Aufgaben

1. Aufgabe

Schreibe ein Programm, das für eine zufällig, i.e. via randint, bestimmte Zahl zwischen -10 und +10 den Absolutwert ausgibt. Den Absolutwert erhältst Du, wenn Du das Vorzeichen der Zahl weglässt: für -5 ist der Absolutwert also 5; für 5 (also +5) ist der Absolutwert ebenfalls 5. Die Printausgabe soll im Falle negativer Werte auch den ursprünglichen Wert enthalten, z.B. “Absoluter Wert = 5, also war a -5”)

---Lösung---

```
if a >= 0:
    print(a)
else:
   print(a, -a)
    print("a absolut betraegt {}, also war a {}" .format(-a, a))
```

2. Aufgabe

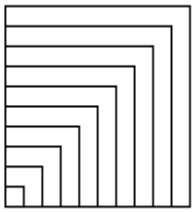
Schreibe ein Programm, das alle geraden Zahlen in der Liste “gerade_zahlen” und alle ungeraden in der Liste “ungerade_zahlen” ausgibt. (0 bis 25).

---Lösung---

```
gerade=[]
ungerade=[]
for i in range(25):
    if i%2==0:
        gerade.append(i)
    else:
        ungerade.append(i)
print(gerade, ungerade)
```

3. Aufgabe

Schreibe ein Programm, das zehn ineinander geschachtelte Quadrate zeichnet, bei denen sich die Seitenlänge um jeweils 10 erhöht, s. Bild:



---Lösung---

```
def sqr(le):
    for i in range(4):
        fd(le)
        rt(90)
#sqr(10)
def sq():
    for i in range(10):
        sqr(i*10)
sq()
```

4. Aufgabe

Schreibe ein Programm, das beliebig viele (zwischen 1 und 10) einstellige Zahlen in eine Liste schreibt und dann aus ihnen den Durschnitt bildet. Die Liste musst du mit randint füllen (list comprehension, du brauchst randint für die einstelligen Zahlen und für die Anzahl)

----Lösung----

```
liste=[randint(1,9) for i in range(randint(1,10))]
for i in numlist:
    summe=summe+i
print(summe/(len(numlist))
```

5. Aufgabe

Erläutere die Ausgabe folgenden Codes, indem du für jeden Schleifendurchlauf die Inhalte von my_list aufschreibst:

```
my_list = [1, 10, 100, 1000]
for i in range(4):
    my_list[i] = 2*my_list[i]
```

```
    print(my_list)
```

Zusatzfrage: was ist der Unterschied zwischen 2*my_list[i] und my_list[i*2]

6. Aufgabe

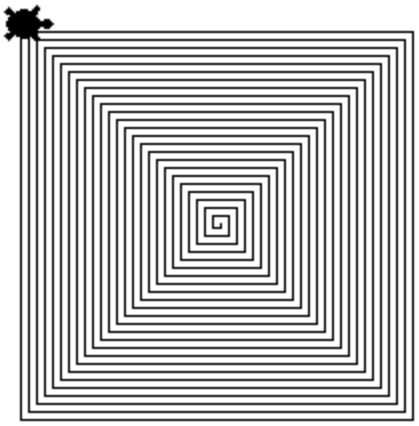
Schreibe eine Funktion, die eine beliebige Menge (z.B. 3 – 20) von vierstelligen Zufallszahlen in einer Liste als Argument entgegennimmt und dann den Median der Liste ermittelt (Liste wie immer mit randint füllen). Tipp: Beginne mit den Variablen l=len(mylist), s=sorted(mylist) und summe=sum(s[l//2-1:l//2+1]). Die Summen-Formel gibt dir die beiden mittleren Listenelemente bei gerader Listenanzahl. In Python 3 ist 5 // 2 = 2 (ganzzahliges Teilen)

```
liste=[randint(1000,9999) for i in range(randint(0,11))]

def middle(L):
    s=sorted(L)
    l=len(L)
    summe=sum(s[l//2-1:l//2+1])
    if l%2==0:
        med=summe/2.0
    else:
        med=s[l//2]
    print(liste, med)
middle(liste)
```

7. Aufgabe

Schreibe eine Funktion, die folgende Figur zeichnet:

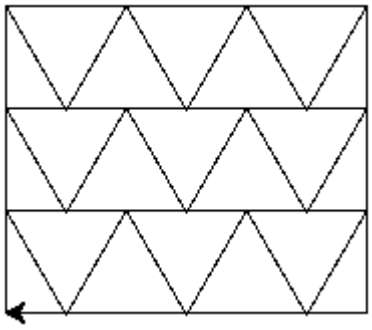


---Lösung---

```
from turtle import *
shape('turtle')
def spirinf(size):
    for i in range(100):
        delay(0.001)
        fd(i*2)
        rt(90)
spirinf(10)
```

8. Aufgabe (Zusatzaufgabe)

Schreibe, analog zu Weihnachtswald in 3.1.2, zwei Funktionen. Die erste zeichnet ein gleichseitiges Dreieck – nichts anderes ist der Wipfel in Weihnachtsbaum in 3.1.2 -, die zweite ordnet die Dreiecke je dreimal hintereinander entlang der x- und y-Achse des Koordinatensystems. Zusätzliche Schwierigkeit: Die Dreieckke sollen in der Höhe aufeinander aufliegen, s. Bild. Dazu benötigst du den Satz des Pythagoras.



---Lösung---

```
def tria(x,y,le):
    pu()
    goto(x,y)
    pd()
    setheading(0)
    for i in range(3):
        fd(le)
        rt(120)
```

```
#tria(50,100,60)


def tria3(le):
    b=le**2-(le/2)**2
    res=int(sqrt(b))
    for x in range(-le, le+1, le):
        for y in range(-res, res+1, res):
            tria(x,y,le)
            pu()
    goto(-le, -res+-res)
    pd()
    setheading(90)
    fd(res*3)
    rt(90)
    fd(le*3)
    rt(90)
    fd(res*3)
    rt(90)
    fd(le*3)

tria3(60)
```