Java Oefeningen – Reeks 1 (extra)

Vraag 3

Voor de laatste oefening gaan we een programma schrijven die statistieken kan uitrekenen op een reeks getallen. Gebruik hiervoor de lijsten op de website[1] (Reeks 1 > Data). Noem de klasse "***Statistiek***".

1. Maak twee nieuwe methodes: **"gemiddelde"** en **"stdafw"**. Deze nemen beiden als input een array **int[]** en geven als returnwaarde een ***double*** (respectievelijk het gemiddelde en de standaardafwijking van de meegegeven array getallen).
2. Maak vervolgens de methodes **"minimum"** en **"maximum"**, die respectievelijk het minimum en het maximum van de input-array teruggeeft. Je mag ervan uitgaan dat de elementen van de lijst altijd positief zijn. Geef een 0 terug als de array leeg is.
3. Maak de methode **"tweedegrootste"**, die het op één na grootste element uit de input-array geeft. Geef een 0 terug als de array 1 of geen elementen bevat.
4. Maak een methode **"histogram"** die een histogram tekent met 10 categorieën. Print de output uit zoals in het voorbeeld hieronder aangegeven:

```
10-15: ****
15-20: *****
20-25: ***
25-30: *********
30-35: ***
35-40: ***********
40-45: **********
45-50: **************
50-55: ******
55-60: *
```

Zorg dat de categoriegroottes zich aanpassen aan de gebruikte lijst.

[1] http://parallel.vub.ac.be/education/java/practica/index.html

Vraag 4

Voor deze oefening gaan we een programma schrijven die een veralgemeende Fibonaccireeks uitrekent. De definitie van de Fibonaccireeks van orde k is de volgende:

$$F_n^k := \begin{cases} n < k & 0 \\ n = k & 1 \\ n > k & \sum_{i=n-k}^{n-1} F_i^k \end{cases}$$

Voor $k = 2$ krijg je dus de klassieke Fibonaccireeks.

1. Maak een nieuw package aan onder het project '***Reeks1'***.
2. Maak hierin een nieuwe klasse met naam '***Fibonacci'***, inclusief een main() methode.
3. Creëer een methode die als input twee ***int***s krijgt: **k** voor de orde van de reeks, en **n** voor het aantal elementen dat je wil uitrekenen. De methode moet als output een **int[]** teruggeven met de juiste waarden.
4. Roep de methode op in de main() methode, en print deze vervolgens af.
5. Zorg ervoor dat je methode correct werkt voor alle waardes van **n** en **k**. Print een foutmelding af indien **n<0** of **k<1**.

Vraag 5

Voor deze oefening gaan we operaties doen op priemgetallen: dit zijn gehele getallen die precies 2 verschillende delers hebben (namelijk 1 en zichzelf). Maak hiervoor een nieuwe package aan.

1. Maak een methode **boolean is_priem(int n)**, met als invoer een integer **n**, en als uitvoer een boolean. Deze functie geeft terug of **n** al dan niet een priemgetal is.
2. Maak een methode **int[] circular_shift(int n)**. Deze functie geeft een array terug met alle combinaties van circulaire cijferverschuivingen van **n**, waarbij telkens alle cijfers naar links worden opgeschoven, en het eerste cijfer op de laatste plaats gezet wordt. Enkele voorbeelden: (Tip: gebruik hiervoor *Math.log10*)
    a. 7865 geeft: {7865, 8657, 6578, 5786}
    b. 340 geeft: {340, 403, 034 (of 34)}
    c. 1127 geeft: {1127, 1271, 2711, 7112}
3. Schrijf een methode **boolean circ_priem(int n)**, die gebruikmaakt van de vorige methodes om te bepalen of **n** al dan niet een “circulair priemgetal” is; d.i. een getal waarvan al zijn circulaire cijferverschuivingen ook priemgetallen zijn
4. Print uit hoeveel circulaire priemgetallen er zijn die kleiner zijn dan 10^5.