

1. (2 Punkte) Liste ist unser selbstgebastelter Datentyp. Wir nummerieren die Einträge der Liste mit Zeiger **anf** an der Stelle 0.
- Welche Zahlenfolge findet sich in den Listenlementen?
  - 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)
```

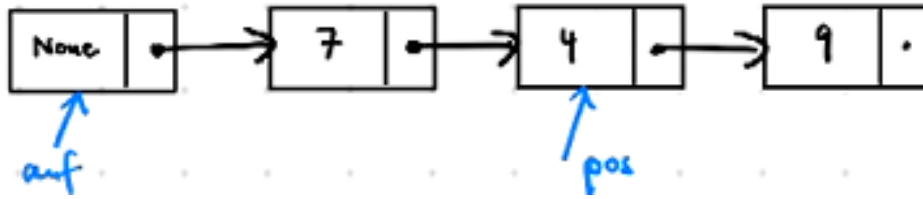
**Lösung:** a. 2 6 4 3 1      b. 1

2. (2 Punkte) Liste ist unser selbstgebastelter Datentyp. Wir nummerieren die Einträge der Liste mit Zeiger **anf** an der Stelle 0.
- Welche Zahlenfolge findet sich in den Listenlementen?
  - 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. (5 Punkte) Liste ist unser selbstgebastelter Datentyp. Das Bild zeigt die Situation der Liste a.



a. Was liefern die folgenden Anweisungen:

```
print(a.elem())
print(a.endpos())
```

b. Zeichne die Situation nach der Ausführung von `a.insert(12)`

c. Was liefern jetzt die folgenden Anweisungen:

```
print(a.elem())
print(a.endpos())
```

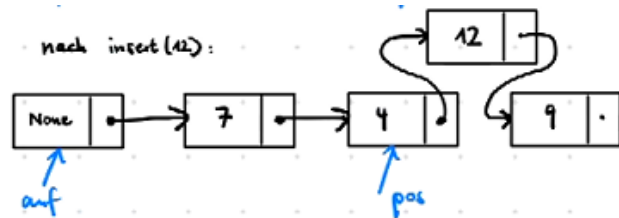
d. Wir gehen nochmal von der Anfangssituation aus. Zeichne die Situation nach Ausführung von `a.delete()`

e. Was liefern jetzt die folgenden Anweisungen:

```
print(a.elem())
print(a.endpos())
```

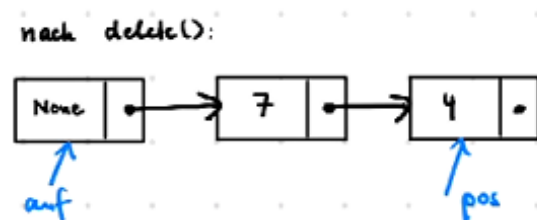
**Lösung:** a. 9, False

b.



c. 12, False

d.



e. Fehler, True

4. (3 Punkte) a. Gib für die beteiligten Verweisboxen die Attributwerte nach Ablauf der untenstehenden Anweisungen an. (Angabe in der Form: inhalt = 'e', oben = a, unten = b)  
b. 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)
d = VerweisBox('d', b, c)
print(a.unten, b.unten.oben, b.oben.oben.oben, c.unten, c.oben.oben.unten.unten.unten)
```

**Lösung:**

Die Werte der Attribute:  
inhalt = 'a', unten = None, oben = c  
inhalt = 'b', unten = c, oben = d  
inhalt = 'c', unten = d, oben = b  
inhalt = 'd', unten = b, oben = c

Die **print**-Ausgabe:  
None b b d d

5. (3 Punkte) a. Gib für die beteiligten Verweisboxen die Attributwerte nach Ablauf der untenstehenden Anweisungen an. (Angabe in der Form: inhalt = 'e', oben = a, unten = b)  
b. 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

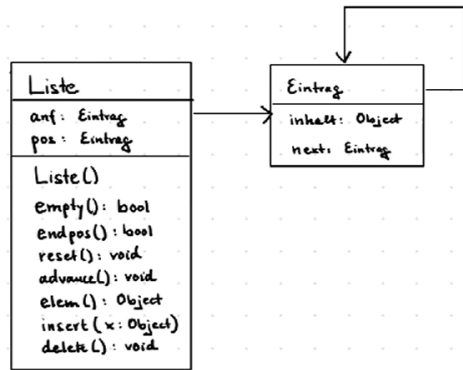
a = VerweisBox('a')
b = VerweisBox('b')
c = VerweisBox('c', a, b)
e = VerweisBox('e', c, c)
print(c.oben.unten, e.oben.unten.unten.oben)
```

**Lösung:**

Die Werte der Attribute:  
inhalt = a, unten = None, oben = c  
inhalt = b, unten = c, oben = None  
inhalt = c, unten = e, oben = e  
inhalt = e, unten = c, oben = c

Die **print**-Ausgabe:  
c e

6. (4 Punkte) Das Bild zeigt das UML-Diagramm unseres Datentyps Liste. Implementiere in Python die Klassen Eintrag und die Klasse Liste (nur Konstruktor und Methode delete). Achte dabei auf mögliche Fehlersituationen.



### Lösung:

```

class Eintrag:
    def __init__(self):
        self.inhalt = None
        self.next = None

class Liste:
    def __init__(self):
        self.anf = Eintrag()
        self.pos = self.anf

    def delete(self):
        if self.endpos(): raise RuntimeError("Fehler: Liste am Ende")
        self.pos.next = self.pos.next.next
  
```

7. (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**

8. (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**