11. Gegeben ist der in Python codierte Quelltext *pumuckl.py*:

```
import os
i = int(input('i = '))
j = 0
while True:
    i = i // 2
    k = i + 1
    j += 1
    print('i =',i)
    print('k =',k)
    print()
    if k == i: break
print('j =', j)
os.system('pause')
```

Bemerkung: Hier wird gewissermaßen eine repeat-Schleife (fußgesteuerte Schleife) in Form einer while-Schleife (kopfgesteuerte Schleife) simuliert; die repeat-Schleife gibt es in Python nicht, im Gegensatz zu anderen Sprachen wie Pascal („repeat . . until") oder Java, C++ („do . . while").

Wie verhält sich das Programm? Bestätige die Vermutung mittels Testläufen.

12. Die **Goldbachsche Vermutung** lautet:

„Jede gerade natürliche Zahl, die größer als 2 ist, läßt sich als Summe zweier Primzahlen darstellen."

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

Definiere die Goldbach-Eigenschaft **G** wie folgt:
Eine gerade Zahl **n** ∈ **N** (**N** = Menge der natürlichen Zahlen) hat die Goldbacheigenschaft **G** genau dann, wenn es Primzahlen **p** und **q** gibt mit **n = p + q**.
Unter der Menge **M** verstehen wir diejenige Teilmenge der geraden natürlichen Zahlen, deren Elemente die Goldbacheigenschaft **G** haben.

a) Erstelle einen in Python codierten Algorithmus, der nach Eingabe einer geraden natürlichen Zahl **n** entscheidet, ob **n** die Goldbacheigenschaft hat.

Hinweis: Verwende in geeigneter Weise die Boolesche Funktion **`prim`** wie in dem Algorithmus ***Primzahltest***:

```
# Primzahltest
# Nach Eingabe einer natuerlichen Zahl n, n>1,
# entscheidet dieser Algorithmus, welche Zahlen
# aus der Menge {2, 3, . . ,n} Primzahlen sind.

while True:
  try:    n = int(input('n = '))
  except:
    print('Gib eine natuerliche Zahl n mit n>1 ein!')
    continue
  if n <= 1:
    print('Gib eine natuerliche Zahl n mit n>1 ein!')
    continue
  break

def prim(x):
  if x == 2: return True
  i = 2
  while i <= x//2:
      if x % i == 0: return False
      i += 1
  return True

for m in range(2,n+1):
  if prim(m): print(m,' ist Primzahl')
  else:       print(m,' ist keine Primzahl')
```

b) Begründe, daß die Menge
M = {**n** ∈ **N** | **n** ist gerade, und **n** hat die Eigenschaft **G**}
entscheidbar ist!
*Bemerkung: Es ist bis heute nicht entschieden, ob **jede** gerade natürliche Zahl **n** mit **n ≥ 4** die Goldbacheigenschaft hat, somit sich als Summe zweier Primzahlen darstellen läßt.*

13. **Collatz-Problem**
https://de.wikipedia.org/wiki/Collatz-Problem

Gegeben sei eine positive natürliche Zahl **n**.

Bilde die Collatz-Zahlenfolge **{a_i}**, i ∈ {0, 1, 2, 3, ... }, gemäß folgenden Regeln:

(1) **a_0 = n** (**n** ist Startwert der Folge)

(2) **a_{i+1} = a_i / 2**, falls **a_i** gerade

(3) **a_{i+1} = 3a_i + 1**, falls **a_i** ungerade

Beispiel: für n = 17 erhält man die Folge

17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 4, 2, 1, 4, 2, 1, ...

Eine positive natürliche Zahl n nennt man auch wundersam, wenn die zugehörige Collatzfolge nach endlich vielen Schritten den Wert 1 erreicht. Folglich ist Zahl 17 wundersam!

Lothar Collatz formulierte 1937 folgende Vermutung:

Für jede positive natürliche Zahl n mündet die Collatz-Folge {a_i} in den Zyklus 4, 2, 1.

Bis heute ist diese Vermutung weder bewiesen noch widerlegt.

Stand Juli 2020: Alle positiven ganzen Zahlen bis 2^{68} (ca. $2{,}95 \times 10^{20}$) als Startwerte bestätigen die Vermutung.

Aufgabe: Erstelle einen in Python codierten Algorithmus, der nach Eingabe einer natürlichen Zahl n, n > 0, die Collatz-Folge berechnet und ausgibt. Ermittle auch die Länge der Collatz-Folge (Anzahl der Folgenglieder, bis die Folge zum ersten Mal den Wert 1 annimmt).