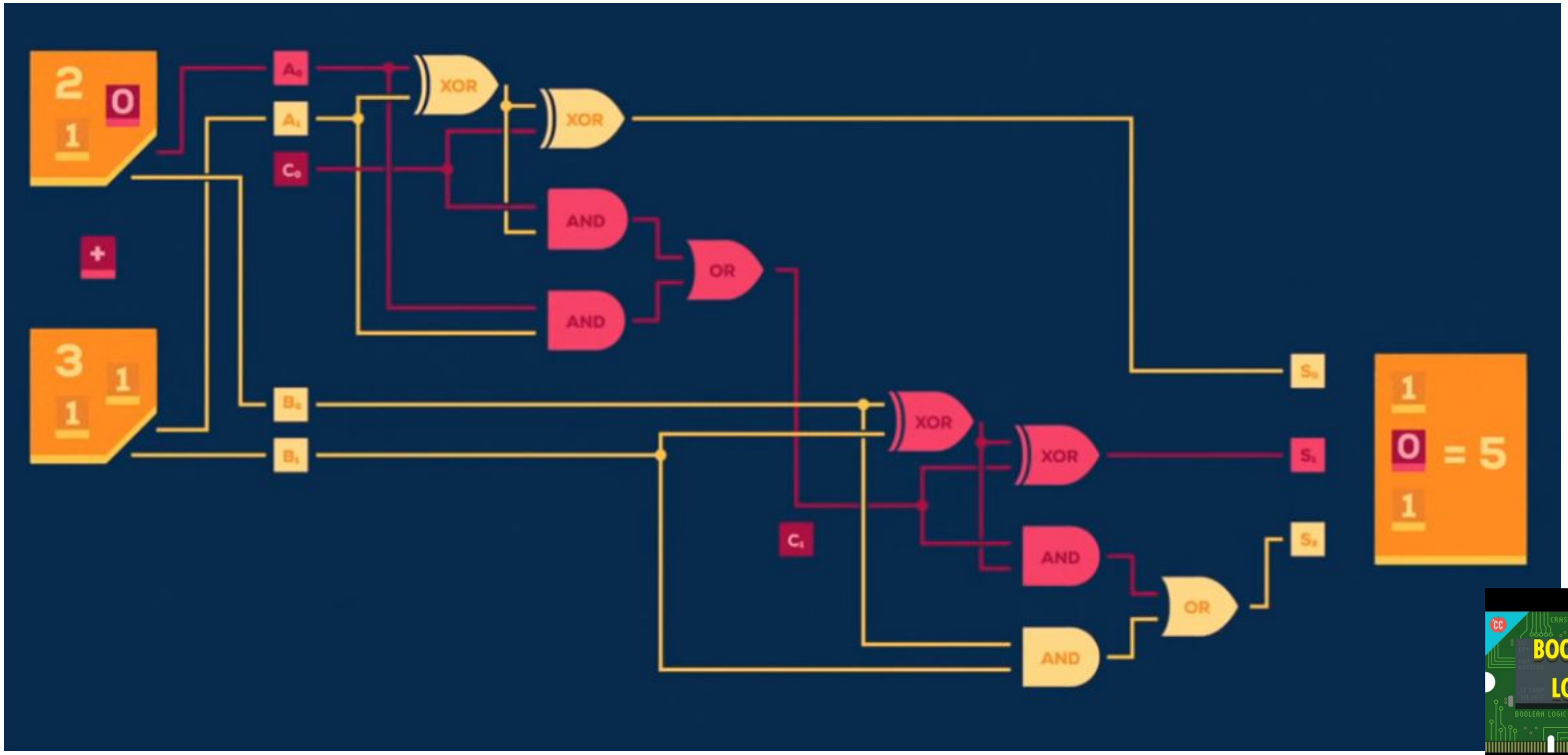
Logikgatter / Logic Gates

Logikgatter sind Anordnungen (elektronische Schaltung) zur Realisierung einer booleschen Funktion, die binäre Eingangssignale durch Implementierung logischer Operatoren (AND, OR, XOR, NOT) zu einem binären Ausgangssignal verarbeitet (einem logischen Ergebnis umgewandelt).

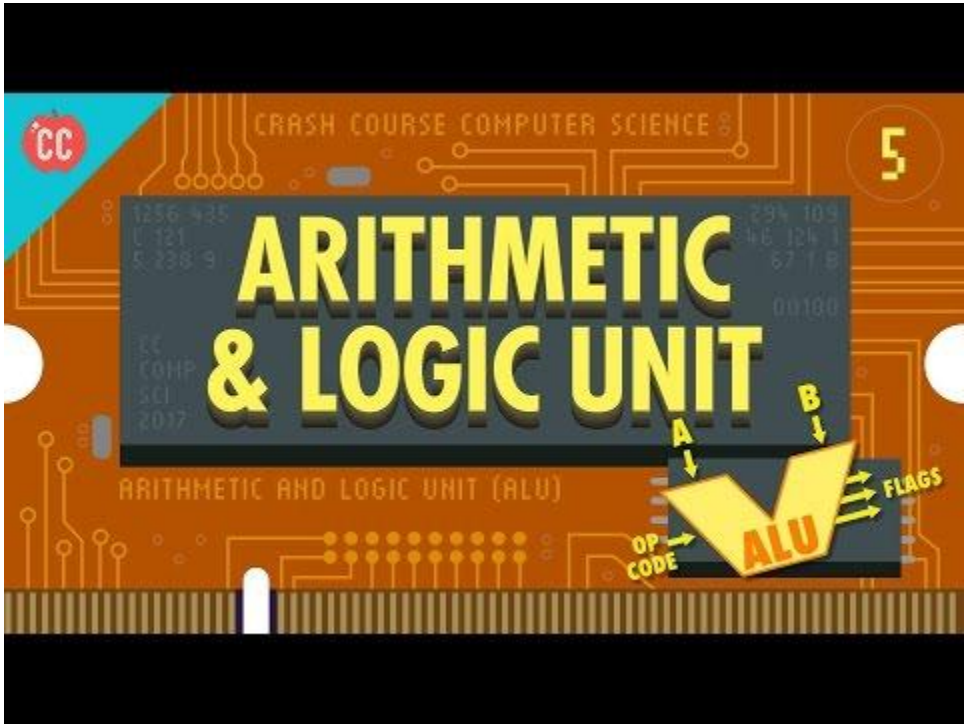


Die Verschachtelung von Logikgattern, um die Addition von 2 und 3 umzusetzen

```
  10
 +11
------
 101
```



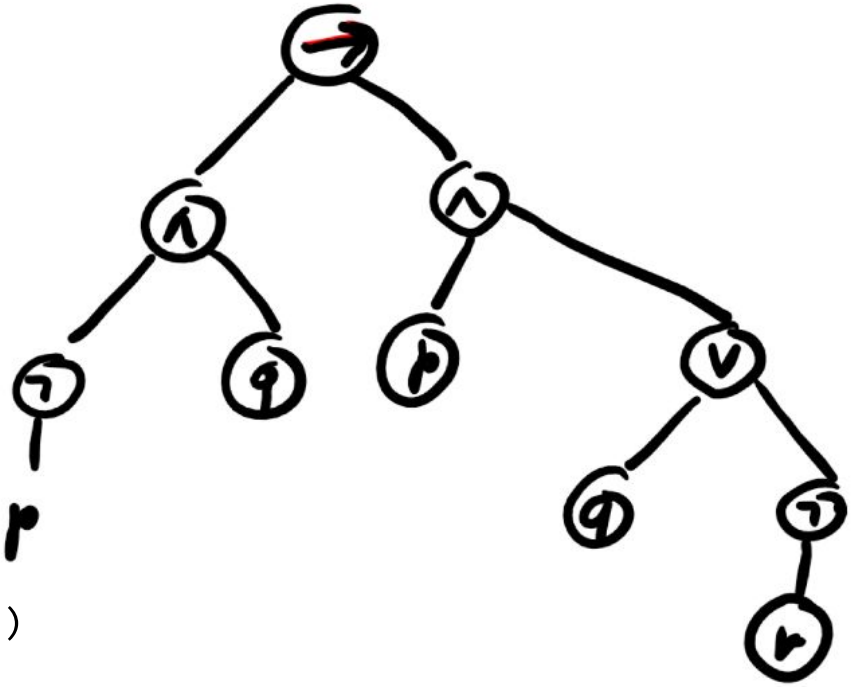
Arithmetic and Logic Unit



Parse Tree / binärer Entscheidungsbaum mit Modell

$(\neg p \wedge q) \Rightarrow (p \wedge (q \vee \neg r))$

M: p = T, q = F, r = T



Wahrheitstabelle!

WolframAlpha:
https://www.wolframalpha.com mit folgendem Input:

```
(not(p) and q) implies (p and (q or not(r))
```

https://www.wolframalpha.com/input/?i=%28not%28p%29++AND+q%29+IMPLIES+%28+p+AND+%28q+OR+not%28r%29%29+%29:

```
(not(p)  AND q) IMPLIES ( p AND (q OR not(r)) )
```

Truth table:

p	q	r	$\neg p \wedge q \Rightarrow p \wedge (q \vee \neg r)$
T	T	T	T
T	T	F	T
T	F	T	T
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	T

M: p = **T**, q = **F**, r = **T**

Ein anderes Modell, bei dem es F wird:

M: p = **F**, q = **T**, r = **T**

Konjunktive Normalform (KNF)

Unter der konjunktiven Normalform versteht man einen aussagenlogischen Ausdruck, der aus einer beliebigen Anzahl von UND-verknüpften Klauseln gebildet wird.

Jede beliebige boolesche Funktion lässt sich als KNF darstellen. Es lässt sich auch ein allgemeiner Algorithmus angeben aus dem die KNF eines beliebigen Ausdrucks gebildet werden kann:

1. Erstelle die Wahrheitstabelle.

2. Wähle alle Konfigurationen unter denen die Formel zu falsch evaluiert.

3. Negiere alle Variablen.

4. Verbinde alle Variablen einer Konfiguration mit ODERs.

5. Verbinde alle Konfigurationen mit mehreren UNDs

Videotutorial: https://www.youtube.com/watch?v=WRYAmf864tQ

Beispiel KNF

a	b	c	$(a \wedge \neg b) \vee c$
T	T	T	T
T	T	F	F
T	F	T	T
T	F	F	T
F	T	T	T
F	T	F	F
F	F	T	T
F	F	F	F

(¬a v ¬b v c)

∧

(a v ¬b v c)

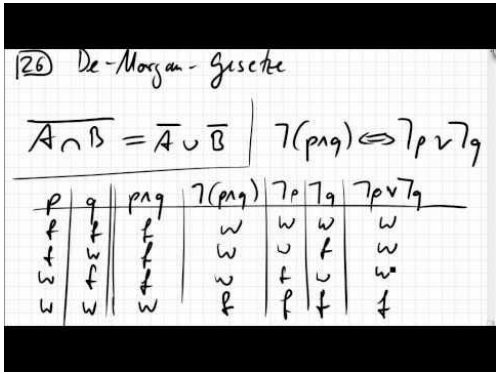
∧

(a v b v c)

Umformen “Rechenregeln”

Teilformel dürfen durch äquivalente Formel ersetzt werden:

A	$\equiv$	$\neg\neg A.$	Doppelte Negation
$A \Rightarrow B$	$\equiv$	$\neg A \vee B.$	Implikation
$A \Leftrightarrow B$	$\equiv$	$(A \Rightarrow B) \wedge (B \Rightarrow A).$	Äquivalenz
$A \vee B$	$\equiv$	$B \vee A.$	Kommutativität
$A \wedge B$	$\equiv$	$B \wedge A.$	
$A \Rightarrow B$	$\equiv$	$(\neg B) \Rightarrow (\neg A).$	Kontraposition
$\neg(A \vee B)$	$\equiv$	$(\neg A) \wedge (\neg B).$	De Morgan
$\neg(A \wedge B)$	$\equiv$	$(\neg A) \vee (\neg B).$	



https://de.wikipedia.org/wiki/Boolesche_Algebra

Boolesche Algebra in Python

https://www.onlinegdb.com/online_python_compiler

Ein *boolescher Ausdruck* ist ein Ausdruck, der entweder *wahr* oder *falsch* ist, in Python True und False genannt. Sind folgende aussagen in Python True oder False:

```
>>> "Hallo Welt" > "Hallo"
```
//true

```
>>> 4 != 5
```
//true

```
>>> 4 == 4 or 4 == 5
```
//true

```
>>> "Hallo" == "Hallo Welt"
```
//false

```
>>> not not 5 == 5
```
//true

```
>>> not 3 <= 4 and not 4 >= 5
```
//not TRUE and not FALSE
//FALSE and TRUE
//FALSE

https://www.onlinegdb.com/online_python_compiler

```
print(4 > 5)

print(4 != 5)

print(4 == 4 or 4 == 5)

print("Hallo" == "Hallo Welt")

print(not not 5 == 5)

print(not 3 >= 4 and not 4 >= 5)
```

Übungen

Assignment 1

1. Kodiere deinen Vornamen (Text) in hexadezimal, dezimal und in binär. Nutze dazu die ASCII Tabelle. Zeige für einen Wert wie die manuelle Umrechnung (binär->hex; binär -> dezimal) funktioniert.

   Text: Christopher
   Hex: 43 68 ...
   Binär: 01000011 01101000 ...
   Dezimal: ...

2. Wandle die dezimal Zahlen 12 und 8 in Binärzahlen um und führe folgende arithmetischen Operationen durch:

   2.1) 12 + 8 = ?
   2.2) 12 - 8 = ?
   2.3) 8 - 12 = ?
   2.4) 5 * 10 = ?
   2.5) 9 : 3 = ?

3. Wie wird die Zeichenkette 123 gespeichert, wenn die UTF8-Kodierung verwendet wird? Welche Codepoints sind das? Wie sieht die binäre Form davon aus.

4. Kann die Zahl 123 in einem Byte als Binärzahl gespeichert werden? Auch im 2-er Komplement?

Assignment 1

5. Kopiere das tschechische Wort *Žluté* in einem beliebigen Texteditor und speichere es ab. Verwende den Befehl `hexdump` auf dieses file. Nimm das Ergebnis das herauskommt und konvertiere es nach Binär (https://www.rapidtables.com/convert/number/hex-to-binary.html) und dieses Ergebnis nach ASCII https://www.rapidtables.com/convert/number/binary-to-ascii.html.

   Welches Ergebnis bekommst du und warum?

6. Gegeben ist ein Alphabet {a, b, c} mit den Auftrittswahrscheinlichkeiten p {a = 0,7; b = 0,2 ; c = 0,1}. Berechne H (mittleren Informationsgehalt). Christopher Egger

Assignment 2

1.) Formalisiere jede der folgenden Aussagen durch eine aussagenlogische Formel:

1.1) *Wenn du die drei Fragen beantwortet hast, dann kannst du die Brücke des Todes überqueren.*
1.2) *Die Ritter der Kokosnuss gibt es nur zusammen mit Pferden und Kokosnüssen.*
1.3) *Entweder wird ein Gebüsch gekauft oder es wird kein Gebüsch gekauft.*

2.) Formuliere die Regel, "wenn der Hahn auf dem Misthaufen kräht, dann ändert sich das Wetter oder es bleibt so, wie es ist" als Formel der Aussagenlogik. Zeige, dass die Formel eine Tautologie ist (Wahrheitstabelle)

3.) Begründe ob folgende Formeln erfüllbar, gültig oder unerfüllbar sind. Erfüllbar bist ist eine Formel, wenn mindesten einmal "wahr" in der Wahrheitstabelle auftaucht. Gültig wenn alle "wahr" sind und unerfüllbar wenn alle "falsch" sind.

3.1) $a \rightarrow a \land b$
3.2) $(a \rightarrow b) \rightarrow (b \rightarrow a)$
3.3) $(a \land b) \rightarrow (\neg a \lor \neg b)$
3.4) $(\bot \rightarrow p) \land \neg(q \leftrightarrow p) \land \neg((\neg p \land \neg q) \rightarrow q)$

4.) Zeichne einen Parse Tree für folgende Formel und verwende den Tree, um zu überprüfen ob die Zuordnung p = F, q =T, r = F die Formel true oder false werden lässt. Finde ein Modell, welches das Gegenteil erreicht.
$(\neg(r \leftrightarrow q) \rightarrow \neg r) \land (\neg(r \rightarrow q) \lor (p \rightarrow q))$

5.) Gib folgende Befehle in deiner Konsole ein. Erkläre was jeweils passiert und um welche unterschiedlichen Operatoren es sich handelt. (https://www.linux.com/tutorials/logical-ampersand-bash/)

- *echo $((2 & 3))*
- *echo $((2 && 3))*
- *echo $((5 | 5))*
- *echo $((5 || 5))*

Bonus:
Was ist die Konjunktive Normalform (KNF) des folgenden logischen Ausdrucks:
(p -> q) ∧ (¬q v p)

Wahrheitstabellen - Beispiele

1. $(\neg p \vee q) \wedge (p \rightarrow q)$

2. $(p \rightarrow q) \wedge (q \rightarrow p)$

Wahrheitstabellen - Beispiele

1.)

p	q	¬p	¬p v q	p → q	(¬p v q) ∧ (p → q)
T	T	F	T	T	T
T	F	F	F	F	F
F	T	T	T	T	T
F	F	T	T	T	T

2.)

p	q	p → q	q → p	(p → q) ∧ (q → p)
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

Hands-On (p IMPLIES q) IMPLIES p

p	q	$(p \rightarrow q) \rightarrow p$		
T	T			
T	F			
F	T			
F	F			

Hands-On

p	q	$p \rightarrow q$	$(p \rightarrow q) \rightarrow p$
T	T	T	T
T	F	F	T
F	T	T	F
F	F	T	F

Hands-On

p	q	$(p \vee q) \leftrightarrow (\neg p \rightarrow q)$		
T	T			
T	F			
F	T			
F	F			

Hands-On (p OR q) EQUAIVALENT (not(p) IMPLIES q)

p	q	¬p	¬p → q	p ∨ q	(p ∨ q) ↔ (¬p → q)
T	T	F	T	T	T
T	F	F	T	T	T
F	T	T	T	T	T
F	F	T	T	F	T

Übung

Verwende Junktoren um **folgende Sätze** so detailliert wie möglich in Aussagenlogik darzustellen:

(a) If you are a good student, then you like Logic and Computability.

(b) In the evening, Paul will study or go out with his friends, but not both

(c) If both Christopher and Sepp pass the exam, they have a party

Übung (Ass2)

Zeige mit einer **Wahrheitstabelle** ob folgende Formel erfüllbar ist.
Eine Formel ist erfüllbar, wenn mindestens eine Belegung der Variablen eine wahre Aussage erzeugt. Erzeuge eine KNF.

(p -> q) ∧ (¬q v p)

Übung

Zeichne einen **Parse Tree** für folgende Formel und verwende die Zuteilungen (das Modell): p = F, q = T, r = F
Ist diese Formel erfüllbar?

((q -> ¬p) v r) -> (q ∧ (r -> p))

Übung

$(p \land q) \rightarrow \neg((q \rightarrow \bot) \land (p \rightarrow \top))$

- Prüfe mit einer Wahrheitstabelle ob diese Aussage gültig oder erfüllbar ist.
- Prüfe mit einem Parse Tree ob die Zuordnung
  p = T, q = T wahr oder falsch ergibt. (Achte auf $\bot$ und $\top$)