Probe-Klausur-Python 1

1. Was ist eine Programmiersprache?
   - ☐ Sprache, die man im Alltag spricht.
   - ☐ Sprache von Menschen erstellt, um ein Programm zu schreiben.
   - ☐ Sprache, die Geheimnis Dienst nutzt
   - ☐ Sprache, die Arbeit organisiert.

2. Welches Handeln **nicht** Teil eines Computer-Algorithmus sind:
   - ☐ Eingabe und Ausgabe
   - ☐ mathematische Anweisung
   - ☐ Bedingte Ausführung und Wiederholung
   - ☐ Überlegung und Kritik auszuüben

3. Wie kann Python arithmetische Operationen durchführen? Was ist das Ergebnis des folgenden Ausdrucks?

   2+(6+2)/(2**2)=

   × 2.5 × 4 × 6 × 5

4. Was ist **nicht** Werte-Type von Python**?**

   × *Int* und *float* Zahlen × Liste × Boolean Werte und None × String

5. Was ist die Typen von den folgende Variable (x):

```
x='3'
print(type(x))
```

   × Int × float × str × boolean

6. Was ist die Werte von x in das folgende Programm:

```
x=3
y=4
if x==y:
    print(x)
```

   × 3 × 4 × 1 × 7

7. Sei a=50 und b=25. Was sind die Werte für die folgende logische Operationen:

   a>40 and b>40
   a>0 and b>0
   a>0 or b<0
   b<0 or b<0

   - ☐ True, True, False, False
   - ☐ False, True, False, True
   - ☐ False, True, True, False
   - ☐ False, False, True, True

8. Warum nutzt man in dem folgenden Programm die Funktion "float()"?

```
import math
r=float(input('radius'))
V=(3/4)*math.pi*r**3
print(V)
```

- ☐ Weil Variable *r* ein Integer ist.
- ☐ Weil wir einen Text mit dem Ergebnis schreiben wollen.
- ☐ Weil *Input()* die Eingabe als eine String Werte zurückgibt.
- ☐ Weil wir Variable *r* mit einer *float* Wert multiplizieren.

9. Welche Ausgabe erzeugt das Programm?

```
a=2
a+=1
print(a)
```

× 2 × 1 × 3 × 4

10. Welche Ausgabe erzeugt das Programm?

```
x = ''
for i in range(10 - 6):
    x += str(i)
print(x)
```

× '0123' × '4444' × 'iiii' × 6

11. Welche Ausgabe erzeugt das Programm?

```
i=1
while i<3:
  i+=1
print(i)
```

× 1 × 2 × 3 × 4

12. Was sind die Ergebnissen von den folgenden Ausdrücke?

- 1+1.0
- 1+'1'
- '1'+'1'

× 2, 11, 2 × 2.0, 'SyntaxError', '11' × 1, 1, '2' × 2, 'SyntaxError', 2

13. Warum gibt es 'SyntaxError' bei dem Programm:

```
def countdown(n)
while n>0:
print(n)
n=n-1
print('Bumm!')
```

- ☐ Weil das Programm keine Indentation hat.
- ☐ Weil die Funktion print ein Klammer fehlt.
- ☐ Weil Variable n ein String ist.
- ☐ Weil *while* in einem unendlichen Loop ist.

14. Was ist die Ausgabe bei dem folgenden Programm?

```
my_list = ['x', 'y', 'z']
my_list.append('a')
empty_string = ''
for i in my_list:
        empty_string += i
print(empty_string)
```

× 'xyz' × 'a' × 'xyza' × ' '

15. Welche Ausgabe erzeugt das Programm?

```
print(type('a') == str and 3.5 > 1)
```

× False × SyntaxError × 'a' and 3.5 ×True

16. Das folgende Programm soll den lexikographisch größten Buchstaben eines zu übergebenen Worts zurückgeben. Warum funktioniert das folgende Programm nicht?

```
word = int(input('Welches Wort soll geprüft werden? '))
def greatestChar(long_word):
        maxChar = 2.5
        for c in long_word:
        if c > maxChar:
        c = maxChar
        else:
        break
 greatest_char = greatestChar(word)
```

- ☐ weil das maxChar ein Float Werte ist.
- ☐ weil weil die For-Schleife Tot (dead) ist.
- ☐ weil das variable *word* eine String Werte sein sollte.
- ☐ weil das variable *word* eine Float Werte sein sollte.

17. Was macht die folgende Funktion?

```
wort=input('Wort')
wort=wort.lower()
i=0
for zeichen in wort:
    if zeichen in [‘a’,’e’,’i’,’o’,’u’]:
        i+=1
print(i)
```

- ☐ Das Programm sagt, wie lang das Wort ist.
- ☐ Das Programm sagt, wie viele Buchstaben klein geschrieben sind.
- ☐ Das Programm sagt, wie viele Vokale das Wort hat.
- ☐ Das Programm drückt die Zeichen in die Liste.

18. Gegebenen die List, List=[['python', 'refugeeks','Monty'],[25,63,41]]
Welches Element ruft die Anweisung List[1][1]?

× 'Python' × 'refugeeks' × 25 ×63

19. In welchem Fall funktioniert die for-Schleife nicht:

- ☐ for z in word:
- ☐ for z in ['a','b','c']:
- ☐ for z in 4:
- ☐ for z in range(4):

20. Welche Ergebnis drückt die Funktion bei der Eingabe **20** .

```
def bmi_interpreter(b):

    if b < 17.5:
        print(b,' : Untergewicht')
    elif 18.5<b<25:
        print(b,' : normal')
    else:
        print(b, ' : Übergewicht')
```

- ☐ 20 : normal
- ☐ 20 : Untergewicht
- ☐ 78.4 : Übergewicht
- ☐ 5.1 : Untergewicht