4. laboratorijska vježba (11 bodova)

Važna napomena: *u svim zadacima potrebno je napisati Javadoc komentare za svaki razred te generirati dokumentaciju. Svi nazivi razreda, metoda i varijabli moraju biti na engleskom. Sav napisani programski kod mora biti napisan u skladu s konvencijama imenovanja varijabli, metoda i razreda (varijable i metode: malo početno slovo,* camel-case*; razredi i sučelja: veliko početno slovo,* camel-case*; konstante: uobičajeno sve veliko i razdvajanje podvlakom) te ostalim pozitivnim praksama (uključivo i korektno uvlačenje redaka; smisleno razdvajanje više različitih semantički grupiranih redaka praznim retcima, pravilnim razmještajem otvorene i zatvorene vitičaste zagrade i slično). Za više informacija pogledajte* ***http://www.oracle.com/technetwork/java/codeconventions-150003.pdf***

Zadatak 1.

Na stranici predmeta dostupna je biblioteka `picture.jar` te zip-arhiva picture-doc.zip koja sadrži pripadnu dokumentaciju (javadoc). Uključite biblioteku u projekt i podesite IDE tako da Vam prikazuje dokumentaciju iz ZIP arhive. Biblioteka sadrži implementaciju razreda `hr.fer.oop.lab4.pic.Picture` koji predstavlja crno-bijelu sliku te razreda `hr.fer.oop.lab4. pic.PictureDisplay` koji nudi mogućnost prikaza takve slike na standardnom izlazu u ASCII grafici ili pak na ekranu u zasebnom prozoru. Proučite pripadnu dokumentaciju.

Napišite razrede `EquilateralTriangle`, `Circle` i `Rectangle` koji modeliraju jednakostranični trokut, krug i pravokutnik. Pretpostavite da jednakostranični trokut ima bazu koja je usporedna s osi *x* te visinu koja raste u smjeru negativne osi *y* (prema gore na ekranu). Sve razrede smjestite u paket `hr.fer.oop.lab4.prob1`.

Opremite napisane razrede adekvatnim konstruktorima. Omogućite korisniku da svaki od likova stvara predavanjem svih potrebnih parametara kroz konstruktor ili pak predavanjem reference na neki drugi takav lik, u kojem slučaju treba preuzeti parametre iz tog lika (naravno, `Circle`-u se može predati samo drugi `Circle` i slično za ostale).

Svaki od razreda opremite metodom `drawOnPicture` koja prima referencu na sliku i na njoj paljenjem slikovnih elemenata crta taj lik (prikazuje se čitava ispuna lika). Iscrtavanje implementirajte na način da slijedno uzimate točku po točku slike. Ukoliko je ta točka unutar geometrijskog lika, slikovni element je potrebno upaliti, a inače ne. U slučaju da dio lika izlazi izvan područja slike, potrebno je nacrtati samo dio koji je vidljiv (ne smiju se događati pogreške odnosno iznimke). Primjetite da je legalno stvoriti geometrijski lik koji čitavom površinom nije prikaziv na slici odabranih dimenzija.

Potom iz razreda `EquilateralTriangle`, `Circle` i `Rectangle` izvedite razrede `EquilateralTriangleFast`, `CircleFast` i `RectangleFast` koji nadjačavaju metodu `drawOnPicture` i nude efikasniju implementaciju koja ne proziva sve slikovne elemente slike već se oslanja na metodu `drawLine` koju nudi razred `Picture`.

Napišite glavni program `Demonstration` (u istom paketu) koji stvara jednu sliku dimenzija 100x50, stvara po dva primjerka svakog od zadanih razreda (likova), crta ih na slici te sliku na kraju prikazuje na standardnom izlazu (`System.out`). Poslužite se prikladnim razredima iz biblioteke `picture.jar`. Da biste to mogli napraviti, uključite `picture.jar` u projekt.

Na vježbi vas možemo pitati da izvorni kod prevedete i pokrenete direktno iz naredbenog retka.

Zadatak 2.

Napišite implementaciju parametriziranog razreda `SimpleHashtable<K,V>`. Razred predstavlja tablicu raspršenog adresiranja koja omogućava pohranu uređenih parova (ključ, vrijednost). Parametar `K` predstavlja tip ključa a parametar `V` tip vrijednosti. Postoje dva javna konstruktora: podrazumijevani (default) koji stvara tablicu veličine 16 slotova, te konstruktor koji prima jedan argument: broj koji predstavlja željeni početni kapacitet tablice i koji stvara tablicu veličine koja je potencija broja 2 koja je prva veća ili jednaka predanom broju (npr. ako zada 30, bira se 32).

Jedan slot tablice modelirajte pomoćnim razredom `TableEntry<K,V>`. Primjerci razreda `TableEntry<K,V>` imaju člansku varijablu `key` u kojoj pamte predani ključ, člansku varijablu `value` u kojoj pamte pridruženu vrijednost te člansku varijablu `next` koja pokazuje na sljedeći primjerak razreda `TableEntry<K,V>` koji se nalazi u istom slotu tablice (izgradnjom ovakve liste rješavat ćete problem preljeva – situacije kada u isti slot treba upisati više uređenih parova). Svojstvo `key` u razredu `TableEntry<K,V>` ne smije se moći mijenjati (mora biti *read-only*).

Ideju uporabe ovakve kolekcije ilustrira sljedeći kod.

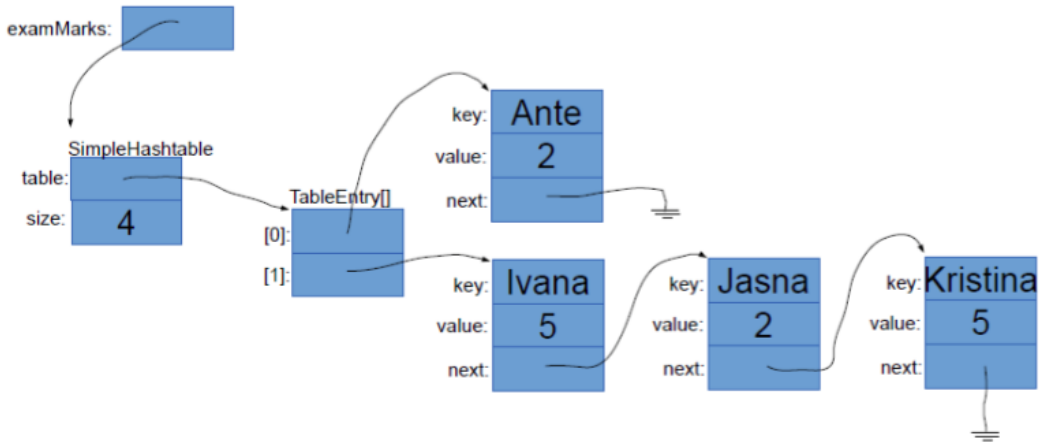
```
// create collection:
SimpleHashtable<String,Integer> examMarks = new SimpleHashtable<>(2);

// fill data:
examMarks.put("Ivana", Integer.valueOf(2));
examMarks.put("Ante", Integer.valueOf(2));
examMarks.put("Jasna", Integer.valueOf(2));
examMarks.put("Kristina", Integer.valueOf(5));
examMarks.put("Ivana", Integer.valueOf(5)); // overwrites old grade for Ivana

// query collection:
Integer kristinaGrade =  examMarks.get("Kristina");
System.out.println("Kristina's exam grade is: " + kristinaGrade); // writes: 5

// What is collection's size? Must be four!
System.out.println("Number of stored pairs: " + examMarks.size()); // writes: 4
```

Za potrebe izračuna slota u koji treba ubaciti uređeni par koristite metodu `hashCode()` ključa, pa modulo veličina tablice. Ključ uređenog para ne smije biti `null` dok vrijednost može biti `null`. Ako ključ u metodi `put()` bude `null`, neka metoda ne radi ništa. Ako bude `null` u metodi `get()`, neka metoda vrati `null`.

Razred `SimpleHashtable<K,V>` treba imati sljedeće članske varijable:

- `TableEntry<K,V>[] table`: polje slotova tablice,
- `int size`: broj parova koji su pohranjeni u tablici.

Pojednostavljeni prikaz stanja u memoriji nakon izvođenja koda iz prethodnog primjera ilustriran je na sljedećoj slici.

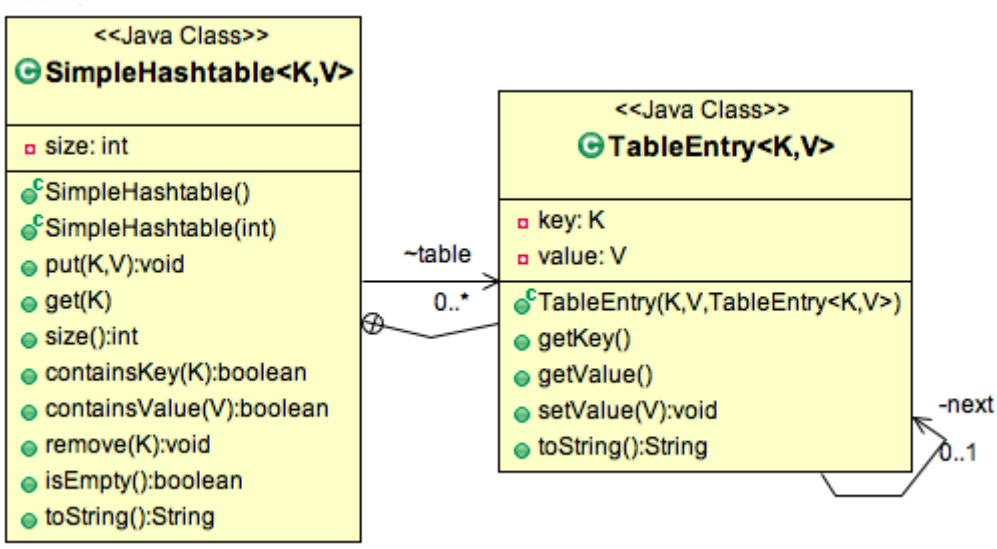


Pri tome na platformi Java 8 vrijedi:

objekt	**hashCode()**	**|hashCode()|**	**slot=|hashCode()| % 2**
"Ivana"	71029095	71029095	1
"Ante"	2045822	2045822	0
"Jasna"	71344303	71344303	1
"Kristina"	-1221180583	1221180583	1

Stoga će ključevi Ivana, Jasna i Kristina biti u slotu 1, a ključ Ante u slotu 0. Razmislite odgovara li prikazana slika stvarnom stanju u memoriji ili bismo za stvarno stanje dio slike trebali drugačije nacrtati?

Dijagram razreda koji prikazuje razrede ovog zadatka prikazan je u nastavku.



Metode i konstruktori koje razred `SimpleHashtable<K,V>` mora ponuditi navedeni su u nastavku bez posebne dokumentacije (iz imena bi moralo biti jasno što se od metode očekuje).

```
public SimpleHashtable();
public SimpleHashtable(int capacity);
public void put(K key, V value);
public V get(K key);
public int size();
public boolean containsKey(K key);
```

```
public boolean containsValue(V value);
public void remove(K key);
public boolean isEmpty();
public String toString();
```

Metoda `put` pozvana s ključem koji u tablici već postoji ažurira postojeći par novom vrijednošću; metoda ne dodaje još jedan par s istim ključem ali drugom vrijednosti.

Metoda `get` pozvana s ključem koji ne tablici ne postoji vraća `null`.

Što možete zaključiti o složenosti metode `containsKey` a što o složenosti metode `containsValue` u ovako implementiranoj kolekciji (uz pretpostavku da je broj parova dodanih u tablicu dosta manji od broja slotova tablice te da funkcija sažetka radi dobro raspršenje)?

Razred TableEntry definirajte kao pomoćni razred unutar razreda SimpleHashTable kao što smo to već radili u drugim primjerima (počev od proceduralnog programiranja gdje smo takve razrede koristili kao pomoćne strukture podataka): samo mu u deklaraciji dodajte ključnu riječ static:

```
public class SimpleHashtable ... {
  private static class TableEntry ... { // pomoćni razred
    ...
  }
  ...
}
```

(tri točkice označavaju mjesta gdje nedostaje dio koda).

Zadatak 3.

Za predmet u osnovnoj školi, nastavnik Ivo treba pomoć. Za sve učenike kojima je držao nastavu, Ivo ima zapise oblika *ime učenika*, *ocjena* (pretpostavimo da u razredu nema više učenika istog imena). Potrebno je napisati program koji će Ivi omogućiti da na kraju polugodišta prepiše te podatke (vidi primjer u nastavku). Nakon toga, program treba na zaslon ispisati za svakog učenika njegovo ime, broj danih ocjena, ocjene (redoslijedom unosa), sve različite ocjene (sortirano od manje prema većoj), prosječnu ocjenu te standardno odstupanje (standardna devijacija populacije; vidi https://www.easycalculation.com/statistics/standard-deviation.php). Program zapise treba čitati preko tipkovnice, jedan zapis po retku. Čitanje prestaje kada Ivo upiše KRAJ. Evo primjera.

```
Jasna 5
Ante 5
Jasna 4
Stjepan 3
Ante 5
Stjepan 4
Jasna 5
Ante 3
Jasna 4
Stjepan 4
KRAJ

Učenik Jasna
  Broj ocjena: 4
  Ocjene: 5 4 5 4
  Različite ocjene: 4 5
  Prosječna ocjena: 4.5
  Standardno odstupanje: 0.5

Učenik Ante
```

```
  Broj ocjena: 3
  Ocjene: 5 5 3
  Različite ocjene: 3 5
  Prosječna ocjena: 4.33333
  Standardno odstupanje: 0.94281

Učenik Stjepan
  Broj ocjena: 3
  Ocjene: 3 4 4
  Različite ocjene: 3 4
  Prosječna ocjena: 3.66667
  Standardno odstupanje: 0.4714
```

Podatke je potrebno ispisati redoslijedom kojim se ime učenika prvi puta pojavi na popisu. Za rješavanje ovog zadatka koristite Java Collection Framework.