

Übung 01: Verkettete Liste

Abgabetermin: 13.03.2007 12:00

Name: _____

Matrikelnummer: _____

Gruppe: G1 (Dhungana) G2 (Wolfinger) G3 (Wolfinger)

Aufgabe	Punkte	gelöst	abzugeben schriftlich	abzugeben elektronisch	Korr.	Pkte
Aufgabe 01.1	12	<input type="checkbox"/>	Java-Programm Testfälle und Ergebnisse	Java-Programm	<input type="checkbox"/>	

Aufgabe 01.1: Zahlenmenge als verkettete Liste

Implementieren Sie eine Zahlenmenge mit Hilfe einer linearen Liste (Hinweise: Ein Menge enthält niemals mehrere Exemplare der gleichen Zahl. Die Reihenfolge der Zahlen ist ohne Bedeutung).

```
class Set {
    Node head; // first node in list
    public Set() { ... }
    ...
}
```

```
class Node {
    int val; // value in node
    Node next; // next node in list
    public Node(int val) {... }
}
```

Vervollständigen Sie die beiden Konstrukturen und implementieren Sie in der Klasse `Set` Klasse folgende Methoden:

`void set(int val)`: fügt eine neue Zahl in die Menge ein, wenn diese Zahl noch nicht enthalten ist

`bool get(int val)`: prüft ob eine Zahl in der Menge enthalten ist

`int size()`: liefert die Anzahl der Zahlen in der Menge

`void remove(int val)`: entfernt eine Zahl aus der Menge

`Set clone()`: liefert eine Kopie der Menge als neue Menge

`void print()`: gibt eine Menge auf der Kommandozeile aus

`Set intersect(Set s)`: liefert Schnittmenge aus `this` und `s` als neue Menge

`Set union(Set s)`: liefert Vereinigungsmenge aus `this` und `s` als neue Menge

`Set range(int from, int to)`: liefert alle Zahlen, die im Bereich zwischen `from` und `to` liegen, als neue Menge (Bereichsgrenzen jeweils eingeschlossen \geq , \leq)

Beispiel: Gegeben sind die Mengen `s1` und `s2`:

```
s1:      { -9 -5 -4 -3 0 2 4 10 }
s2:      { -5 -3 0 1 2 7 9 }
```

Folgende Operation auf diesen Mengen `s1` und `s2` sollen folgende Ergebnisse liefern:

```
s1.union(s2):  { 10 4 -4 -9 9 7 2 1 0 -3 -5 }
s1.intersect(s2): { 2 0 -3 -5 }
s1.range(0, 10): { 10 4 2 0 }
s2.range(-10, 0): { 0 -3 -5 }
s2.remove(0):   { -5 -3 1 2 7 9 }
s2.remove(2):   { -5 -3 1 7 9 }
s1.intersect(s2): { -3 -5 }
```

Abzugeben ist:

- Das Java-Programm
- Testfälle und die Ergebnisse