Datenstrukturen: Bag (Multimenge) **Prüfungsstoff**

1. Du kannst die Merkmale dieser Datenstruktur (*Reihenfolge unwesentlich*, *Wiederholungen eines Elements erlaubt*) aufzählen.

2. Du kannst mindestens zwei Anwendungen für diese Datenstruktur angeben. (Tabellen in SQL-Datenbanken, Faktorisierung von Primzahlen, Wörter in einem Text)

3. Du kannst die für die Datenstruktur typischen Operationen beschreiben: *einen leeren Bag erzeugen*, *Elemente hinzufügen*, *den Bag löschen*, *über den Bag iterieren*, *die Anzahl der Elemente zurückgeben*. Das Löschen von Elementen ist wünschenswert und *kann* aber muss nicht in der Antwort auf diese Frage angegeben werden. (Beachte, dass es mehrere Arten gibt, „ein“ Element zu löschen, wenn die Datenstruktur Duplikate erlaubt.)

4. Du kannst die referenzbasierte (Synonym: *zeigerbasierte*) Implementierung dieser Datenstruktur im Speicherabbild nachvollziehen. Die Struktur eines Knotens im Speicher wird gegeben.

5. Du kannst Einfüge- und Löschoperationen am Speicherabbild durchführen.

6. Du kannst einfache Codebeispiele mit der `yield`-Anweisung nachvollziehen.

7. Du kannst die folgenden Methoden in Python implementieren und ihre jeweilige asymptotische Laufzeitkomplexität angeben, wenn der Bag n Elemente enthält.

   - `b.add(item)`
   - `b.clear()`
   - `b.size()`

   Der Code für die Konstruktoren der Klassen `Node` und `Bag` (siehe Dossier) wird gegeben. Um die Darstellung zu vereinfachen, können einige der Variablen andere Namen haben (z. B. `self.n` statt `self._size`).

   Die *Codes* zum Finden und zum Entfernen eines Elements (siehe Aufgaben 4 und 5 im Dossier) werden *nicht* verlangt.