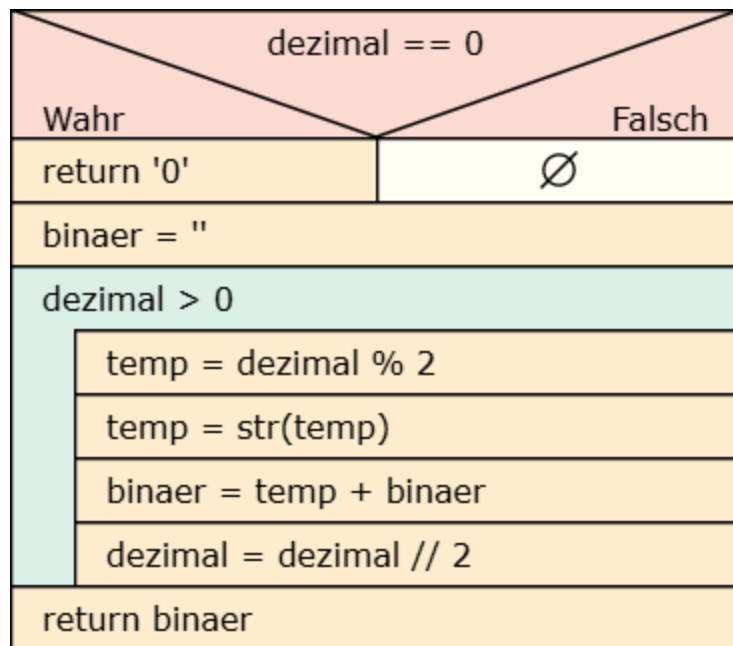


Kapitel 6: Vertiefung

6.1. Dezimal- zu Binärzahl

Die Umrechnung von Dezimalzahlen und Binärzahlen ist im folgenden Struktogramm dargestellt.



Implementiere eine Funktion mit dem Namen `dez2bin` .

Übergeben wird der Parameter `dezimal` - also die umzurechnende Dezimalzahl. Teste deine Funktion!

Beispiele:

$$2_{10} = 10_2$$

$$7_{10} = 111_2$$

$$8_{10} = 1000_2$$

Eine tiefgestellte 10 bedeutet "zur Basis 10" - also eine Dezimalzahl. Bei einer Binärzahl wird dies mit einer tiefgestellten 2 dargestellt.

```
In [ ]: # HIER IST PLATZ FÜR DEINE LÖSUNG! :)
```

```
In [1]: # MUSTERLÖSUNG
```

```
def dez2bin(dezimal):
```

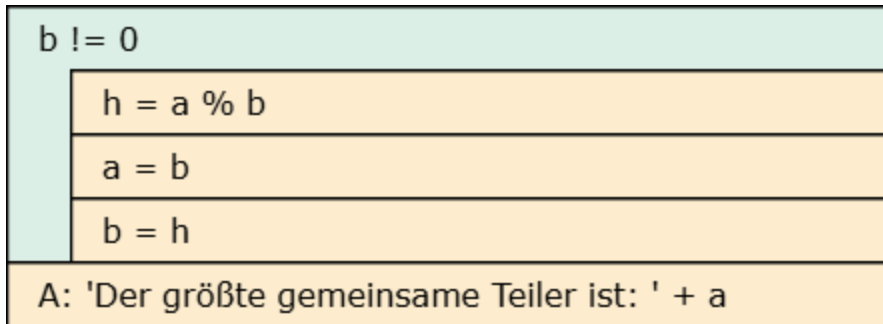
```
if dezimal == 0:
    return '0'
binaer = ''
while dezimal > 0:
    temp = dezimal % 2
    temp = str(temp)
    binaer = temp + binaer
    dezimal = dezimal // 2
return binaer

print(f'2 in binär: {dez2bin(2)}')
print(f'7 in binär: {dez2bin(7)}')
print(f'8 in binär: {dez2bin(8)}')
```

```
2 in binär: 10
7 in binär: 111
8 in binär: 1000
```

6.2. Euklidischer Algorithmus bzw. größter gemeinsamer Teiler

Das folgende Struktogramm zeigt den euklidischen Algorithmus, welcher in der Lage ist, den größten gemeinsamen Teiler (kurz: ggT) zweier ganzer Zahlen zu bestimmen.



Implementiere eine Funktion mit dem Namen `ggT`. Übergeben werden zwei Parameter `a` und `b`. Teste deine Funktion.

Beispiele:

$ggT(2, 4) \rightarrow 2$

$ggT(10, 99) \rightarrow 1$

$ggT(13, 104) \rightarrow 13$

```
In [ ]: # HIER IST PLATZ FÜR DEINE LÖSUNG! :)
```

```
In [ ]: # MUSTERLÖSUNG

def ggT(a, b):
    eingabe_a = a # nur für die schöne Ausgabe, nicht Teil des Struktogramms
    eingabe_b = b # nur für die schöne Ausgabe, nicht Teil des Struktogramms
    while b != 0:
        h = a % b
        a = b
        b = h
    print(f'Der größte gemeinsame Teiler von {eingabe_a} und {eingabe_b} ist: {a}')

a = int(input('Erste Zahl: '))
b = int(input('Zweite Zahl: '))

ggT(a, b)
```

Der größte gemeinsame Teiler von 13 und 104 ist: 13

6.3. Binominalkoeffizient

Implementiere ein Funktion zur Berechnung des Binomialkoeffizienten.

$$\binom{n}{k} = \frac{n!}{k! \cdot (n - k)!}$$

Hinweis: Um das Problem sinnvoll und effizient zu lösen, empfiehlt sich das Implementieren einer zusätzlichen "Hilfsfunktion", welche die Fakultät einer Zahl berechnet.

Anmerkung: Dein Programm soll dabei außerdem beachten, dass gilt: $n \geq k$ und dass n sowie k natürliche Zahlen sind. Wird eine dieser Bedingungen verletzt, soll ein entsprechender Hinweis ausgegeben werden und die Funktion `None` zurückgeben. Teste deine Funktion!

Beispiele:

- $\binom{1}{1} = 1$
- $\binom{4}{2} = 6$
- $\binom{24}{3} = 2024$

```
In [ ]: # HIER IST PLATZ FÜR DEINE LÖSUNG! :)
```

```
In [4]: # MUSTERLÖSUNG
```

```
def fakultaet(z):
    erg = 1
    for i in range(1, z + 1):
        erg = erg * i
    return erg

def binom(n, k):
    if n < k or n < 0 or k < 0:
        print('Fehler: n < k oder n < 0 oder k < 0')
        return None
    return fakultaet(n) / (fakultaet(k) * fakultaet(n - k))

n = int(input('Gib n ein: '))
k = int(input('Gib k ein: '))
print(f'Binomialkoeffizient von {n} über {k}: {binom(n, k)}')
```

Binomialkoeffizient von 4 über 2: 6.0

6.4. Zahlenfolge

Implementiere ein Programm, welches die nachfolgende Zahlenfolge ausgibt.

-30 -27 -24 -21 -18 -15 -12 -9 -6 -3 0 3 6 9 12 15 18 21 24 27 30

Anmerkung: Nutze dabei **keine** Zählschleife. Kopf- und fußgesteuerte Schleife sind okay!

```
In [ ]: # HIER IST PLATZ FÜR DEINE LÖSUNG! :)
```

```
In [ ]: # MUSTERLÖSUNG

def zahlenfolge(start, ende, schritt):
    ausgabe = ''
    zahl = start
    while zahl <= ende:
        ausgabe = ausgabe + str(zahl) + ' '
        zahl = zahl + schritt
    return ausgabe

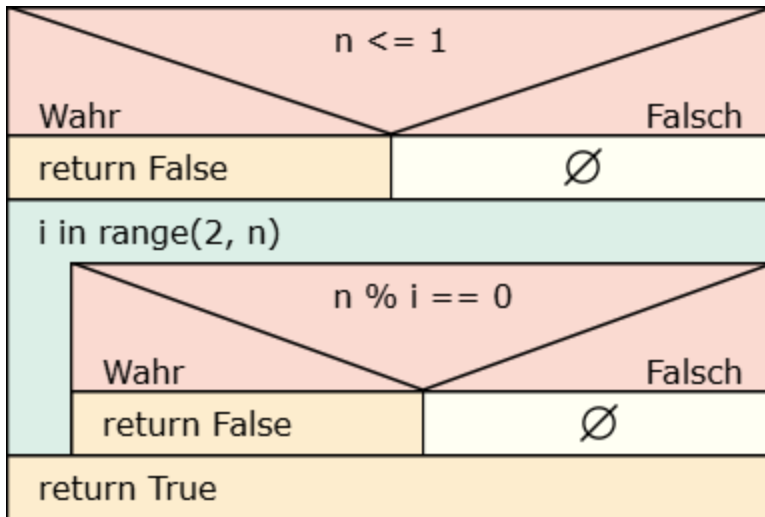
print(zahlenfolge(-30, 30, 3))
```

6.5. Primzahlen

Implementiere ein Programm, welches in der Lage ist zu überprüfen, ob eine eingegebene Zahl n eine Primzahl ist.

Implementiere dazu eine Funktion, der die Zahl n als Parameter übergeben wird. Der Nutzer soll vor dem Funktionsaufruf aufgefordert werden diese Zahl einzugeben.

Orientiere dich am folgenden Struktogramm.



Beispiele:

- `ist_primzahl(4) = False`
- `ist_primzahl(7) = True`

```
In [ ]: # HIER IST PLATZ FÜR DEINE LÖSUNG! :)
```

```
In [6]: # MUSTERLÖSUNG
```

```
def ist_primzahl(n):
    if n <= 1:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True

eingabe = int(input('Gib eine Zahl ein: '))
print(f'ist_primzahl({eingabe}) = {ist_primzahl(eingabe)}')
```

```
ist_primzahl(7) = True
```

