Name:

Matrikelnummer:

Bearbeitungszeit: 120 min.

Aufgabe 1. Schreiben Sie eine Klasse `Frachtschiff` mit dem Konstruktor

```
public Frachtschiff(int maxDistanz),
```

dem die maximale Distanz (in Seemeilen) übergeben wird, die das Frachtschiff bei vollem Tank zurücklegen kann. Weiters soll die Klasse über die Methode

```
public int getAnzBetankungen(int[] distanzen)
```

verfügen, die die minimale Anzahl an notwendigen Betankungen zurückgibt, wenn das Schiff eine Route fährt, für die die Distanzen zwischen den anzufahrenden Häfen in `distanzen` angegeben sind. Zu Beginn ist das Frachtschiff bereits voll betankt, und keiner der Werte in `distanzen` ist größer als `maxDistanz`.

Beispiel: Für `maxDistanz=100` und `distanzen={20,30,40,50,50,10}` soll die Methode den Wert 2 liefern.

Aufgabe 2. Schreiben Sie eine Klasse `Wegfindung` mit einem parameterlosen Konstruktor und der Methode

```
public boolean existiertWeg(boolean[][] istWeg,
                            int iS, int jS, int iZ, int jZ).
```

Das doppelt indizierte Array `istWeg` stellt ein rechteckiges Labyrinth dar, wobei ein Feld `[i][j]` des Labyrinths begehbar ist, wenn `istWeg[i][j]==true` ist. Die Methode `existiertWeg(istWeg,iS,jS,iZ,jZ)` soll `true` zurückgeben, wenn es einen Weg vom Startfeld `[iS][jS]` zum Zielfeld `[iZ][jZ]` gibt, und `false` andernfalls. Von einem begehbaren Feld `[i][j]` sind jeweils die Nachbarfelder `[i-1][j]`, `[i+1][j]`,`[i][j-1]`,`[i][j+1]` direkt erreichbar, sofern sie begehbar sind und innerhalb des Labyrinths liegen.

Beispiel: Für das Labyrinth

	0	1	2	3	4	5	6
0		F	F				F
1	z_2	F	F		F	z_1	
2	F		s			F	
3		F	F	F			

mit Startfeld `[2][2]` soll die Methode für Zielfeld `[1][5]` `true` zurückgeben, und für das Zielfeld `[1][0]` `false`.

Aufgabe 3. Gegeben sind die Klassen `Maschine` und `Steuerung` mit folgenden Konstruktoren und Methoden:

```
public class Maschine {
        public Maschine(String bezeichnung);
        public boolean fertige(String auftrag);
}

public class Steuerung {
        public Steuerung();
        public void steuere(Maschine masch, String[] auftragsliste);
        public String[] getFehler();
}
```

Die Methode `fertige(auftrag)` bearbeitet einen Auftrag auf der Maschine. Falls die Fertigung fehlschlägt, liefert die Methode `false` zurück, andernfalls `true`. Die Methode `steuere(masch,auftragsliste)` ruft `masch.fertige(auftrag)` für eine geeignete Reihenfolge der in `auftragsliste` angegebenen Aufträge auf. (Die Auftragsbezeichnungen sind eindeutig.) Fehlgeschlagene Fertigungen werden protokolliert und werden von `getFehler()` zurückgeliefert.

Schreiben Sie Unterklassen `MaschineMitMeldung` und `SteuerungMitMeldung` von `Maschine` bzw. `Steuerung`. Die Klasse `SteuerungMitMeldung` soll über die zusätzlichen Methoden

```
public void meldung(String fehlermeldung),
public String[] getFehlermeldungen(),
```

verfügen. Der Aufruf von `meldung(fehlermeldung)` ermöglicht es einer Maschine, zu einem fehlgeschlagenen Auftrag eine Fehlermeldung an die Steuerung zu senden. Die Methode `getFehlermeldungen()` liefert alle Fehlermeldungen zurück.

Die Klasse `MaschineMitMeldung` soll über die Methode

```
public void setSteuerung(SteuerungMitMeldung steuer)
```

verfügen, mit der der Maschine die für sie zuständige Steuerung bekanntgegeben wird. Weiters soll in `MaschineMitMeldung` die Methode

```
public boolean fertige(String auftrag)
```

so überschrieben werden, dass die Anzahl der fehlerhaften Fertigungen mitgezählt wird, und bei einer fehlerhaften Fertigung die Fehlermeldung `"x. Fertigungsfehler"` an `steuer` gesandt wird (`x` ist die Anzahl der bisher aufgetretenen Fehler). Die Fehlermeldung soll nur dann an `steuer` gesandt werden, wenn die Steuerung zuvor gesetzt wurde, ansonsten soll keine Fehlermeldung versandt werden.