

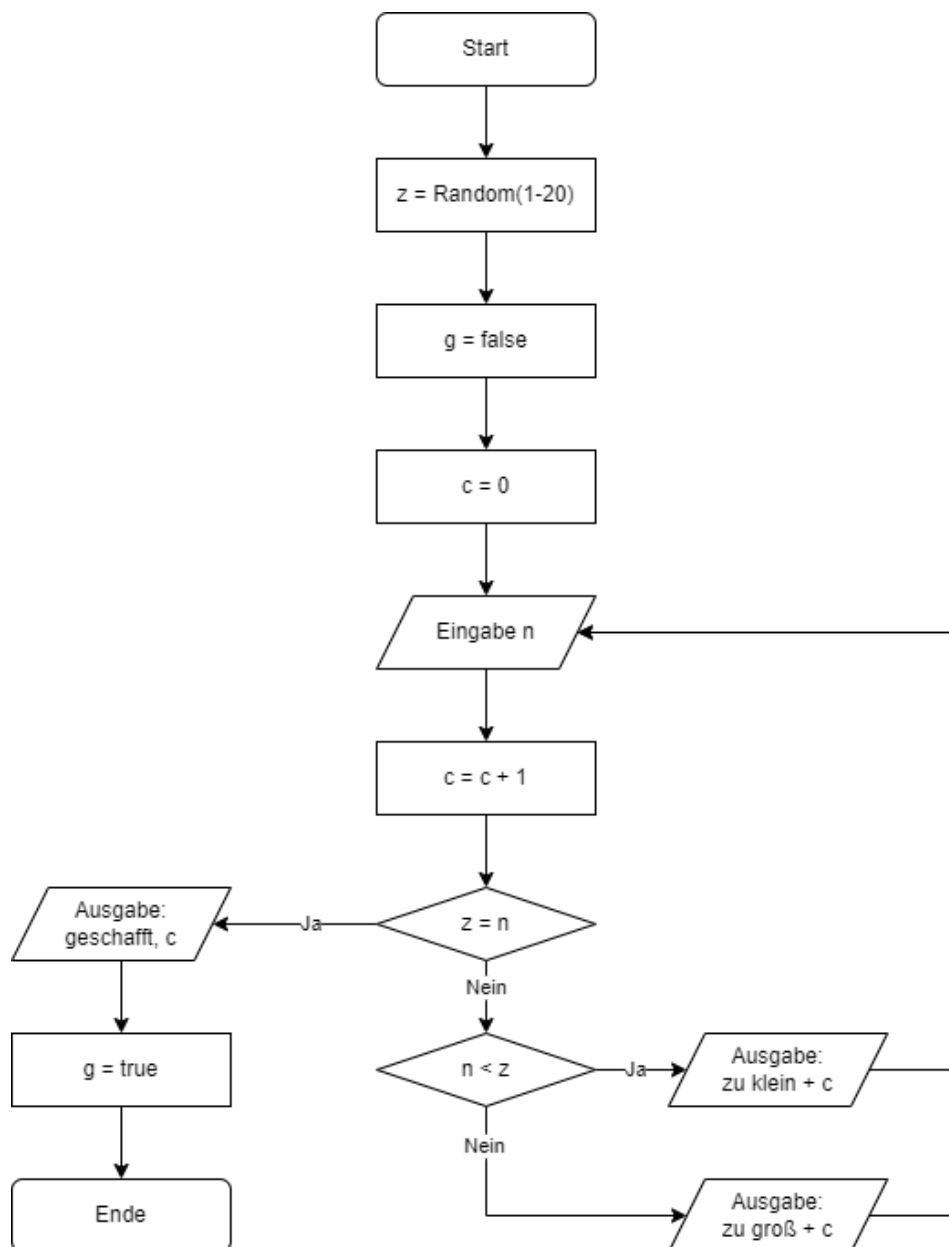
Programmbeschreibung und Flussdiagramm

**Hinweis**

Dieses Programm „denkt“ sich eine zufällige Zahl zwischen 1 und 20 aus. Der Nutzer soll diese Zahl erraten. Das Programm gibt die Anzahl der Versuche wieder.

Die Umsetzung des Programm verwendet:

- while-Schleife
- if-Abfrage
- Variablen
- Boolean (wahr/falsch)
- Inkrementieren



Schritt-für-Schritt-Anleitung

In dem folgenden Teil führt „Schritt-für-Schritt“ durch das Flussdiagramm.

Am Ende dieser Anleitung steht das vollständige Programm passend zum Flussdiagramm.

1. Schritt: Die Eingabe (Zeile 1 und 5)

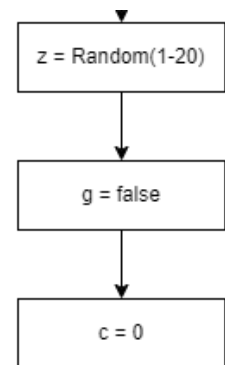
Die Variablen `z`, `g` und `c` werden deklariert.

Die Variable `c` dient zum „Hochzählen“, um die Länge der Reihe einzuhalten.

Mit „`from random import`“ wird die Bibliothek für die Zufallszahl geladen.

Python

```
1 from random import *
2
3 z = randint(1, 20)
4 g = False
5 c = 0
```



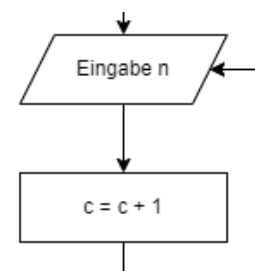
2. Schritt: Die while-Schleife (Zeile 5 bis 7)

Solange **g nicht wahr** ist, die Frage nach der richtigen Zahl gestellt.

Die Variable **a** wird mit dem Wert aus "**prompt**" deklariert. Der Zähler `c` erhöht sich um eine Einheit.

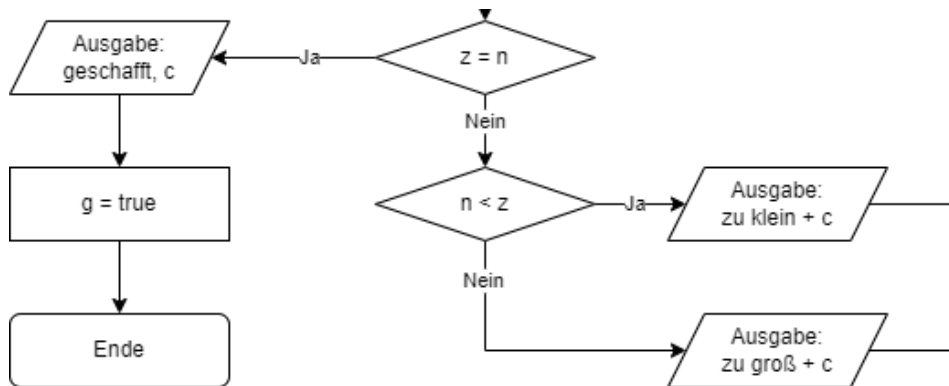
Python

```
1 from random import *
2
3 z = randint(1, 20)
4 g = False
5 c = 0
6
7 while g == False:
8     n = int(input("Gebe die Zahl an: "))
9     c += 1
```



3. Schritt: Die Abzweigungen (Zeile 10 bis 16)

- Ist $z = n$, dann wird g auf wahr gestellt. Zuvor gibt es die Ausgabe „geschafft“ mit dem Zählerwert.
- Ist $n < z$, dann wird die Ausgabe „zu klein“ mit dem Zähler ausgegeben.
- Ist $n > z$, dann wird die Ausgabe „zu groß“ mit dem Zähler ausgegeben.



Python

```
1 from random import *
2
3 z = randint(1, 20)
4 g = False
5 c = 0
6
7 while g == False:
8     n = int(input("Gebe die Zahl an: "))
9     c += 1
10    if z == n:
11        print('geschafft ', str(c))
12        g = True
13    elif n < z:
14        print('zu klein ', str(c))
15    elif n > z:
16        print('zu groß ', str(c))
```

ENDE

Aufgaben

- ① „Baue“ das Programm mit Hilfe der Schritt-für-Schritt-Anleitung nach.
Füge in dem Quellcode passend zu den Schritten Kommentare ein, die den jeweiligen Teil in seiner Funktion beschreiben.

Python

```
1 # Ein "Hashtag" ergibt einen Zeilenkommentar
2
3 """Drei Anführungszeichen am Anfang und Ende ergeben
4 einen Kommentar auf mehreren Zeilen"""
```

- ② Überlege Dir mögliche Verbesserungen für das Programm. Beispielsweise:
- Könnte der Text der Eingabe und der Ausgabe verbessert werden.
 - Lässt sich der Schwierigkeitsgrad erhöhen?
 - Könnte die letzte Else-If-Abfrage durch Else ersetzt werden?

[illegible]

- ③ Passe das Flussdiagramm und den Quelltext entsprechend Deiner Überlegungen aus Aufgabe 2 an.