Informatik Klasse 8

Unterrichtsbegleitende Materialien und Aufgaben

Wie rechnet ein Computer?

Zahlendarstellung

Das Dezimalsystem

Das Dezimalsystem kennen wir alle. Das sind die Zahlen, die wir lesen und sofort verstehen können. Schauen wir noch einmal, wie diese Zahlen mathematisch funktionieren. Jede Ziffer bekommt eine unterschiedliche Bedeutung, je nachdem, wo sie in der Zahl steht. Ganz rechts sind die "Einer". Das heißt, eine 1 bedeutet auch 1. An der zweiten Stelle von rechts bedeutet die 1 schon 10 usw. Das lässt sich mathematisch so darstellen:

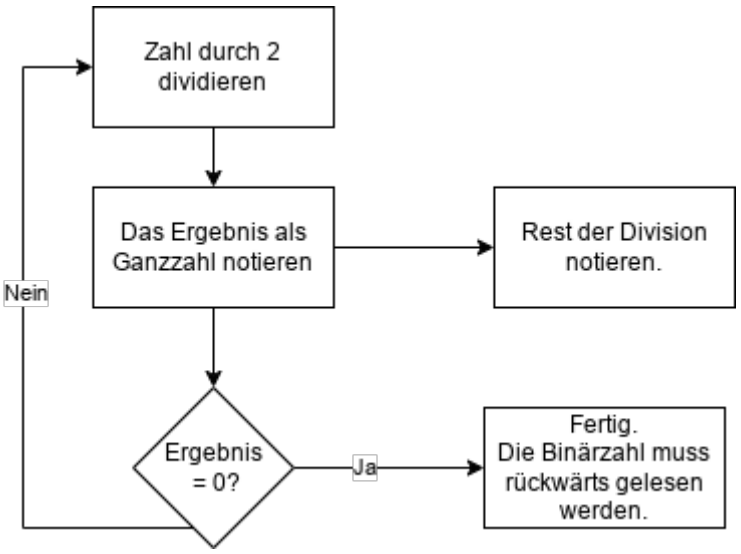
$$4711_{10} = 4 * 10^3 + 7 * 10^2 + 1 * 10^1 + 1 * 10^0$$

Dasselbe funktioniert auch, wenn wir weniger Ziffern zur Verfügung haben, z.B. nur zwei, nämlich 0 und 1:

$$1101_2 = 1 * 2 \ + 1 * 2 \ + 0 * 2 \ + 1 * 2 \ = 1 * 8 + 1 * 4 + 0 * 2 + 1 * 1 = 13_{10}$$

Auf diese Weise können Binärzahlen in Dezimalzahlen umgewandelt werden. Zur umgekehrten Rechnung kann man einen Algorithmus verwenden.

Algorithmus zum Berechnen einer Binärzahl



Beispiel:

```
190:2=□95□Rest 0
95:2=□47□Rest 1
47:2=□23□Rest 1
23:2=□11□Rest 1
11:2=□5□Rest 1
5:2=□2□Rest 1
2:2=□1□Rest 0
1:2=□0□Rest 1
```

Die Binärzahl lautet dann 10111110. Sie muss also von unten nach oben gelesen werden.

Programmierung dieses Algorithmus

So sieht die Grundstruktur des Programms aus:

```
import math
def convertDecToBin(dec):
    pass
    # Code

def convertToDec(number):
    pass
    # Code
number = input("Welche Zahl moechtest du umwandeln?\n")

zahl= convertDecToBin(int(number))
print(f"Die Zahl {number} als Binaerzahl lautet: {zahl}")
print(f"Die urspruengliche Zahl war: {convertToDec(zahl)}")
```

Addition von Binärzahlen

Addition von Dezimalzahlen

$$\begin{array}{rr} & 2752 \\ + & 4261 \\ \hline & 7013 \end{array}$$

Addition von Binärzahlen

$$\begin{array}{rr} & 10010 \\ + & 100111 \\ \hline & 111001 \end{array}$$

Die Addition funktioniert wie die Addition von Dezimalzahlen. Bei Dezimalzahlen stehen 10 Ziffern zur Verfügung. Ab der 10. Ziffer kommt es zu einem Überlauf, d. h., sie lässt sich nicht mehr darstellen. Daher muss die 10er-Stelle als Übertrag notiert werden und bei der Addition der nächsten Stelle berücksichtigt werden. Bei den Binärzahlen stehen nur zwei Ziffern zur Verfügung. Daher kommt es schon früher zu einem Überlauf. Dieser muss genauso berücksichtigt werden wie bei der Addition von Dezimalzahlen.

Logische Schaltungen

Ein Computer kann nur Nullen und Einsen. Das weiß mehr oder weniger jeder. Doch wie kann ein Computer damit rechnen? Darum geht es in diesem Kapitel.

Simulation von logischen Schaltungen

Zur Übung simulieren wir die Schaltungen, aus denen ein Computerchip aufgebaut ist, mit Digital Logic Sim. Die ZIP-Datei muss nur entpackt werden und das Programm wird dann mit einem Doppelklick auf `Digital LOgic Sim.exe` gestartet.

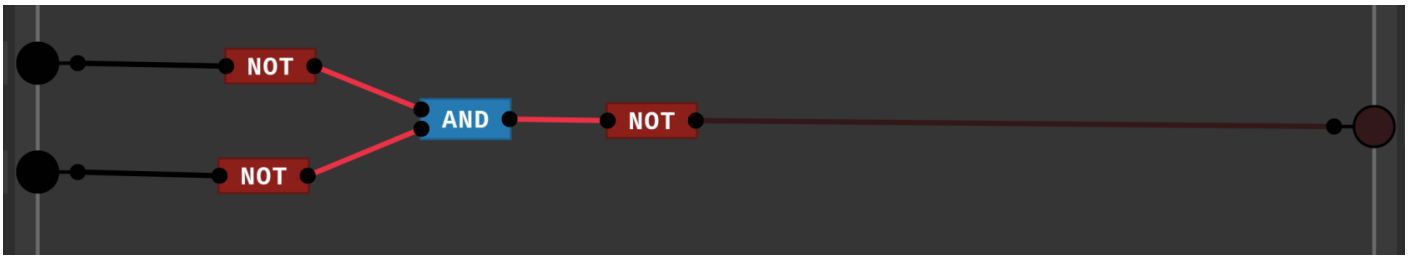
Mithilfe dieses Programms soll aus einer AND- und einer NOT-Schaltung ein Addierer aufgebaut werden, mit dem man zwei 1-Bit Binärzahlen addieren kann. Wer weitermachen möchte, kann damit natürlich auch einen 4-Bit Addierer bauen. Unter Windows werden die Daten unter diesem Verzeichnis gespeichert:

`C:\Benutzer\BENUTZERNAME\AppData\LocalLow\SebastianLague\Digital Logic Sim\V1\Projects`.

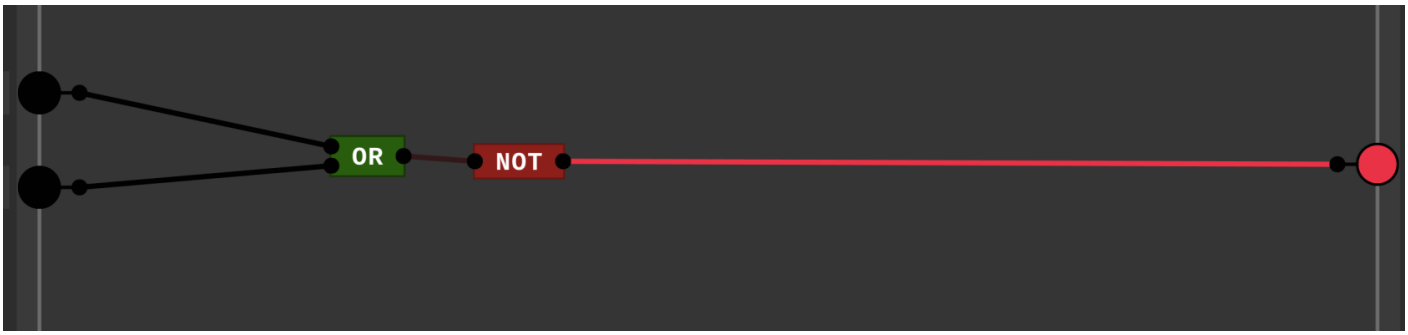
Um das Verzeichnis im *Windows Explorer* anzuzeigen muss unter *Ansicht* noch *Ausgeblendete Elemente* angewählt werden.

Hier sind die Schaltbilder der einzelnen Elemente. Diese müssen in dieser Reihenfolge angelegt werden.

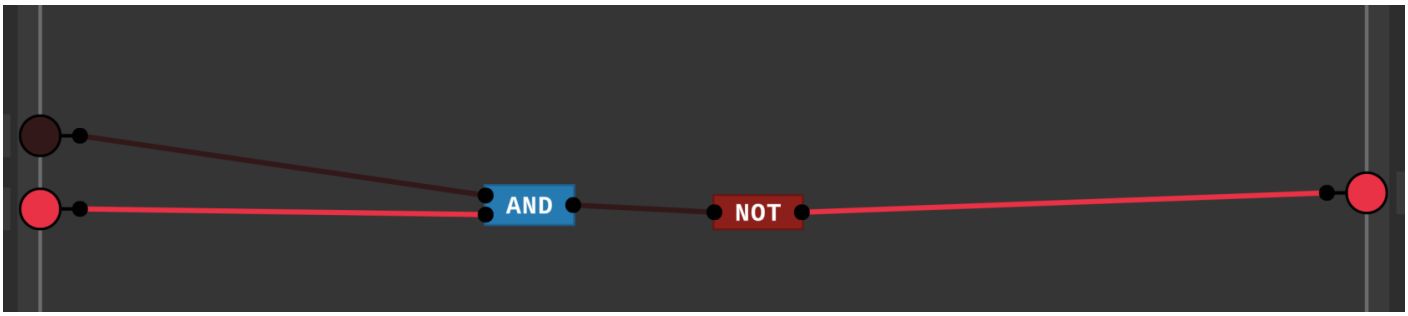
OR



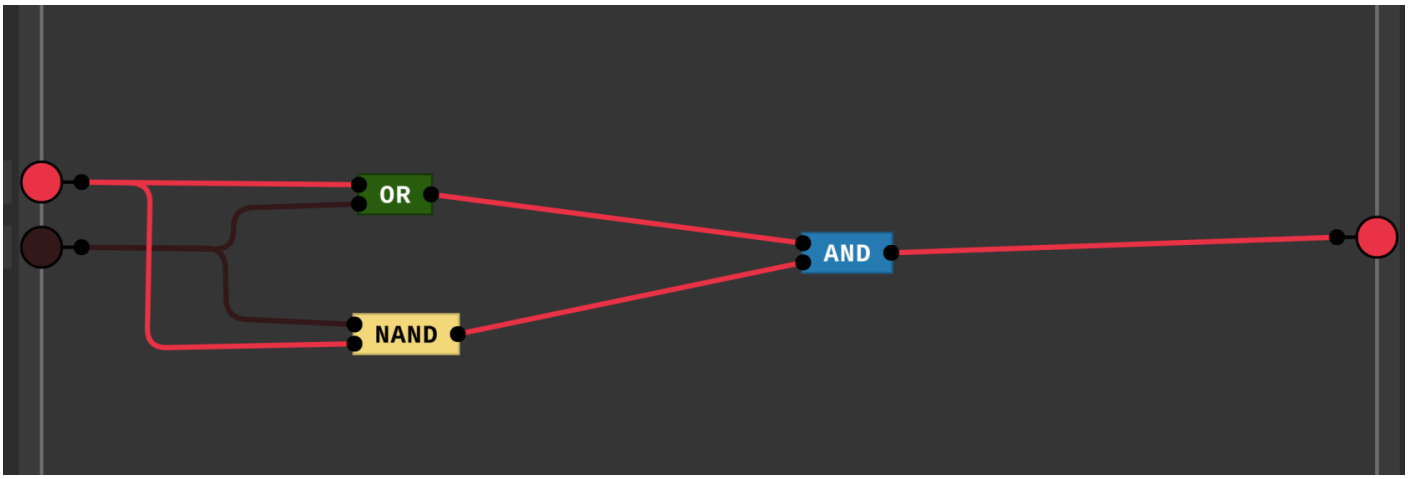
NOR



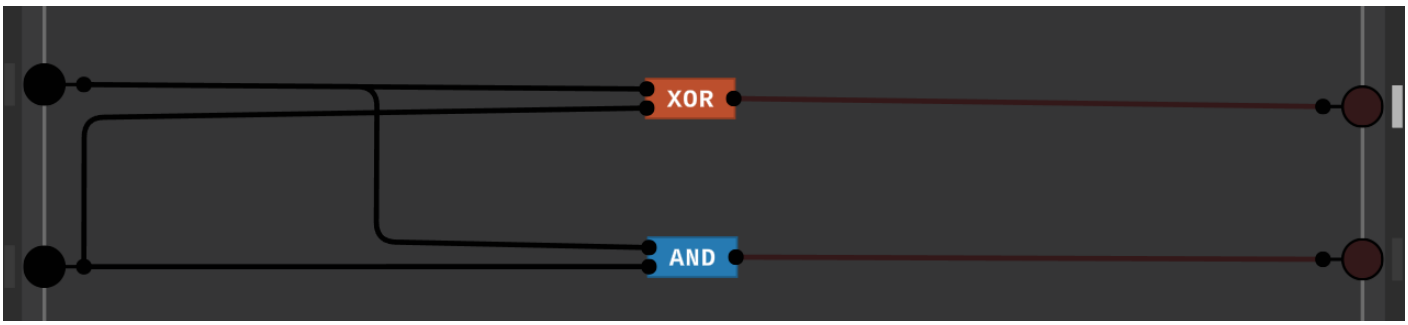
NAND



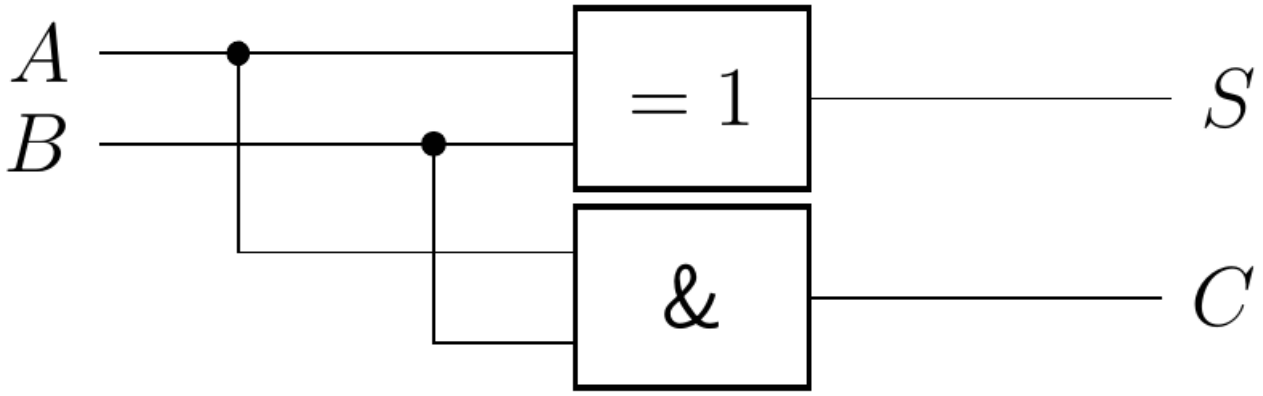
XOR



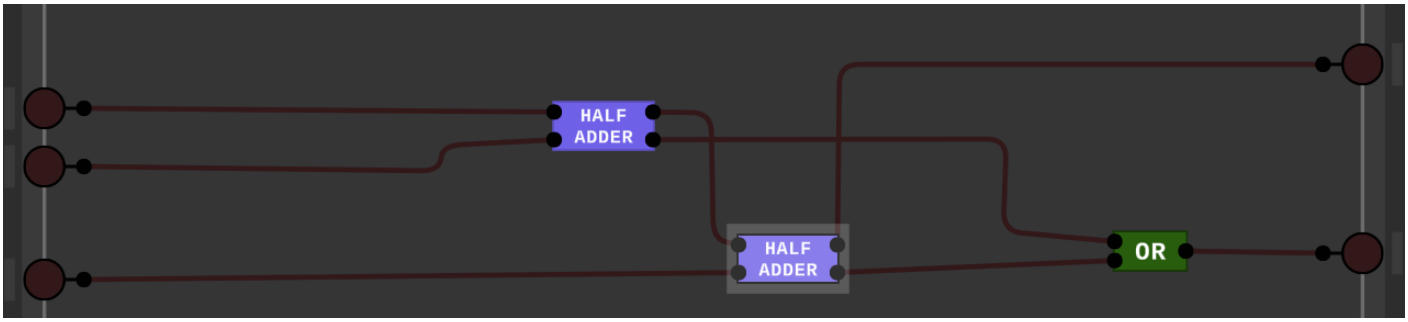
HALF ADDER



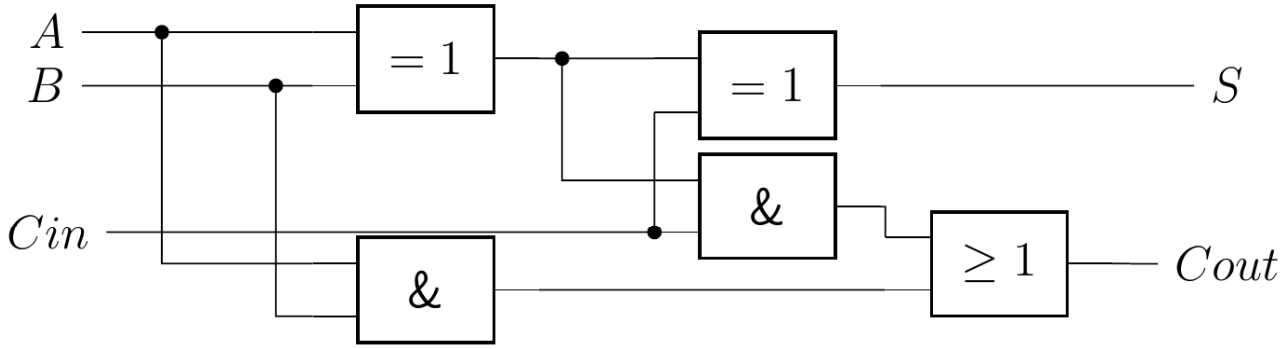
oder die Version mit Schaltsymbolen:



ADDER

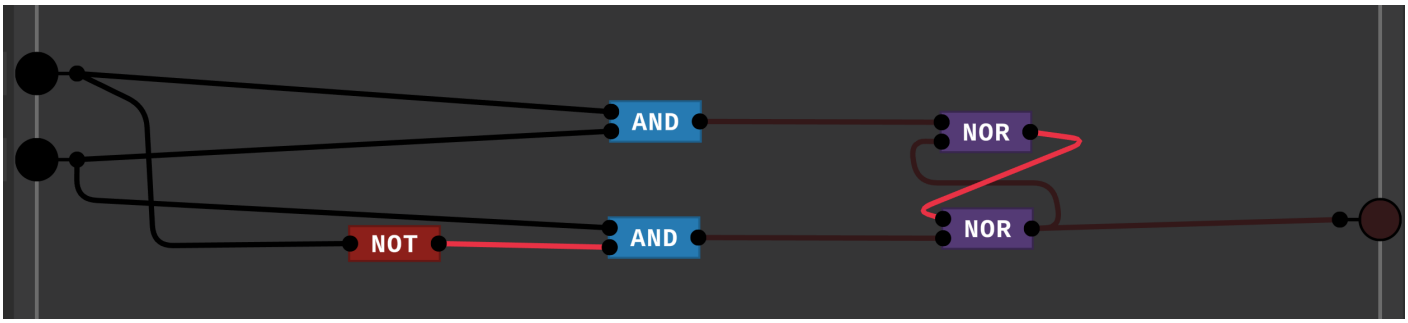


oder eine Version ohne den Halbaddierer:

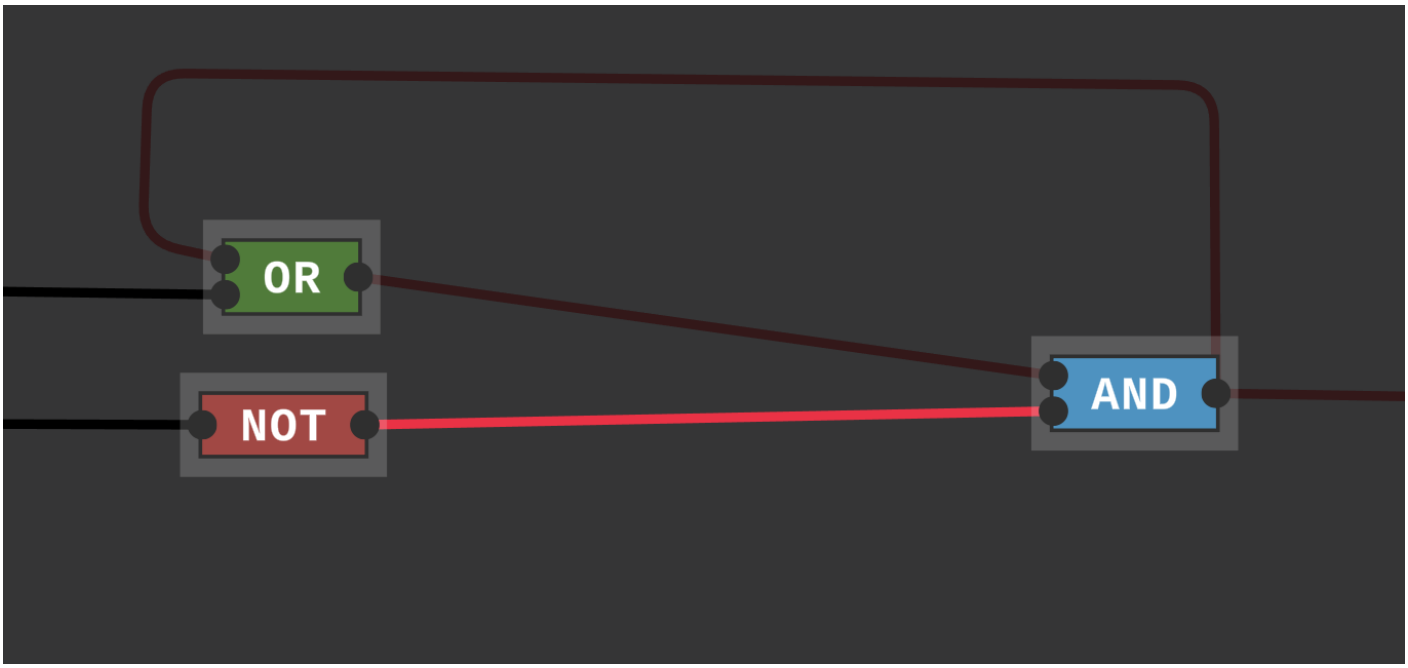


Simulation von Computerspeicher

D-Latch

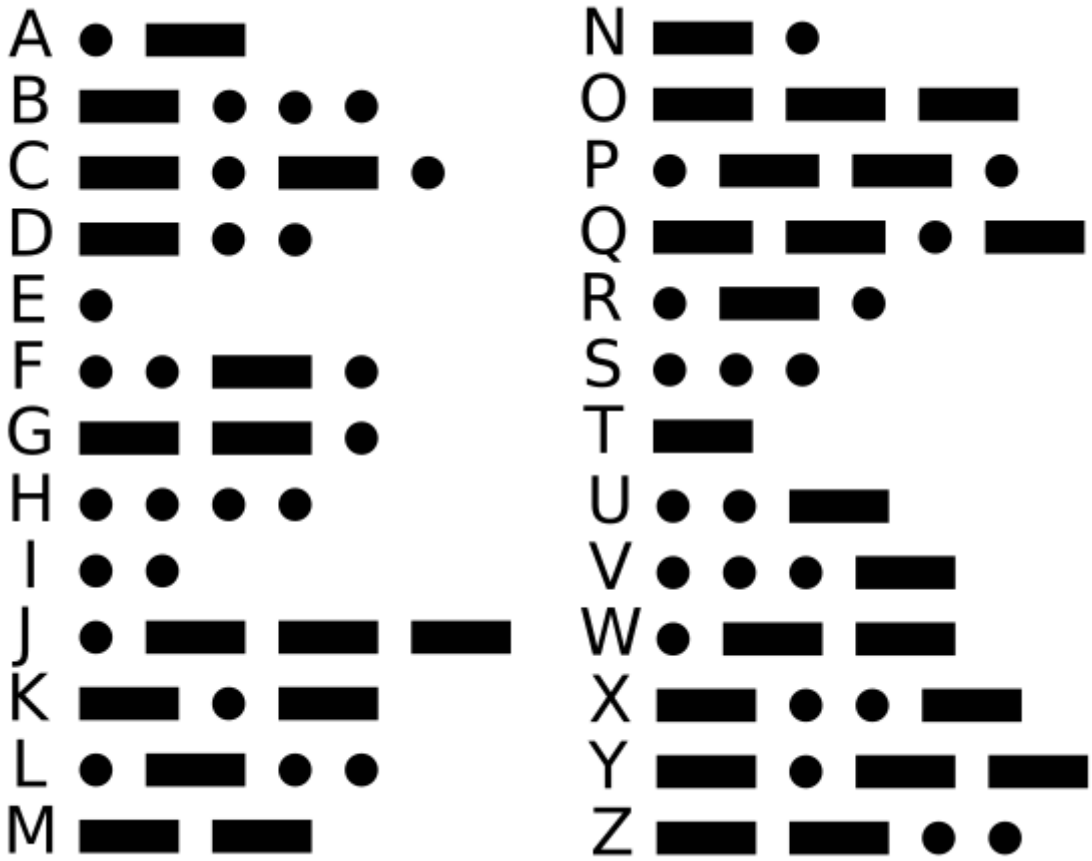


SR-Latch



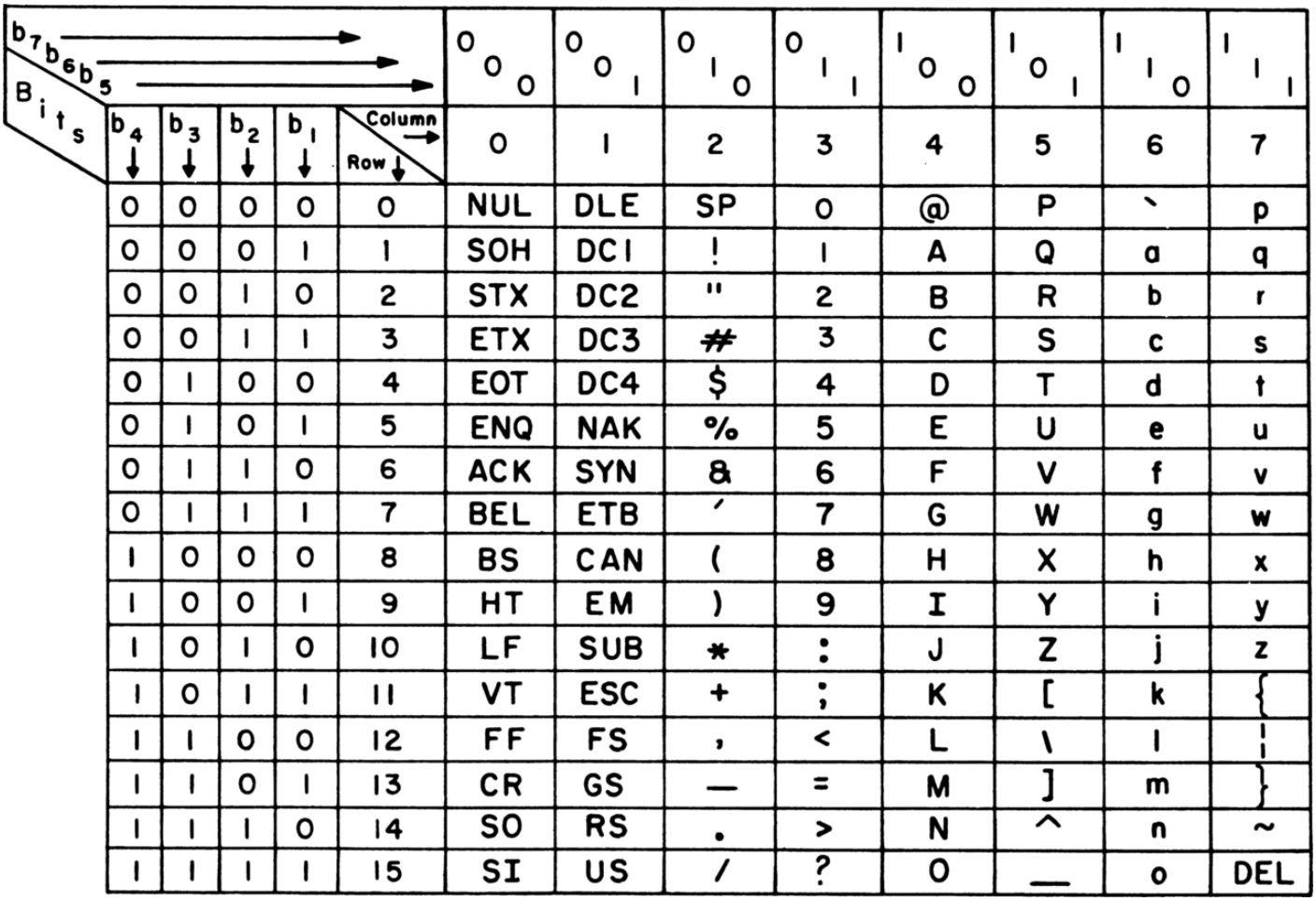
Kodierung von Buchstaben

Ein Vorläufer der heutigen Kodierung von Buchstaben ist das Morsealphabet.



So wie ein Computerchip, konnte man mit Morsegeräten nur zwischen einem kurzen und einem langen Signal unterscheiden. Jeder Buchstabe bekam also eine Kombination aus kurzen und langen Impulsen zugewiesen.

Für die Kodierung von Buchstaben und anderen Zeichen auf dem Computer wurde auch eine Kodierungstabelle entwickelt. Die erste Tabelle hier ASCII (American Standard Code for Information Interchange)



$b_7 b_6 b_5$ →	000	001	010	011	100	101	110	111
Bits b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Column → / Row ↓	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	¦
1	1	0	1	13	CR	GS	—	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Weitere Informationen

Programmierung

Programmierung

Scratch

Adresse zu Scratch

Eine Anmeldung ist nicht erforderlich, um mit Scratch zu programmieren. Die eigenen Programme können heruntergeladen und auch wieder hochgeladen werden. Damit ist eine Anmeldung auch nicht empfohlen.

Online Scratch Editor

Kurzanleitung

Das Prinzip von Scratch ist mit dem von Lego zu vergleichen. Verschiedene Bausteine der Programmierung können zusammengesteckt werden. Ob verschiedene Elemente zusammen passen wird durch die Form deutlich gemacht. Die Grundelemente von Programmiersprachen gibt es auch in Scratch. So gibt es Schleifen, Bedingungen, Variablen und auch Zufallszahlen. Besonders praktisch für die Einführung in die Robotik ist, dass es auch virtuelle Sensoren gibt. So kann z.b. der Bildschirmrand oder Hindernisse erkannt werden. Damit können wir viele Funktionen der Roboter hier virtuell darstellen.

Einführungskurs auf appcamps

Programmierung

Turtle-Grafiken mit Python

Um mit Python Turtle-Grafiken zu zeichnen, muss man nur die Bibliothek "turtle" importieren: `from turtle import *` Beispiel für eine Grafik:

```
from turtle import *
for i in range(0,4):
    forward(100)
    right(90)
```

Dies zeichnet ein Quadrat auf dem Bildschirm. Etwas interessanter ist die Spirale:

```
from turtle import *
for steps in range(100):
    for c in ('blue', 'red', 'green'):
        color(c)
        forward(steps)
        right(30)
```

Die offizielle Dokumentation der Turtle

Programmierung

Einführung in Python

Die Einführung findet ihr hier.

Programmierung einfacher Spiele

Wörterraten

Kopiere den Quellcode in Thonny und führe ihn aus. Bearbeite folgende Arbeitsaufträge:

1. Finde heraus, was du tun musst, um das Spiel zu spielen.
2. Schau dir den Quellcode an und vollziehe nach, wie das Programm abläuft.
3. Verändere das Programm nach deinen Vorstellungen.

```
# Quelle: https://www.geeksforgeeks.org/python-program-for-word-guessing-game/?ref=lbp
import random
# library that we use in order to choose
# random words from a list of words

name = input("What is your name? ")

# Here the user is asked to enter the name first

print("Good Luck ! ", name)

words = ['rainbow', 'computer', 'science', 'programming',
        'python', 'mathematics', 'player', 'condition',
        'reverse', 'water', 'board', 'geeks']

# Function will choose one random
# word from this list of words
word = random.choice(words)



print("Guess the characters")

guesses = ''
```

```
# any number of turns can be used here
turns = 12


while turns > 0:

    # counts the number of times a user fails
    failed = 0

    # all characters from the input
    # word taking one at a time.
    for char in word:

        # comparing that character with
        # the character in guesses
        if char in guesses:
            print(char, end=" ")

        else:
            print("_")

            # for every failure 1 will be
            # incremented in failure
            failed += 1

    if failed == 0:
        # user will win the game if failure is 0
        # and 'You Win' will be given as output
        print("You Win")

        # this print the correct word
        print("The word is: ", word)
        break

    # if user has input the wrong alphabet then
    # it will ask user to enter another alphabet
    print()
    guess = input("guess a character:")
```

```
    # every input character will be stored in guesses
    guesses += guess

    # check input with the character in word
    if guess not in word:

        turns -= 1

        # if the character doesn’t match the word
        # then “Wrong” will be given as output
        print("Wrong")

        # this will print the number of
        # turns left for the user
        print("You have", + turns, 'more guesses')

        if turns == 0:
            print("You Loose")
```

Erweiterungen

Wortliste in eigener Datei

```
# definiere die Datei, die du verwenden möchtest.
file = "wordlist.txt"
with open(file, "r") as f: # Öffne die Datei im Nur-Lesen-Modus
    wordlist = f.read().splitlines() # Teile die Datei zeilenweise.
print(wordlist)
```

Ausgabe der Wörter besser formatieren

Die Wörter werden in diesem Programm von oben nach unten geschrieben. Das liegt daran, dass der `print`-Befehl mit einem *carriage return* endet. Speichere die Läsungen zunächst in einer Variablen und gib diese am Schluss mit dem `print`-Befehl aus.

Anzahl Runden

Definiere die Anzahl der Fehlversuche in Abhängigigkeit der Wortlänge.

Speichern der Gewinner

```
from datetime import datetime
with open("winner.txt", "a") as text_file:
            text_file.write(f"{name} hat gewonnen. {datetime.now()}\r\n") # oder nur \n, falls
das nicht funktioniert.
```

Mastermind

Kopiere den Quellcode in Thonny und führe ihn aus. Bearbeite folgende Arbeitsaufträge:

1. Finde heraus, was du tun musst, um das Spiel zu spielen.
2. Schau dir den Quellcode an und vollziehe nach, wie das Programm abläuft.
3. Verändere das Programm nach deinen Vorstellungen.

```
# Quelle: https://www.geeksforgeeks.org/mastermind-game-using-python/?ref=lbp
import random



# the .randrange() function generates a
# random number within the specified range.
num = random.randrange(1000, 10000)

n = int(input("Guess the 4 digit number:"))

# condition to test equality of the
# guess made. Program terminates if true.
if (n == num):
    print("Great! You guessed the number in just 1 try! You're a Mastermind!")
else:
    # ctr variable initialized. It will keep count of
    # the number of tries the Player takes to guess the number.
    ctr = 0

    # while loop repeats as long as the
    # Player fails to guess the number correctly.
    while (n != num):
```

```
        # variable increments every time the loop
        # is executed, giving an idea of how many
        # guesses were made.
        ctr += 1

        count = 0

        # explicit type conversion of an integer to
        # a string in order to ease extraction of digits
        n = str(n)

        # explicit type conversion of a string to an integer
        num = str(num)

        # correct[] list stores digits which are correct
        correct = ['X']*4
        print(correct)
        # for loop runs 4 times since the number has 4 digits.
        for i in range(0, 4):

            # checking for equality of digits
            if (n[i] == num[i]):
                # number of digits guessed correctly increments
                count += 1
                # hence, the digit is stored in correct[].
                correct[i] = n[i]
            else:
                continue

        # when not all the digits are guessed correctly.
        if (count < 4) and (count != 0): #- this condition is not needed as we are starting with the condition, n!=num, which is, count<4
            print("Not quite the number. But you did get ",
                    count, " digit(s) correct!")
            # second code is not supposed to print the guessed numbers, from the sample output, here I get we are not recording the position of the guess,but count. But as per the explanation, the code should not print the guessed numbers, rather give their count.
                # print("Also these numbers in your input were correct.")
                # for k in correct:
```

```
                # print(k, end=' ')
            print('\n')
            print('\n')
            n = int(input("Enter your next choice of numbers: "))

        # when none of the digits are guessed correctly.
        elif (count == 0):
            print("None of the numbers in your input match.")
            n = int(input("Enter your next choice of numbers: "))

    # condition for equality.
    if n == num:
    # ctr must be incremented when the n==num gets executed as we have the other incrmentation
in the n!=num condition
        ctr+=1
        print("You've become a Mastermind!")
        print("It took you only", ctr, "tries.")
```

Siegerehrung

Die Siegerehrung kann in jedes Spiel eingebaut werden. Der Import muss natürlich an den Anfang der Datei.

```
import time
width=70
speed=0.015
for x in range(0,3):
    for i in range(width,0,-1):
        out = ""
        for j in range(width,i,-1):
            out += " "
        out += "You win"
        print(out)
        time.sleep(speed)
```

```
    for i in range(0,width):
        out = ""
        for j in range(i,width):
            out += " "
        out += "You win"
        print(out)
        time.sleep(speed)
```

Kryptologie

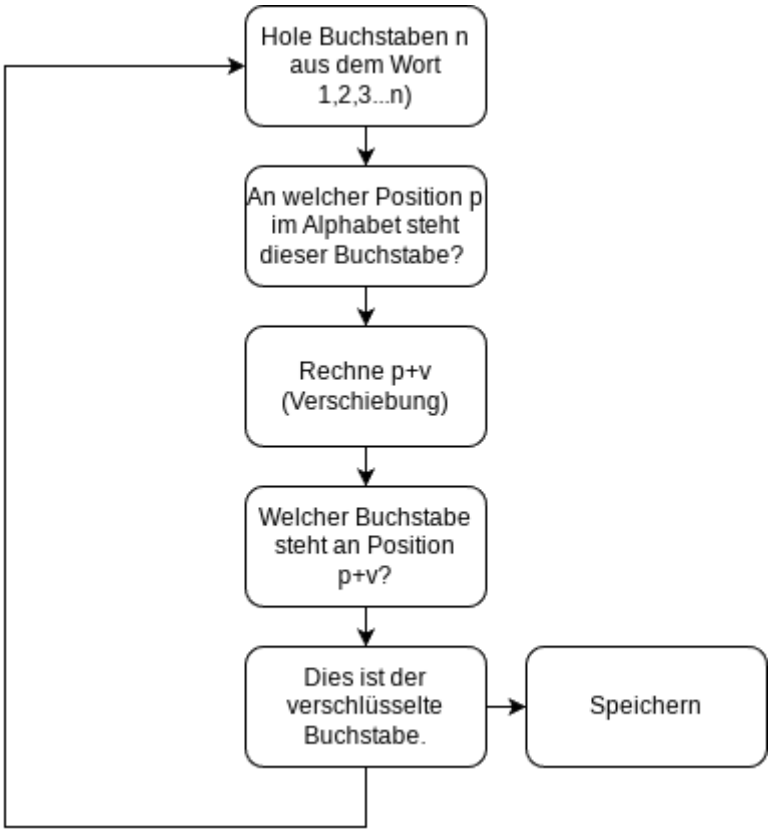
Kryptologie

Die CESAR-Verschlüsselung

Die Verschlüsselungsscheibe

CESAR-Verschlüsselung mit Python programmieren

Algorithmus für die Verschlüsselung mit CESAR



Für die Programmierung einer Ver- und Entschlüsselung mit der CAESAR-Verschlüsslung benötigt man folgende Elemente:

Beispiel für eine Modulorechnung

```
for i in range(100):
    x=i%5
    print(x)
```

Die Modulorechnung benötigt man für die Verschiebung, wenn die Rechnung p+v (siehe Algorithmus) über den letzten Buchstaben hinausgeht.

Funktionen zu Strings

- `lower()` gibt einen String in Kleinbuchstaben aus. Der entschlüsselte Text ist immer in Kleinbuchstaben.
- `len()` gibt die Länge eines Strings (oder einer Liste) zurück.
- `index()` gibt die Position des Zeichens in einem String zurück.

Iteration über einen String

Eine Iteration ist, wenn man in einem Algorithmus eine bestimmte Anweisung in einer Schleife immer wieder ausführt, wobei sich mindestens ein Parameter mit jedem Durchgang ändert. Dies ist häufig ein Zähler. In dem folgenden Beispiel wird mit jedem Durchgang der Schleife ein neuer Buchstabe aus dem String `s` geholt.

```
s = "Theodor-Heuss-Schule"
print(s.lower())
for c in s:
    a = s.index(c)
    print("Buchstabe ", c, " taucht erstmalig an Position ", a,  "auf.")

print(f"Der erste Buchstabe in {s}: {s[0]}")
for i in range(len(s)):
    print(f"Dies ist der {i+1}. Buchstabe: {s[i]}")
```

CESAR Verschlüsselung

Dies hier ist das Grundgerüst für ein Skript, das die CESAR-Verschlüsselung realisiert. Es muss nur noch ein wenig Code ergänzt werden. Natürlich gibt es auch andere Lösungen.

```
ALPHABET = "abcdefghijklmnopqrstuvwxyz"


def encrypt(text, key):
    encryptedText = ""
    text = text.lower()
    # Hier wird der Text verschlüsselt.
```

```
    return encryptedText.upper()

def decrypt(text, key):
    return ""

text = input("Den Text eingeben: ")
key = int(input("Den Schlüssel eingeben: "))
geheim = encrypt(text, key)
klar = decrypt(geheim, key)
print("Der verschlüsselte Text: ", geheim)

print("Der aus dem Geheimtext entschlüsselte Text: ", klar)
```

Bildverarbeitung

Das Portable Anymap Bildformat

Das Portable Anymap Bildformat gibt es in drei Versionen. Diese Bilder lassen sich mit einem Texteditor erstellen. Jedes Bildformat beginnt mit einer „magischen Zahl“ zur Identifizierung des Bildtyps, gefolgt von der Auflösung des Bildes und evtl. der Anzahl der Grau-, bzw. der Farbstufen. Die drei Bildarten lassen sich so definieren:

Schwarz-Weiß Bild

```
P1
11 7
1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0
```

Dieses Bild stell ein schwarzes T auf weißem Hintergrund dar.

Graustufenbild

```
P2
11 7
15
15 15 15 15 15 15 15 15 15 15 15
0 0 0 0 0 13 0 0 0 0 0
0 0 0 0 0 11 0 0 0 0 0
0 0 0 0 0 9 0 0 0 0 0
0 0 0 0 0 8 0 0 0 0 0
0 0 0 0 0 6 0 0 0 0 0
0 0 0 0 0 4 0 0 0 0 0
```

Dieses Bild stellt ein T auf weißem Hintergrund da, dessen Bein nach unten hin immer heller wird.

Farbbild

```
P3
11 7
255
0 0 0 0 0 0 0 0 0 0 0 0 0 0                              125 125 125     0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 0 250 100       255 255 255 255
255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 225 255 50      255 255 255 255
255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 30 65 128       255 255 255 255
255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 0 0 215         255 255 255 255
255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 200 100 200     255 255 255 255
255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 22 0 20         255 255 255 255
255 255 255 255 255 255 255 255 255 255 255
```

Dieses Bild stellt ein farbiges T dar.

Bildverarbeitung

Filter für das Portable Anymap Format mit python

```
from tkinter import *
from PIL import Image, ImageTk
# Ändere nichts oberhalb dieser Zeile



# Schreibe hier den Filter für P1 Dateien
def filter_p1(n):
    pass

# Schreibe hier den Filter für P2 Dateien
def filter_p2(n):
    pass

# Schreibe hier den Filter für P3 Dateien
def filter_p3(n):
    pass



# Ändere nichts unterhalb dieser Zeile.
new_image = ""
filename= input("Bitte Dateinamen angeben: ")
file = filename
newfile = file.split(".")[0]+ "_filtered." + file.split(".")[1]

with open(file, "r") as f:
    file = f.read()
    success = False
    if "P1" in file:
        success = True
        filetype = "P1"
    elif "P2" in file:
        success = True
```

```
    filetype = "P2"
elif "P3" in file:
    success = True
    filetype = "P3"
if not success:
    print("Could not determine filetype. Abort.")
    quit()
file_content = file.split("\n")
image_data = []
value = 0
for l in file_content:
    if not l.startswith("#"):
        match value:
            case 0:
                new_image = new_image + l +"\n"
                value += 1
            case 1:
                new_image = new_image + l +"\n"
                resolution = l
                value += 1
            case 2:
                if filetype != "P1":
                    new_image = new_image + l +"\n"
                    depth = l
                else:
                    image_data += filter(lambda x: x != "", l.split(" "))
                value += 1
            case _:
                image_data += filter(lambda x: x != "", l.split(" "))

#print(image_data)
columns = int(resolution.split(" ")[0])
if filetype == "P3":
    columns = columns *3
count = 1
for i in image_data:
    if filetype =="P1":
        if i != "":
            new_image = new_image + str(filter_p1(int(i))) + ""
    elif filetype == "P2":
```

```
            if i != "":
                if "\n" in i:
                    print("I: "+i)
                new_image = new_image + str(filter_p2(int(i))) + " "
        elif filetype == "P3":
            if i != "":
                new_image = new_image + str(filter_p3(int(i))) + " "
        if count%columns == 0:
            new_image = new_image +"\n"
        count += 1
    f.close()

with open(newfile, "w") as nf:
    nf.write(new_image)
    nf.close()

root = Tk()

# Read the Image
original_image = Image.open(filename)
filtered_image  = Image.open(newfile)
# Resize the image using resize() method
w = int(resolution.split(" ")[0])
h = int(resolution.split(" ")[1])
scale=10
if w < 300:
    original_image = original_image.resize((w*scale, h*scale))
    filtered_image = filtered_image.resize((w*scale, h*scale))

oimg = ImageTk.PhotoImage(original_image)
fimg = ImageTk.PhotoImage(filtered_image)
# create label and add resize image
label1 = Label(root, image=oimg)
label1.image = oimg
label1.pack()
label2 = Label(root, image=fimg)
label2.image = fimg
label2.pack()
# Execute Tkinter
root.mainloop()
```

```
print("Bye, bye")
```

Farbmischung

Farbpalettengenerator

Arbeiten mit Anwendungsprogrammen

Empfohlene Software

Im Unterricht benutzte Software

Thonny

Installiere auch die Erweiterung "notebook".

Das Notebook wird mit dem Befehl !jupyter notebook gestartet.

Digital Logic Sim

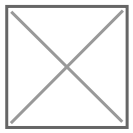
Tabellenkalkulation

Enführung in die Tabellenkalkulation

Formeln kopieren

Sind Formeln lang und benötigt man gleiche oder ähnliche Formeln in vielen Feldern der Tabelle, kann mein eine Formel kopieren und an anderer Stelle wieder einfügen. Wenn du alle Tricks zum Kopieren und Einfügen kennst, kannst du selbst große Tabellen in kürzester Zeit mit Tausenden korrekter Formeln füllen.

Also: Es lohnt sich!



Die Ausgangslage: Es sollen mehrere Potenzen der Basis 2 berechnet werden. Die Formel für die nullte Potenz von 2 steht schon im Feld D3 und soll in die Felder D4 bis D10 kopiert werden.

Rezept zum Kopieren und Einfügen

1. Klicke mit der Maus auf das Feld D3.
2. Halte auf der Tastatur die Taste
   |Strg| gedrückt und drücke dann die Taste |c|.
3. Klicke mit der Maus auf das Feld D4 und ...
4. ziehe bei gedrückter Maustaste den Zeiger bis zum Feld D10.
5. Halte auf der Tastatur die Taste
   Strg
   gedrückt und drücke dann die Taste
   v
   .

Besonders nützlich ist, dass die Formeln beim Kopieren automatisch angepasst werden. Das Programm „ahnt“ nämlich, dass du die Potenzen zeilenweise mit den Zahlen der jeweiligen Zeile berechnet möchtest und ändert die Ausgangsformel automatisch entsprechend ab.
Im Bild siehst du, dass an Stelle der kopierten Formel

=B2^C2

in das Feld D7 die automatisch angepasste Formel

=B7^C7

eingetragen wurde.

Erläuterung und Hinweise

- Mit
  Strg
  +
  c
  kopiert man den Inhalt eines oder mehrerer aktivierter Felder in eine unsichtbare Zwischenablage.
- Zieht man mit der Maus bei gedrückter Maustaste über Felder hinweg, wird optisch ein Rechteck aufgezogen. Danach sind alle Felder markiert, die das Rechteck berühren oder darin liegen.
- Mit
  Strg
  +
  v
  fügt man den Inhalt der Zwischenablage in alle Felder ein.

Anpassung von Formeln beim Kopieren verhindern

In vielen Fällen ist die automatische Formelanpassung nicht erwünscht. Stelle dir vor, dass man die Basis 2 nicht in jeder Zeile, sondern nur einmal angegeben möchte. Dadurch würde man nämlich Tipparbeit sparen, weil man die Basis nicht in jede Zeile eintragen müsste.

Man würde folgendermaßen beginnen und dann die Formel aus D3 in die darunter liegenden Felder kopieren:

Kopiert man die Formel aus Feld D3 in die Felder D4 bis D10, erhält man nicht das erwartete Ergebnis, weil das Feld der Basis 2 so wie im obigen Beispiel verändert wird. Wenn in einem Feld kein Eintrag steht, geht die Software einfach von der Zahl 0 aus. Wenn man 0 potenziert, erhält man auch 0 als Ergebnis.

Man muss also verhindern, dass die Koordinate für Zeile 3 beim Kopieren verändert wird. Dies erreicht man, indem man vor die 3 bei den Koordinaten B3 in der Formel

=B3^C3

ein Dollarzeichen setzt. Man schreibt also

=B$3^C3

. Das Dollarzeichen sorgt dafür, dass die 3 beim Kopieren auch eine 3 bleibt.

Soll beim Kopieren in eine andere Spalte der Buchstabe der Spalte unverändert bleiben, setzt man das Dollarzeichen vor den Buchstaben der beim Kopieren unverändert bleiben soll.