Name: _____ Mat.Nr.: _____

Aufgabe 1

In einem Kino sollen freie nebeneinander liegende Sitze gefunden werden. Dazu gibt es die Klasse *Reihe* mit der Methode

boolean[] getFreieSitze(),

die ein Bool'sches Array zurückliefert, in dem die freien Sitze mit *true* gekennzeichnet sind. Jeder Index in dem Array entspricht einer Sitznummer, d.h. die Sitznummern beginnen mit 0.

In der Klasse *Kino* gibt es eine Methode

Reihe[] getReihen(),

die die Reihen des Kinos zurückliefert.

Ergänzen Sie die Klasse *Kino* um die Methode

ArrayList<Reihe> getFreieReihen(int anzahlFreieSitze),

die alle Reihen des Kinos zurückliefert, in denen es zumindest *anzahlFreieSitze* nebeneinander liegende freie Sitze gibt.

Aufgabe 2

Implementieren Sie in einer Klasse *Packen* die Methode

ArrayList<Integer> packeVoll(int[] gewichte, int maxGewicht),

die eine Auswahl der Zahlen im Array *gewichte* zurückliefert, deren Summe genau *maxGewicht* ergibt. In der Auswahl darf jedes Element von *gewichte* höchstens einmal vorkommen. Wenn es eine solche Auswahl nicht gibt, soll die Methode *null* zurückliefern. Sie können annehmen, dass alle Zahlen in *gewichte* positiv sind.

Aufgabe 3

Implementieren Sie eine Klasse *Hoersaal*, die die Bestuhlung eines Hörsaals darstellt. Die Klasse *Hoersaal* soll über folgende Konstruktoren und Methoden verfügen:

Hoersaal()

erzeugt einen leeren Hörsaal.

boolean addReihe(int reihennummer, int sitzeVon, int sitzeBis)

fügt dem Hörsaal eine Stuhlreihe mit der angegebenen *reihennummer* hinzu. Falls es eine Stuhlreihe mit dieser Nummer schon gibt, hat die Methode keine Wirkung und liefert *false* zurück, ansonsten *true*. Die Sitze in der neuen Reihe sind durchgehend von *sitzeVon* bis *sitzeBis* nummeriert.

boolean existsSitz(int reihennummer, int sitznummer)

liefert *true* zurück, wenn es in der angegebenen Reihe einen Sitz mit der angegebenen Nummer gibt, ansonsten *false*.