Name:

Matrikelnummer:

Aufgabe 1.
Zur Regelung der Einfahrt in ein Parkhaus sind die Klassen *Parkkarte* und *Schrankenanlage* gegeben. Die Klasse *Schrankenanlage* verfügt über einen Konstruktor

```
1  public Schrankenanlage(String typ),
```

der eine Schrankenanlage vom angegebenen Typ erstellt.

Die Klasse *Schrankenanlage* verfügt außerdem über folgende Methoden:

```
public Parkkarte fuerEinfahrtOeffnen()
```

öffnet den Schranken für die Einfahrt und gibt eine Parkkarte zurück.

```
public boolean fuerAusfahrtOeffnen(Parkkarte eineKarte)
```

öffnet den Schranken für die Ausfahrt, falls auf *eineKarte* eine korrekte Zahlung für den Aufenthalt gebucht wurde. In diesem Fall gibt die Methode *true* zurück, andernfalls *false*.

Schreiben Sie eine Klasse *SchrankenanlageMitKapa*, die von *Schrankenanlage* erbt. Der Konstruktor

```
public SchrankenanlageMitKapa(String typ, int kapa)
```

soll eine Schrankenanlage vom angegebenen Typ für ein Parkhaus mit maximaler Auslastung von *kapa* Fahrzeugen erstellen.

Überschreiben Sie geeignete Methoden, sodass nach Aufruf von *fuerEinfahrtOeffnen* der Schranken nur dann geöffnet und eine Parkkarte zurückgegeben wird, wenn die maximale Auslastung noch nicht erreicht wurde. Andernfalls soll *null* zurückgegeben werden.

Schreiben Sie außerdem eine Methode

```
public int gibFreiePlaetze(),
```

die die Anzahl der freien Plätze im Parkhaus zurückgibt.

Aufgabe 2.
Gegeben seien die Klassen *Produkt* und *Anbieter*. Die Klasse *Anbieter* verfügt über die folgenden Methoden:

```
public boolean kannLiefern(Produkt einProdukt)
```

gibt genau dann *true* zurück, wenn der Anbieter *einProdukt* liefern kann.

```
public double gibPreis(Produkt einProdukt)
```

gibt den Preis des Anbieters für *einProdukt* zurück.

Schreiben Sie eine Methode

```
public static Anbieter gibBestenAnbieter(List<Produkt> dieProdukte,
                                         List<Anbieter> dieAnbieter),
```

die einen Anbieter aus *dieAnbieter* zurückgibt, der unter jenen Anbietern, die alle Produkte aus *dieProdukte* liefern können, diese für den kleinsten Gesamtpreis anbietet. Sollte es keinen Anbieter geben, der alle Produkte liefern kann, soll die Methode *null* zurückgeben.

Aufgabe 3.
Gegeben seien die Klasse *Uhrzeit*, die das Interface *Comparable* implementiert (dementsprechend sind natürlich auch *equals* und *hashCode* überschrieben), sowie die Klasse *Zugverbindung* mit der Methode

```
public ArrayList<String> gibStrecke(),
```

die die Haltebahnhöfe der Zugverbindung in einer ArrayList zurückgibt.

Schreiben Sie eine Klasse *BahnhofsInfo* mit Konstruktor

```
public BahnhofsInfo(String ort, TreeMap<Uhrzeit,Zugverbindung> dieVerbindungen),
```

wobei *ort* der Ort ist, in dem der Bahnhof steht, und in *dieVerbindungen* alle Zugverbindungen des Bahnhofs an einem Tag unter der entsprechenden Abfahrtszeit abgelegt sind. (Sie können davon ausgehen, dass die Strecken dieser Verbindungen nur die Haltebahnhöfe ab *ort* enthalten.)

Schreiben Sie in die Klasse *BahnhofsInfo* außerdem eine Methode

```
public Zugverbindung gibNaechsteVerbindung(Uhrzeit aktZeit, String zielOrt),
```

die die nächste Zugverbindung nach *zielOrt* zurückgibt, wenn *aktZeit* die aktuelle Uhrzeit ist. Falls es am selben Tag keine entsprechende Verbindung gibt, so soll die Methode *null* zurückgeben.