Uvod in predstavitev jezika Java

1. Razcep števil na prafaktorje

Reši zadnjo nalogo iz prejšnjih vaj.

2. Promet skozi predor Golovec

Podana je datoteka `golovec.txt` s **podatki o prometu skozi predor Golovec** za izbran dan. V vsaki vrstici sta najprej zapisani celi števili s in f, ki predstavljata sekundi v dnevu, ko je vozilo vstopilo oziroma izstopilo iz predora, $0 \leq s < f \leq 86400$. Nato sledi niz znakov, ki predstavlja registrsko številko vozila. Vse tri vrednosti so ločene s presledki. Predor Golovec je dolg $622\ m$, dočim je omejitev hitrosti $80\ km/h$.

V programskem jeziku Java najprej **sestavite razred** `Drive`, ki naj **predstavlja vožnjo vozila** (npr. skozi predor Golovec). Vožnja naj bo predstavljena s štirimi objektnimi spremenljivkami, tj. začetek in konec vožnje v sekundah `int start, finish`, razdaljo v m `int distance` in registrsko številko vozila `String registration`. Vse spremenljivke naj bodo privatne, zato poskrbite za potrebne `get` funkcije in morda dodajte še funkcijo `double getSpeed()`, ki vrne povprečno hitrost vožnje v km/h.

Nato **sestavite statično funkcijo** `int speeding(File in, File out, int distance, double limit) throws IOException`, ki sprejme vhodno datoteko `in` s podatki o prometu (npr. `golovec.txt`) in ustvari izhodno datoteko `out` (npr. `speeding.txt`), v katero zapiše rezultate. Parameter `distance` naj predstavlja dolžino vožnje v m, parameter `limit` pa omejitev hitrosti v km/h. Funkcija naj najprej iz datoteke `in` **prebere rezultate o vožnjah**, ki naj bodo predstavljene kot objekti razreda `Drive`, in nato **poišče vozila, ki so zagotovo vozila prehitro** (t.j. povprečna hitrost vozila je bila nad omejitvijo). Funkcija naj v datoteko `out` **zapiše podatke o vozilih, ki so kršila omejitev**. Vrstice v datoteki `out` naj vsebujejo registrsko številko vozila in njegovo povprečno hitrost na dve decimalni mesti natančno. Funkcija naj tudi **vrne število vseh kršiteljev** omejitve hitrosti in naj bo definirana v razredu `Speeding`.

Delovanje razredov `Drive` in `Speeding` preizkusite s pomočjo naslednjega programa, ki ga vključite v metodo `void main(String[] args)` v razredu `Speeding`.

```java
try {
    System.out.println(speeding("golovec.txt", "speeding.txt", 622, 80.0));
} catch (IOException e) {
    e.printStackTrace();
}
```

Pričakovan izpis zgornjega programa je podan spodaj.

```
4130
```

Pričakovana vsebina datoteke `speeding.txt` je podana spodaj.

```
KK612ER 86.12
MB703KW 106.63
GO366FA 93.30
GO893IZ 124.40
GO583QI 82.93
LJ084ST 82.93
LJ771QL 86.12
KP832GP 89.57
NM233JK 82.93
CE458BG 86.12
CE705BG 106.63
PO854MT 93.30
MS935US 97.36
KP007MZ 86.12
MS626CX 106.63
KR478GW 111.96
CE069NW 82.93
CE110NF 86.12
LJ771FR 86.12
KR091XR 93.30
LJ000JU 93.30
GO664HU 89.57
LJ439WT 106.63
GO908MG 124.40
NM244DN 117.85
...
```

3. Histogram naključnih števil

V programskem jeziku Java najprej **sestavite statično metodo** `void write(File file) throws IOException`, ki v datoteko `file` **zapiše naključno izbrana cela števila** iz intervala $[0, M)$, pri čimer je vrednost M določena v programu s statično konstanto `MAXIMUM`. Datoteka naj vsebuje 1000 vrstic, dočim naj vsaka vrstica ne vsebuje več kot 5 naključno izbranih števil ločenih s presledkom. Nato **sestavite statično funkcijo** `List<Integer> read(File file) throws IOException`, ki iz datoteke `file` **prebere predhodno zapisana števila** kot opisano zgoraj in vrne seznam števil. Metoda in funkcija naj bosta definirani v razredu `Histogram`.

Nato **sestavite sam razred** `Histogram`, ki naj predstavlja **histogram pozitivnih celih števil**. Histogram naj bo predstavljen s štirimi objektnimi spremenljivkami, tj. število števil `int size`, širina stolpcev histograma `int interval`, največje možno število `int maximum` in tabela frekvenc števil po posameznih stolpcih oziroma intervalih `int[] histogram`. Vse spremenljivke naj bodo privatne, zato poskrbite za potrebne `get` funkcije.

Konstruktor razreda `Histogram` definirajte kot `Histogram(List<Integer> numbers, int interval, int maximum)`, kjer prvi parameter predstavlja seznam pozitivnih celih števil za gradnjo histograma `histogram`, zadnji dve spremenljivki pa sta zgornji objektni spremenljivki s privzetimi vrednostmi 12 in statično konstanto `MAXIMUM`. Pazite, da je zadnji interval histograma `histogram` lahko krajši od `interval`, v kolikor `maximum` ni deljivo z `interval`.

Delovanje razreda `Histogram` preizkusite s pomočjo naslednjega programa.

Java

```java
public class Histogram {

  ...

  @Override
  public String toString() {
    String string = "";
    for (int i = 0; i < histogram.length; i++)
      string += (i > 0? "\n": "") + String.format("%10s %3d - %4.1f%%", "[" + i * interval + "," + Math.min(maximum,

    return string;
  }

  static final int MAXIMUM = 123;
```

Java

```java
  public static void main(String[] args) {
    List<Integer> numbers = null;
    try {
      write("numbers.txt");

      numbers = read("numbers.txt");
    } catch (IOException e) {
      e.printStackTrace();

      System.exit(0);
    }

    Histogram histogram = new Histogram(numbers);

    System.out.println(histogram);
  }

}
```

Primer vsebine datoteke `numbers.txt` je podan spodaj.

```
21 63 69 87
21 79
109 109 84
104 7 87
86 59
52 16 29
69 64 33
94 23
...
```

Primer izpisa zgornjega programa je podan spodaj.

```
   [0,12): 271 -  9.1%
  [12,24): 266 -  8.9%
  [24,36): 289 -  9.7%
  [36,48): 320 - 10.7%
  [48,60): 293 -  9.8%
  [60,72): 283 -  9.5%
  [72,84): 285 -  9.6%
  [84,96): 324 - 10.9%
 [96,108): 283 -  9.5%
[108,120): 292 -  9.8%
[120,123):  72 -  2.4%
```