5. Zugelassene Hilfsmittel

- GTR (grafikfähiger Taschenrechner) oder CAS (Computer-Algebra-System)
- Wörterbuch zur deutschen Rechtschreibung

6. Modelllösungen

Die jeweilige Modelllösung stellt eine mögliche Lösung bzw. Lösungsskizze dar. Der gewählte Lösungsansatz und -weg der Schülerinnen und Schüler muss nicht identisch mit dem der Modelllösung sein. Sachlich richtige Alternativen werden mit entsprechender Punktzahl bewertet (Bewertungsbogen: Zeile „Sachlich richtige Lösungsalternative zur Modelllösung“).

Teilaufgabe a)

Ein Objekt der Klasse `Veranstaltungsverwaltung` verwaltet im Attribut `veranstaltungen` seine Veranstaltungen in einer linearen Liste mit Objekten der Klasse `Veranstaltung`. Jedes Objekt der Klasse `Veranstaltung` verwaltet seine Sitzplätze in einem zweidimensionalen Array mit dem Bezeichner `sitzplaetze`, welches Objekte vom Typ `Person` verwaltet. Zusätzlich wird unter dem Bezeichner `warteliste` eine Schlange (`Queue`) mit Objekten vom Typ `Person` gespeichert.

Da vorab nicht bekannt ist, wie viele Personenobjekte in der Warteliste für eine Veranstaltung verwaltet werden müssen, ist es sinnvoll, eine dynamische Datenstruktur zu verwenden und nicht ein statistisches Feld (Array). In diesem Anwendungsfall ist eine Schlange sinnvoll, da diese Datenstruktur die Verwaltung der Reihenfolge der Objekte durch das FIFO-Prinzip sicherstellt, ein faires Nachrücken nach Sitzplatzfreigabe garantiert und ein wahlfreier Zugriff auf jedes Objekt in der Warteliste nicht notwendig ist.

Für eine Veranstaltung wird zunächst eine Bestuhlungsvorgabe (Sitzplatzreihen mit Sitzen) festgelegt, die sich grundsätzlich nicht verändern soll. Wenn alle Plätze reserviert sind, dann können sich Personen auf eine Warteliste setzen lassen. Der Zugriff auf ein Personenobjekt im zweidimensionalen Array über die Indizes lässt sich im Gegensatz zu einer dynamischen Datenstruktur effizienter umsetzen.

Teilaufgabe b)

Nur falls der Wert des Parameters `pAnzahl` größer `0` ist, geschieht etwas.

In Zeile 6 wird ein neues zweidimensionales Array für Personenobjekte deklariert und initialisiert, welches um `pAnzahl` mehr Sitzplatzreihen verfügt.

Die ersten zwei geschachtelten Zählschleifen (vgl. Zeilen 7 – 11) stellen sicher, dass alle vorhandenen Reservierungen in das neue Array übertragen werden.

Die zweiten zwei geschachtelten Zählschleifen (vgl. Zeilen 12 – 19) stellen sicher, dass – nur für die neuen Sitzplatzreihen – die Sitzplätze zunächst reihenweise von vorne nach hinten und in jeder Reihe von links nach rechts durchlaufen werden. Wenn ein Sitzplatz frei ist

(vgl. Zeile 14), dann wird das erste Personenobjekt von der Warteliste als neue Reservierung (vgl. Zeile 15) eingetragen. Falls Die Warteliste leer ist, wird an der Stelle `null` eingetragen (vgl. Zeile 15). Das erste Personenobjekt wird anschließend aus der Warteliste gelöscht (vgl. Zeile 16).

In Zeile 20 wird die Referenz `sitzplaetze` auf das neue Array-Objekt gesetzt.

Die Bestuhlung der Veranstaltung wird in diesem Beispiel um zwei Sitzplatzreihen mit je drei Sitzplätzen nach hinten erweitert. Die vier Personen von der Warteliste füllen dann die neuen Sitzplätze auf. Die Warteliste ist anschließend leer. Zwei Sitzplätze sind noch frei.

Der Parameter `pAnzahl` bestimmt, um wie viele Reihen die Bestuhlung erhöht wird, falls `pAnzahl` größer `0` ist. Die Personen von der Warteliste rücken automatisch auf die freien Plätze nach.

Die Verzweigung in Zeile 2 verhindert, dass durch einen negativen Wert des Parameters `pAnzahl` Sitzplatzreihen entfernt werden. So wird sichergestellt, dass die Anzahl der Reihen nur erhöht werden kann.

Die Verzweigung in Zeile 14 ist nicht notwendig, da die Schleife in Zeile 12 nur über die neu hinzugefügten Sitzplatzreihen läuft. Die neu hinzugefügten Sitzplatzreihen beinhalten zunächst leere Sitzplätze, die dann mit Personen von der Warteliste aufgefüllt werden.

Teilaufgabe c)

Erzeuge eine lokale, anfangs leere Liste `sortierte`, die Objekte vom Typ `Veranstaltung` verwaltet.

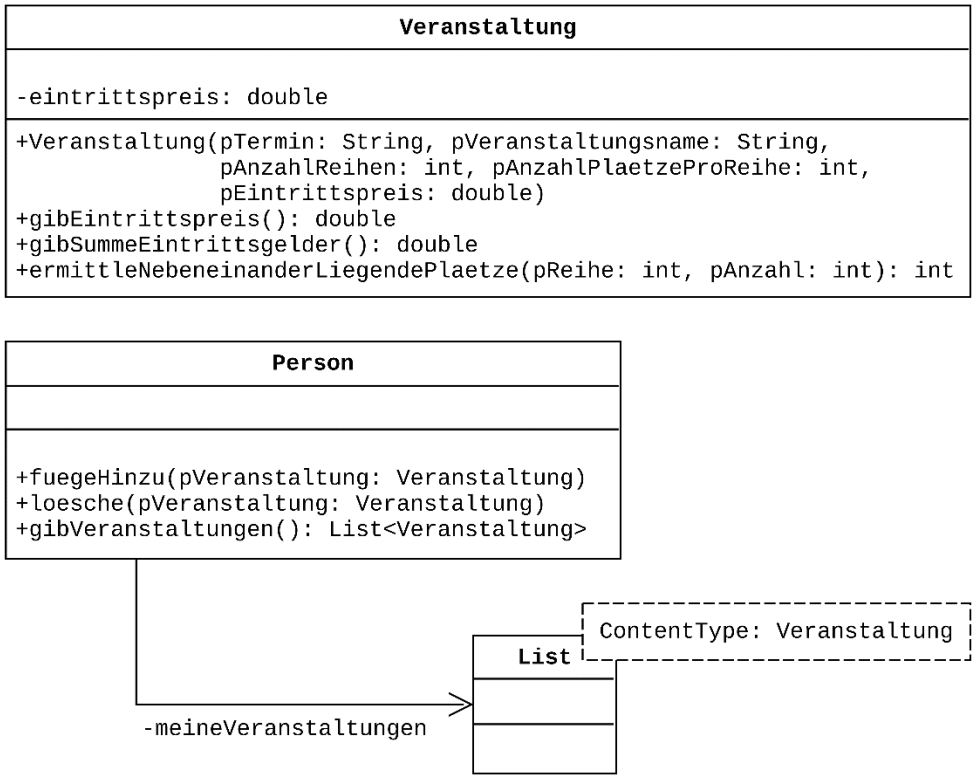
Durchlaufe die Liste `veranstaltungen` mit allen Objekten vom Typ `Veranstaltung` und füge schrittweise jedes Objekt sortiert in die Liste `sortierte` ein. Suche dafür in der Liste `sortierte` – beginnend am Anfang – die richtige Einfügestelle, d. h. man geht in der Liste der sortierten Veranstaltungsobjekte so lange weiter, wie die Anzahl der freien Plätze von der neu einzufügenden Veranstaltung maximal so groß ist wie die Anzahl der freien Plätze des aktuellen Veranstaltungsobjekts der sortierten Liste.

Diese lokale Liste mit den sortierten Veranstaltungen wird zurückgeliefert.

Eine mögliche Implementation der Methode:

```
public List<Veranstaltung> ermittleSortierte() {
  List<Veranstaltung> sortierte = new List<Veranstaltung>();
  veranstaltungen.toFirst();
  while (veranstaltungen.hasAccess()) {
    Veranstaltung veranstaltung = veranstaltungen.getContent();
    sortierte.toFirst();
    while (sortierte.hasAccess()
           && veranstaltung.gibAnzahlFreiePlaetze() <=
              sortierte.getContent().gibAnzahlFreiePlaetze()) {
      sortierte.next();
    }
    if (sortierte.hasAccess()) {
      sortierte.insert(veranstaltung);
    } else {
      sortierte.append(veranstaltung);
    }
    veranstaltungen.next();
  }
  return sortierte;
}
```

Teilaufgabe d)



Realisierung der zweiten Anforderung:
Die Methode `ermittleNebeneinanderLiegendePlaetze` in der Klasse `Veranstaltung` bekommt die Nummer einer Sitzplatzreihe und die gewünschte Anzahl der nebeneinanderliegenden Plätze übergeben.
Die Methode liefert einen Integerwert zurück, der die Nummer des ersten freien Sitzes der nebeneinanderliegenden Plätze in der gewünschten Reihe angibt. Falls keine nebeneinanderliegenden Plätze in der gewünschten Anzahl in der Reihe vorhanden sind, wird `-1` zurückgegeben.

Realisierung der dritten Anforderung:
Ein Objekt der Klasse `Person` verwaltet eine Liste mit Veranstaltungsobjekten. Über die Methode `fuegeHinzu` kann eine Veranstaltung der genannten Liste hinzugefügt und über die Methode `loesche` aus ihr gelöscht werden. Die Methode `gibVeranstaltungen` der Klasse `Person` liefert alle Veranstaltungen für dieses Personenobjekt als Liste zurück.

Teilaufgabe e)

Der Vorschlag der Schulleitung verstößt gegen das Grundprinzip des Verbots mit Erlaubnisvorbehalt.

Die Verwaltung der zusätzlichen personenbezogenen Daten (E-Mail-Adressen der Personen) wird mit dem Zweck legitimiert, eine schnelle Erreichbarkeit bei Absagen zu gewährleisten. Wenn die personenbezogenen Daten dann für Werbezwecke für weitere Veranstaltungen verwendet würden, entspräche dies nicht mehr dem konkreten Zweck, dem zugestimmt wurde.

Die Verarbeitung, d. h. zum Beispiel die Erhebung, Speicherung, Weitergabe oder allgemeine Verwendung personenbezogener Daten ist grundsätzlich verboten – es sei denn, die betroffene Person hat der Verarbeitung für einen konkreten Zweck zugestimmt oder es gibt eine explizite gesetzliche Regelung, die eine Verarbeitung für einen konkreten Zweck erlaubt.