Programiranje I — 4. domača naloga

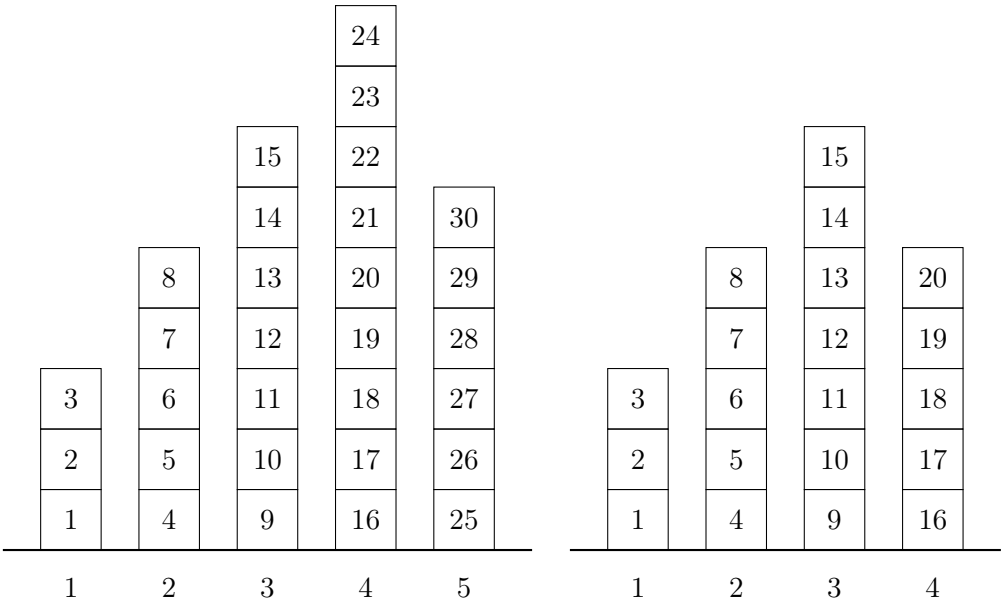
Rok za oddajo: nedelja, 3. december 2017, ob 23:55

Skladovnica

Naloga

Škatle zlagamo v kupe, ti pa tvorijo skladovnico. Prvi kup lahko vsebuje največ K_1 škatel, vsak naslednji pa ΔK škatel več kot prejšnji kup. Pričnemo s prazno skladovnico, nato pa škatle nanjo dodajamo in jih z nje odvzemamo. Pri dodajanju najprej napolnimo prvi kup, nato drugi itd., pri odvzemanju pa počnemo ravno obratno: najprej izpraznimo zadnji kup, nato predzadnji itd.

Za primer vzemimo skladovnico s parametroma $K_1 = 3$ in $\Delta K = 2$. Ko na prazno skladovnico dodamo 30 škatel, dobimo situacijo, prikazano na levi polovici slike 1. Po odstranitvi 10 škatel je skladovnica videti tako, kot prikazuje desni del te slike.



Slika 1: *Skladovnica s parametroma $K_1 = 3$ in $\Delta K = 2$ po dodajanju 30 in odstranitvi 10 škatel.*

Napišite razred `Skladovnica`, čigar objekt predstavlja skladovnico, sestavljeno iz kupov škatel. Razred naj implementira sledeče javno dostopne konstruktorje in metode, po želji in potrebi pa dodajte še lastne privatne atribute, konstruktorje in/ali metode:

- `public Skladovnica(int kapacitetaPrvega, int prirast)` [J1–J10, S1–S50]:[1]

  Izdela objekt, ki predstavlja prazno skladovnico s parametroma `kapacitetaPrvega` (K_1) in `prirast` (ΔK). V vseh testnih primerih velja `kapacitetaPrvega` $\geqslant 1$ in `prirast` $\geqslant 0$.

- `public int kapacitetaKupa(int kup)` [J1–J10, S1–S50]:

[1] V oglatih oklepajih so navedeni testni razredi, ki lahko kličejo obravnavani konstruktor oz. metodo.

Vrne kapaciteto (maksimalno število škatel) kupa z zaporedno številko `kup`. V vseh testnih primerih velja `kup` $\geqslant 1$.

- `public void dodaj(int stSkatel)` [J3–J10, S11–S50]:

  Na skladovnico doda `stSkatel` škatel. V testnih primerih J3–J4 in S11–S20 na skladovnici nikoli ne bo več kot en kup, v primerih J5 in S21–S25 pa bo posamezna operacija dodajanja ustvarila kvečjemu en nov kup. V vseh testnih primerih velja `stSkatel` $\geqslant 1$.

  Skupno število škatel na skladovnici v nobenem testnem primeru ne bo preseglo vrednosti 10^6.

- `public int skupnoSteviloSkatel()` [J3–J10, S11–S50]:

  Vrne skupno število škatel na skladovnici.

- `public int zasedenostKupa(int kup)` [J5–J10, S21–S50]:

  Vrne trenutno število škatel v kupu z zaporedno številko `kup`. Velja `kup` $\geqslant 1$.

- `public boolean odvzemi(int stSkatel)` [J7–J10, S31–S50]:

  Če je na skladovnici vsaj `stSkatel` škatel, s skladovnice odstrani `stSkatel` škatel in vrne `true`, sicer pa ne naredi ničesar in zgolj vrne `false`. Velja `stSkatel` $\geqslant 1$.

- `public int poisciKup(int skatla)` [J8–J10, S36–S50]:

  Vrne zaporedno številko kupa, ki vsebuje škatlo z zaporedno številko `skatla`, oziroma -1, če takega kupa ni. Prvi kup vsebuje škatle s številkami od 1 do K_1, drugi od $K_1 + 1$ do $2K_1 + \Delta K$ itd. (gl. sliko 1). Velja `skatla` $\geqslant 1$.

- `public Skladovnica prestavi(int kapacitetaPrvega, int prirast)` [J9–J10, S41–S50]:

  Prestavi škatle s skladovnice `this` na novo skladovnico s parametroma `kapacitetaPrvega` in `prirast` ter vrne objekt, ki predstavlja novo skladovnico. Kot posledica te operacije se skladovnica `this` izprazni. Velja `kapacitetaPrvega` $\geqslant 1$ in `prirast` $\geqslant 0$.

Testni primer J9

Testni razred:

```
public class Test09 {
    private static final String LOCILO = "--------------------";

    public static void main(String[] args) {
        Skladovnica skladovnica = new Skladovnica(3, 2);
        System.out.println(LOCILO);

        izpisiKapacitete(skladovnica);
        System.out.println(LOCILO);

        System.out.println("Zacetno stanje:");
        izpisiZasedenosti(skladovnica);
        System.out.println(LOCILO);

        System.out.println("Dodamo 30 skatel:");
        skladovnica.dodaj(30);
        izpisiZasedenosti(skladovnica);
```

```
        System.out.println(LOCILO);

        System.out.println("Poskusimo odstraniti 10 skatel:");
        System.out.println(skladovnica.odvzemi(10));
        izpisiZasedenosti(skladovnica);
        System.out.println(LOCILO);

        System.out.println("Poskusimo odstraniti 21 skatel:");
        System.out.println(skladovnica.odvzemi(21));
        izpisiZasedenosti(skladovnica);
        System.out.println(LOCILO);

        System.out.println("Dodamo 22 skatel:");
        skladovnica.dodaj(22);
        izpisiZasedenosti(skladovnica);
        System.out.printf("Skupno stevilo skatel: %d%n",
                          skladovnica.skupnoSteviloSkatel());
        System.out.println(LOCILO);

        System.out.println(skladovnica.poisciKup(2));
        System.out.println(skladovnica.poisciKup(24));
        System.out.println(skladovnica.poisciKup(25));
        System.out.println(skladovnica.poisciKup(37));
        System.out.println(skladovnica.poisciKup(45));
        System.out.println(LOCILO);

        System.out.println("Skatle prelozimo na skladovnico 5/3 ...");
        Skladovnica nova = skladovnica.prestavi(5, 3);
        System.out.print("Nova skladovnica: ");
        izpisiZasedenosti(nova);
        System.out.print("Originalna skladovnica: ");
        izpisiZasedenosti(skladovnica);
        System.out.println(LOCILO);
    }

    private static void izpisiKapacitete(Skladovnica skladovnica) {
        for (int kup = 1;  kup <= 10;  kup++) {
            System.out.printf("| %d ", skladovnica.kapacitetaKupa(kup));
        }
        System.out.println("|");
    }

    private static void izpisiZasedenosti(Skladovnica skladovnica) {
        for (int kup = 1;  kup <= 10;  kup++) {
            System.out.printf("| %d ", skladovnica.zasedenostKupa(kup));
        }
        System.out.println("|");
    }
}
```

Izhod:

```
--------------------
| 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 |
--------------------
Zacetno stanje:
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
--------------------
Dodamo 30 skatel:
```

```
| 3 | 5 | 7 | 9 | 6 | 0 | 0 | 0 | 0 | 0 |
--------------------
Poskusimo odstraniti 10 skatel:
true
| 3 | 5 | 7 | 5 | 0 | 0 | 0 | 0 | 0 | 0 |
--------------------
Poskusimo odstraniti 21 skatel:
false
| 3 | 5 | 7 | 5 | 0 | 0 | 0 | 0 | 0 | 0 |
--------------------
Dodamo 22 skatel:
| 3 | 5 | 7 | 9 | 11 | 7 | 0 | 0 | 0 | 0 |
Skupno stevilo skatel: 42
--------------------
1
4
5
6
-1
--------------------
Skatle prelozimo na skladovnico 5/3 ...
Nova skladovnica: | 5 | 8 | 11 | 14 | 4 | 0 | 0 | 0 | 0 | 0 |
Originalna skladovnica: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
--------------------
```

Oddaja naloge

Oddajte datoteko z nazivom `Skladovnica.java`. V prvi vrstici datoteke v komentarju navedite vašo vpisno številko. Če je, denimo, vaša vpisna številka enaka 63170999, mora datoteka izgledati takole:

```
// 63170999

public class Skladovnica {
    ...
}
```

Testiranje

Program `tj.exe` boste tokrat pognali takole:

`tj.exe <mapa_z_vašim_razredom> <mapa_s_testnimi_razredi> <mapa_za_rezultate>`

Če si želite postopek testiranja karseda poenostaviti, postavite datoteko `Skladovnica.java` v mapo, kjer se nahajajo testni razredi. Znotraj te mape boste namreč lahko program `tj.exe` pognali preprosto takole:

`tj.exe`

To je okrajšava za ukaz

`tj.exe . . .`

kar pomeni, da se vse, tudi bodoči rezultati, nahaja v trenutni mapi. Če se vaš program nahaja v isti mapi kot testni razredi, boste testne razrede lahko prevajali in poganjali tudi ročno (npr. `javac Test01.java` in `java Test01` za prvi testni razred).