

# Programiranje 1 — deseta domača naloga

Rok za oddajo: nedelja, 19. januarja 2020, ob 23:55

## Vreča

### Naloga

Vreča je vsebovalnik, podoben množici, vendar pa lahko vsebuje več kopij istega elementa. Tako kot za množico je tudi za vrečo mogoče definirati operacije, kot so presek, unija in razlika. Na primer, za vreči

$$\begin{aligned}A &= \{a, a, a, a, b, d, d, d, e, e\}, \\B &= \{a, c, c, c, d, d, d, d, e, e\}\end{aligned}$$

velja

$$\begin{aligned}A \cap B &= \{a, d, d, d, e, e\}, \\A \cup B &= \{a, a, a, a, b, c, c, c, d, d, d, d, e, e\}, \\A \setminus B &= \{a, a, a, b\}, \\B \setminus A &= \{c, c, c, d\}.\end{aligned}$$

Napišite razred `Vreca`, ki implementira vrečo z elementi poljubnega referenčnega tipa. Razred naj ne bo generičen; elementi vreče naj bodo deklarirani kot objekti tipa `Object`. Pri implementaciji razreda si **ne smete** pomagati z vgrajenimi javinimi vsebovalniki, torej z vmesnikom `Collection` in njegovimi podvmesniki in podrazredi. Glede na to, da bodo vreče v testnih razredih razmeroma majhne in da tudi število dodajanj in odvzemanj ne bo posebej veliko, lahko vrečo predstavite z navadno tabelo.

V vseh testnih razredih velja sledeče:

- Razred, ki mu (v času izvajanja) pripadajo elementi vreče, implementira vmesnik `Comparable`. Primerjava elementov vreče (vključno s preverjanjem enakosti) naj zato temelji na metodi `compareTo`. Elementa `x` in `y` obravnavamo kot enaka natanko tedaj, ko je rezultat klica

```
((Comparable) x).compareTo((Comparable) y)
```

enak 0. Če je rezultat negativen, obravnavamo element `x` kot manjši od elementa `y`, pozitiven rezultat pa pomeni, da je element `x` večji od elementa `y`.

- Razred, ki mu pripadajo elementi, ne vsebuje »setterjev« in drugih metod za spreminjanje stanja objekta.
- Nobena vreča ne vsebuje več kot 1000 različnih elementov. Število dodajanj in odvzemanj (tj. število klicev metod `dodaj` in `odstrani`) ne bo v nobenem testnem razredu preseglo 1000.
- Vsak element lahko nastopa v kvečjemu  $10^9$  kopijah.

Vreče v skritih testnih razredih 1–5 in 11–15 vsebujejo kvečjemu po eno kopijo vsakega elementa in so tako enakovredne množicam.

Razred `Vreca` naj vsebuje sledeče konstruktorje in metode (oznaka v oglatih oklepajih pove, kateri skriti testni razredi lahko kličejo ta konstruktor ali metodo):<sup>1</sup>

- [1–] `public Vreca()`

Objekt `this` inicializira tako, da predstavlja prazno vrečo.

- [21–] `public Vreca(Vreca vreca)`

Objekt `this` inicializira tako, da predstavlja kopijo podane vreče. Kasnejše spremembe vreče `vreca` ne smejo vplivati na vrečo `this` (in seveda tudi obratno).

- [1–] `public void dodaj(Object element)`

V vrečo `this` doda eno kopijo podanega elementa. (Če v vreči `this` že obstaja element `x` z lastnostjo `x.compareTo(element) == 0`, se število kopij elementa `x` poveča za 1.)

- [6–10, 16–] `public void dodaj(Object element, int stKopij)`

V vrečo `this` doda `stKopij` kopij podanega elementa.

- [11–] `public void odstrani(Object element)`

Iz vreče `this` odstrani eno kopijo podanega elementa (torej elementa `x` z lastnostjo `x.compareTo(element) == 0`). Če vreča ne vsebuje podanega elementa, naj se ne zgodi nič.

- [16–] `public void odstrani(Object element, int stKopij)`

Iz vreče `this` odstrani `stKopij` kopij podanega elementa. Število kopij seveda ne more biti negativno: če je parameter `stKopij` večji ali enak številu kopij elementa v vreči, naj se element iz vreče v celoti odstrani.

- [1–] `public int steviloRazlicnihElementov()`

Vrne število različnih elementov v vreči `this`.

- [6–10, 16–] `public int steviloKopij(Object element)`

Vrne število kopij podanega elementa v vreči `this`.

- [26–] `public String toString()`

Vrne niz oblike

$[e_1/k_1, e_2/k_2, \dots, e_n/k_n]$

pri čemer so  $e_1, \dots, e_n$  elementi vreče `this`, urejeni v naraščajočem vrstnem redu glede na metodo `compareTo`,  $k_1, \dots, k_n$  pa pripadajoča števila kopij.

- [31–] `public Vreca presek(Vreca druga)`

Izdela in vrne nov objekt, ki predstavlja presek vreč `this` in `druga`.

- [36–] `public Vreca razlika(Vreca druga)`

Izdela in vrne nov objekt, ki predstavlja razliko vreč `this` in `druga`.

- [41–] Razred `Vreca` naj implementira vmesnik `Iterable`, in sicer tako, da bo zanka

---

<sup>1</sup>npr. 21– pomeni 21–50

```
// vreca je objekt tipa Vreca
for (Object element: vreca) {
    System.out.println(element);
}
```

izpisala elemente v naraščajočem vrstnem redu glede na metodo `compareTo`. Vsak element naj se izpiše tolikokrat, kolikor kopij ima v množici `this`.

## Javni testni primer

Testni razred (in pripadajoči izhod):

```
public class Test01 {

    public static void main(String[] args) {
        System.out.println("--- prva ---");
        Vreca prva = new Vreca();

        prva.dodaj("cvetko");
        prva.dodaj("eva", 5);
        prva.dodaj("cvetko", 3);
        prva.dodaj("ana", 10);
        prva.dodaj("bojan", 7);
        prva.dodaj("eva", 4);

        prva.odstrani("eva");
        prva.odstrani("ana", 7);
        prva.odstrani("cvetko", 10);
        prva.odstrani("eva", 7);

        System.out.println(prva.steviloRazlicnihElementov());    // 3
        System.out.println();
        System.out.println("ana: " + prva.steviloKopij("ana"));    // 3
        System.out.println("eva: " + prva.steviloKopij("eva"));    // 1
        System.out.println("cvetko: " + prva.steviloKopij("cvetko")); // 0
        System.out.println("bojan: " + prva.steviloKopij("bojan")); // 7
        System.out.println("darja: " + prva.steviloKopij("darja")); // 0
        System.out.println();
        System.out.println(prva.toString());    // [ana/3, bojan/7, eva/1]
        System.out.println();

        for (Object element: prva) {
            System.out.println(element);
        } // ana ana ana bojan bojan bojan bojan bojan bojan eva
        System.out.println();

        System.out.println("--- druga ---");
        Vreca druga = new Vreca(prva);
        druga.odstrani("bojan", 3);
        druga.dodaj("franci", 5);
        druga.dodaj("eva", 2);
    }
}
```

```

        druga.odstrani("ana", 4);
        System.out.println(druga.toString()); // [bojan/4, eva/3, franci/5]
        System.out.println();

        System.out.println("--- tretja ---");
        Vreca tretja = prva.presek(druga);
        System.out.println(tretja.toString()); // [bojan/4, eva/1]
        System.out.println();

        System.out.println("--- cetrti ---");
        Vreca cetrti = prva.razlika(druga);
        System.out.println(cetrti.toString()); // [ana/3, bojan/3]
        System.out.println();

        System.out.println("--- prva ---");
        System.out.println(prva); // [ana/3, bojan/7, eva/1]
    }
}

```

## Oddaja naloge

Oddajte datoteko z nazivom `Vreca.java`. V prvi vrstici datoteke v komentarju navedite svojo vpisno številko. Če je, denimo, vaša vpisna številka enaka 63190999, mora datoteka izgledati takole:

```

// 63190999

// definicija razreda Vreca
// ...

```

Morebitne pomožne razrede definirajte kot statične notranje razrede v razredu `Vreca`.