Datenstrukturen (Teil 1) **Prüfungsstoff**

1. Du kannst den Begriff des *Datentyps* beschreiben.
2. Du kannst drei elementare Datentypen von Python aufzählen.
3. Du kannst den Begriff der *Schnittstelle* beschreiben.
4. Du kannst die Schnittstelle des in Python eingebauten Datentyps `list` für folgende Aufgaben verwenden (`L` und `M` sind jeweils Listen, `e` steht für ein Element):

   - Zugriff auf das i-te Element einer Liste: `L[i]` $[O(1)]$
   - Überschreiben des i-ten Elements einer Liste mit `e`: `L[i]=e` $[O(1)]$
   - Hinzufügen von `e` am Ende der Liste: `L.append(e)` $[O(1)]$
   - Entfernen des Elements am Ende der Liste: `L.pop()` $[O(1)]$
   - Entfernen eines Elements an einer Position $0 \leq i < n-1$: `L.pop(i)` $[O(n)]$
   - Einfügen von `e` an einer Position $0 \leq i < n-1$: `L.insert(i,e)` $[O(n)]$
   - Anzahl Elemente von `L`: `len(L)` $[O(1)]$
   - Iterieren über `L`: `for e in L:` oder `for i in range(0,len(L)):` $[O(n)]$
   - Verketten von zwei Listen: `L+M` $[O(n_L + n_M)]$

   Darüber hinaus kennst du die Laufzeitkomplexitäten dieser Operationen (siehe [...]).
5. Du kannst die Schnittstelle des in Python eingebauten Datentyps `dict` für folgende Aufgaben verwenden. `D` ist jeweils ein Dictionary, `a` ein Schlüssel (ein unveränderliches Objekt wie eine Zahl oder ein String und `e` ist ein beliebiger Wert.

   - Zugriff auf das Element mit dem Schlüssel `a`: `D[a]` $[O(1)]$
   - Hinzufügen von `e` zum Dictionary: `D[a] = e` $[O(1)]$
   - Entfernen des Elements mit dem Schlüssel `a`: `D.del(a)` $[O(1)]$
   - Prüfen, ob der Schlüssel `a` im Dictionary vorkommt: `a in D` $[O(1)]$
   - Anzahl Elemente von `D`: `len(D)` $[O(1)]$
   - Iterieren über `D`: `for key in D:` $[O(n)]$

   Ferner kennst du die Laufzeitkomplexitäten dieser Operationen (siehe [...]).
6. Du verstehst die Semantik (Bedeutung) der folgenden Anweisungen aus der Schnittstelle des in Python eingebauten Datentyps `set` (Menge). `A` und `B` sind jeweils Sets und `e` ein Wert.

   - Vereingungsmenge $A \cup B$: `A.union(B)`
   - Schnittmenge $A \cap B$: `A.intersection(B)`
   - Differenzmenge $A \setminus B$: `A.difference(B)`
   - Liegt `e` $\in$ `A`?: `e in A`
   - `e` zu `A` hinzufügen: `A.add(e)`

- `e` aus `A` entfernen: `A.remove(e)`
- Anzahl Elemente von `A`: `len(A)`
- Iterieren über `A`: `for item in A:`

Die Effizienz dieser Python-Operationen wurden nicht besprochen und gehören deshalb nicht zum Prüfungsstoff.

7. Abstrakter Datentyp *Stack*:

   - Du kannst die für den Datentyp zentralen Methoden `push(element)`, `pop()`, `peek()`, `size()` und `isEmpty()` in Worten beschreiben und den dazugehörigen Python-Code im Falle einer Implementierung (mittels Liste) angeben.
   - Du kannst die Kurzfomrel des Datentyps *Last In – First Out (LIFO)* angeben.
   - Du kannst mindestens drei verschiedene Anwendungen für Stacks aufzählen.
   - Du kannst die Postifix-Darstellung eines arithmetischen Terms in die entsprechende Infix-Darstellung umwandeln und umgekehrt.
   - Du kannst den Verlauf von Operationen auf einem Stack nachvollziehen.

8. Abstrakter Datentyp *Queue*:

   - Du kannst die für den Datentyp zentralen Methoden `enqueue(element)`, `dequeue()`, `size()` und `isEmpty()` in Worten beschreiben und den dazugehörigen Python-Code im Falle einer Implementierung (mittels Liste) angeben.
   - Du kannst die Kurzformel des Datentyps *First In – First Out (FIFO)* angeben.
   - Du kannst mindestens drei verschiedene Anwendungen für Queues aufzählen.
   - Du kannst den Verlauf von Operationen auf einer Queue nachvollziehen.