1. (2 Punkte) Liste ist unser selbstgebastelter Datentyp. Wir nummerieren die Einträge der Liste mit Zeiger `anf` an der Stelle 0.

   a. Welche Zahlenfolge findet sich in den Listenlementen?
   b. Gib den Index des Elements an, auf das der Zeiger `pos` zeigt.

```
a = Liste()
a.insert(1)
a.insert(2)
a.advance()
a.insert(3)
a.insert(4)
a.reset()
a.advance()
a.insert(6)
```

Lösung: a. `2 6 4 3 1` b. `1`

2. (2 Punkte) Liste ist unser selbstgebastelter Datentyp. Wir nummerieren die Einträge der Liste mit Zeiger `anf` an der Stelle 0.

   a. Welche Zahlenfolge findet sich in den Listenlementen?
   b. Gib den Index des Elements an, auf das der Zeiger `pos` zeigt.

```
a = Liste()
a.insert(1)
a.insert(2)
a.insert(3)
a.insert(4)
```

Lösung: a. `4 3 2 1` b. `0`

3. (3 Punkte) Welche Ausgabe erscheint auf der Konsole?

```
class VerweisBox:
    def __init__(self, inhalt, unten=None, oben=None):
        '''
        inhalt: ein Zeichen
        unten, oben: eine Verweisbox
        '''
        self.inhalt = inhalt
        self.unten = unten
        self.oben = oben
        if unten is not None: self.unten.oben = self
        if oben  is not None: self.oben.unten = self

    def __str__(self):
        return self.inhalt

a = VerweisBox('a')
b = VerweisBox('b')
c = VerweisBox('c',a,b)
d = VerweisBox('d',b,c)
print(a.unten)
print(b.unten.oben)
print(b.oben.oben.oben)
print(c.unten)
print(c.oben.oben.unten.unten.unten)
```

Lösung: `None b b d d`

4. (3 Punkte) Welche Ausgabe erscheint auf der Konsole?

```
class VerweisBox:
    def __init__(self, inhalt, unten=None, oben=None):
        self.inhalt = inhalt
        self.unten = unten
        self.oben = oben
        if unten is not None: self.unten.oben = self
        if oben  is not None: self.oben.unten = self

    def __str__(self):
        return self.inhalt

a = VerweisBox('a')
b = VerweisBox('b')
c = VerweisBox('c',a,b)
e = VerweisBox('e',c,c)
print(a.oben)
print(b.oben)
print(b.unten)
print(c.oben.unten)
print(e.oben.unten.unten.oben)
```

Lösung: `c None c c e`

5. (4 Punkte) Die Klasse `FeatureListe` erbt von Liste. Sie hat zusätzlich ein Attribut `gerade`, das genau dann `True` ist, wenn die Anzahl der gespeicherten Listenelemente gerade ist. Schreibe die Klasse `FeatureListe`.

Lösung:

```
class FeatureListe(Liste):
    def __init__(self):
        super().__init__()
        self.gerade = True
    def insert(self,x):
        super().insert(x)
        self.gerade = not self.gerade
    def delete(self):
        super().delete()
        self.gerade = not self.gerade
```

6. (2 Punkte) a. Welche Datenstruktur wird hier verwendet (Liste/Keller/Schlange)?
   b. Was erscheint auf der Konsole?

```
a = []

for k in range(3):
    for i in range(3):
        a.append(i)
    print(a.pop(), end = ' ')

while a:
    print(a.pop(), end = ' ')
```

Lösung: a. Keller b. 2 2 2 1 0 1 0 1 0

7. (2 Punkte) a. Welche Datenstruktur wird hier verwendet (Liste/Keller/Schlange)?
   b. Was erscheint auf der Konsole?

```
from collections import deque
a = deque([])

for i in range(3):
    for i in range(3):
        a.append(i)
    print(a.popleft(), end = ' ')

while a:
    print(a.popleft(), end = ' ')
```

Lösung: a. Schlange b. 0 1 2 0 1 2 0 1 2