FMUD

Termin 3

FMuD

ArrayList, LinkedList, HashMap

LinkedList vs ArrayList

LinkedList	ArrayList
Verwendung:	
- Wenn häufig Element der Liste hinzugefügt werden oder entfernt werden => Zeitlich konstante Ausführung für Insert und Remove - Und eher seltener auf die Elemente in der Liste zugegriffen werden muss => Sequenzieller Zugriff auf Elemente	- Wenn häufig auf Elemente innerhalb der Liste zugegriffen wird =>Zeitlich kostanter Zugriff auf ein bestimmtes Element - Und eher seltener Element innerhalb der Liste, nicht am Ende der Liste, hinzugefügt oder entfernt werden => Erfordert das Verschieben aller Elemente

Aufgabe 9

Erstellen Sie eine “Mitarbeiter”-Klasse und eine entsprechende Klasse, die die main-Methode abbildet. Die “Mitarbeiter”-Klasse soll die beiden Attribute “Personalnummer” und “Name” enthalten. Erzeugen Sie zwei Instanzen der Klasse “Mitarbeiter” und füllen Sie die Attributen mit entsprechenden Werten.

ArrayList

Neue ArrayList:

```
ArrayList<Datentyp> varName =
    new ArrayList<Datentyp>( );
```

Objekte in ArrayList ablegen:

```
varName.add(objName);
```

Objekte aus ArrayList auslesen:

```
varName.get(int-Wert);
```

```
import java.util.*;
```

Aufgabe 10

Erstellen Sie eine ArrayList, in die Objekte des Datentyps Mitarbeiter abgelegt werden können.

Legen Sie die vorher erstellten Mitarbeiter-Objekte in dieser ArrayList ab.

Geben Sie die Werte der Attribute Personalnummer und Name aus, indem Sie auf die ArrayList-Einträge zugreifen.

LinkedList

Neue LinkedList:

```
LinkedList<Datentyp> varName =
  new LinkedList<Datentyp>( );
```

Objekte in LinkedList ablegen:

```
varName.add(objName);
```

Objekte aus LinkedList auslesen:

```
varName.get(int-Wert);
```

Aufgabe 11

Erstellen Sie eine LinkedList, in die Objekte des Datentyps Mitarbeiter abgelegt werden können.
Legen Sie die vorher erstellten Mitarbeiter-Objekte in dieser LinkedList ab.
Geben Sie die Informationen zu Personalnummer und Name aus, indem Sie auf die LinkedList-Einträge zugreifen.

HashMap

Neue HashMap:

```
HashMap<Datentyp, Datentyp>
varName = new HashMap<Datentyp,
 Datentyp>( );
```

Typ des Schlüssels

Typ der abzulegenden Objekte

Objekte in HashMap ablegen:

```
varName.put(schlName, objName);
```

Objekte aus HashMap auslesen:

```
varName.get(schlName);
```

```
import java.util.*;
```

Aufgabe 12

Erstellen Sie eine HashMap, in der Objekte des Datentyps Mitarbeiter abgelegt werden können. Als Schlüssel soll die PersonalNr genutzt werden. Legen Sie die vorher erstellten Mitarbeiter-Objekte in dieser HashMap ab. Geben Sie die Informationen zu Personalnummer und Name aus, indem Sie auf die HashMap-Einträge zugreifen.

HashMap Schlüssel und/oder Werte ausgeben

Um mittels einer Schleife alle Werte einer HashMap ausgeben zu lassen, eignen sich zwei Varianten

- Ein Set der Schlüssel generieren (keySet()), welche in der HashMap gespeichert sind und mit Hilfe des Schlüssels auf den Wert zugreifen

- Eine Collection mit den Werten generieren (values()), welche in der HashMap gespeichert sind

Aufgabe 13

Geben Sie alle Werte Ihrer HashMap aus Aufgabe 12 aus. Verwenden Sie hierfür einmal die Variante mit keySet() und einmal mit der Variante values().