# Fra reeksamen F16.

# Opg. 4: Navne på piger og drenge (tæller 25 %)

Filerne *Boys.txt* og *Girls.txt* indeholder oplysninger om de 1000 mest populære drenge- og pigenavne, for babyer født i USA i 2014 (taget fra *http://www.babynamewizard.com/the-top-1000-baby-names-of-2014-united-states-of-america*).

Filerne er *tabulator* separerede og hver linje repræsenterer ét navn med 4 oplysninger:

placering2014 navn antal placering2013

I denne opgave skal vi finde de navne, som er på begge lister.

## Opg. 4a CommonName implements Comparable<CommonName> ( 6 %)

Klassen repræsenterer ét navn, som er på begge lister over mest brugte navne. Færdiggør klassen så den indeholder:

* private variable til name (String), girls (int) og boys (int).
* Constructor, som tager 3 parametre, svarende til de 3 variable.
* getName() metode, som returnerer name.
* getTotal() metode som returnerer det samlede antal piger og drenge.
* toString() metode. Denne er implementeret i det udleverede kodeskelet; men er udkommenteret. Fjern udkommenteringen, når de ovenstående punkter er implementeret.
* compareTo() metode, som sammenligner total antal piger og drenge (med getTotal()). Hvis to instanser af klassen har samme total, skal der sammenlignes på navnene.
* main() metode til test. Her skal udkommenteringerne også blot fjernes, efterhånden som arbejdet skrider frem.

Eksempel på korrekt output (navnene her er noget vrøvl og altså ikke taget fra filerne):

Usorteret:

[Ibi Girls: 10 Boys: 20 Total: 30

, Ib Girls: 20 Boys: 10 Total: 30

, Ea Girls: 20 Boys: 20 Total: 40

, Eva Girls: 20 Boys: 5 Total: 25

]

Sorteret:

[Ea Girls: 20 Boys: 20 Total: 40

, Ib Girls: 20 Boys: 10 Total: 30

, Ibi Girls: 10 Boys: 20 Total: 30

, Eva Girls: 20 Boys: 5 Total: 25

]

## Opg. 4b GirlsAndBoys ( 12 %)

Klassen er delvis implementeret. Den indeholder allerede:

* private Map<String, Integer> girlsMap; pigenavn og antal.
* private Map<String, Integer> boysMap; drengenavn og antal.
* private Set<CommonName> commonSet; instanser af CommonName for navne, som er i begge mapper.
* Constructor, som 2 gange kalder createNameMap(File file), for at indlæse data fra de 2 filer i de to mapper.
* toString() metode, som giver en ”pæn” udskrift af commonSet.
* main() metode til test af klassen.

**Implementer disse metoder:**

private Map<String, Integer> createNameMap(File file):

* Erklær en lokal instans af Map<String, Integer>, og initialiser den til en standard implementation, som sikrer *hurtigst mulig indsættelse* og *søgning*.
* Åben input filen og læs den én linje ad gangen.
* Indsæt sammenhørende værdier (*navn, antal*) i mappen.
* Benyt exception-handling inde i metoden. Brug enten *try-with-ressource* eller eksplicit lukning af filen efter brug.
* Returner mappen.

public void makeCommonNames():

* Initialiser commonSet, så det hele tiden er *sorteret* ved hjælp af compareTo() -metoden i CommonName.
* Gennemløb én af de to mapper.
* Hvis et navn også optræder i den anden mappe, instancieres et objekt af typen CommonName (med navn, antal piger og antal drenge), som indsættes i commonSet.

public void write2file(File f):

* Benyt en FileWriter eller lignende til at skrive i filen.
* Gennemløb commonSet og benyt toString()-metoden fra CommonName til at skrive én linje pr. navn.
* Benyt exception-handling inde i metoden og brug enten *try-with-ressource* eller eksplicit lukning af filen efter brug.

**Eksempel.** Med den udleverede main()-metode bør de første 5 linjer i filen nu indeholde:

Logan Girls: 828 Boys: 13579 Total: 14407

Jayden Girls: 569 Boys: 12878 Total: 13447

Avery Girls: 9517 Boys: 2269 Total: 11786

Dylan Girls: 822 Boys: 10350 Total: 11172

Carter Girls: 355 Boys: 10599 Total: 10954

## Opg. 4c Sortering med Comparator ( 7 %)

* Opret en klasse med signaturen

public class CommonNamesComparator implements Comparator<CommonName>.

* Implementer compare()-metoden, så objekter af typen CommonName sammenlignes på name.
* Implementer public void sortCommonByName(Comparator<CommonName> comparator) i GirlsAndBoys, så der erklæres et nyt Set, som holdes *sorteret* med comparatoren.
  + Indsæt alle elementer fra commonSet i det nye set.
  + Lad commonSet variablen pege på det nye set.

**Eksempel**. Med den udleverede main()-metode bør de første 5 linjer i filen nu indeholde:

Alexis Girls: 4188 Boys: 1172 Total: 5360

Amari Girls: 585 Boys: 970 Total: 1555

Angel Girls: 1179 Boys: 6255 Total: 7434

Ariel Girls: 2419 Boys: 319 Total: 2738

Armani Girls: 422 Boys: 570 Total: 992