# Opgave 1: Generering af brugernavne 25%

*Udleveret kodeskelet:* User.java

I denne opgave skal der dannes brugernavne ud fra det fulde navn, efter regler, som er lidt inspireret af måden brugernavne tildeles til studerende på SDU.

Opg. a) 4%

Tilføj følgende til klassen User:

* 2 private felter, String fullName og Date createdAt.
* Constructoren skal have signaturen public User(String name), hvor name skal initialisere fullName og createdAt skal sættes til tidspunktet hvor constructoren er kaldt.
* Overskriv public String toString(), så objekter af klassen kan udskrives

Test af klassen med den udleverede main() metode bør give et resultat, som ligner dette:

User: Joachim Holger Waldemar Christian - Fri May 25 13:18:08 CEST 2012

User: Ib Bo - Fri May 25 13:18:08 CEST 2012

User: Åge Bøgeskov - Fri May 25 13:18:08 CEST 2012

Opg. b) 10%

Implementer public String makeUserName(), så der ud fra feltet fullName returneres et brugernavn efter følgende regler:

* et brugernavn forsøges dannet af de første 2 bogstaver i fornavnet og de første 3 bogstaver i efternavnet. Eventuelle mellemnavne ignoreres og store bogstaver ændres til små:
  + ”Joachim Holger Waldemar Christian” bliver ”jochr”.
* hvis ikke der er 2 bogstaver i fornavnet eller 3 i efternavnet, benyttes alle bogstaver:
  + ”Ib Bo” bliver ”ibbo”.
* hvis det dannede brugernavn indeholder ’æ’, ’ø’ eller ’å’, skal disse karakterer erstattes af hhv. ”ae”, ”oe” og ”aa”:
  + ”Åge Bøgeskov” bliver ”aagboeg”.

Metoden kan testes ved at fjerne udkommenteringen i main()-metoden.

Opg. c) 6%

Der skal nu implementeres en klasse, public class UserMap, som kun indeholder metoden

public static void main(String[] arg). Der skal:

* Erklæres en variabel af typen java.util.Map<String, User>
* Den skal initialiseres, så opslag i mappen bliver så hurtig som mulig.
* I et loop skal der promptes for indtastning af fuldt navn på brugere:
  + For hvert indtastet navn, oprettes en User user, som indsættes i mappen med   
    user .makeUserName() som nøgle.
  + Når der indtastes en tom linje, afsluttes loop’et og mappen udskrives. Det kan ligne:

Enter full name:

Joachim Holger Waldemar Christian

Ib Bo

Åge Bøgeskov

{aagboeg=User: Åge Bøgeskov - Fri May 25 15:30:14 CEST 2012, jochr=User: Joachim Holger Waldemar Christian - Fri May 25 15:29:56 CEST 2012, ibbo=User: Ib Bo - Fri May 25 15:30:04 CEST 2012}

Opg. d) 5%

Der er nu risiko for konfliktet hvis der er flere brugere, som genererer ens brugernavne.

Implementer en strategi til at undgå dette. Fx. Kan det tjekkes om et netop genereret brugernavn allerede findes i mappen og tilføje et løbenummer i så tilfælde. Det kan give følgende output:

Enter full name:

Lars Schmit

Lasse Schmeigel

Lauritz Emil Schmalhals

{lasch=User: Lars Schmit - Fri May 25 14:56:31 CEST 2012, lasch2=User: Lauritz Emil Schmalhals - Fri May 25 14:57:06 CEST 2012, lasch1=User: Lasse Schmeigel - Fri May 25 14:56:45 CEST 2012}

# Opgave 2: Bjerge i Frankrig 25%

*Udleveret kodeskelet: Mountain.java*

*Udleveret tekst fil: FranskeBjerge.csv*

*FranskeBjerge.csv* er en semikolon-separeret tekst fil, hvor hver linje giver oplysninger om de bjerge i Frankrig, som har en *topographic prominence* højere end 1000 meter (*topographic prominence* er den relative højde af en bjergtop, se fig. 1, som er oversat fra en.wikipedia.org):

I filen er et bjerg beskrevet ved:

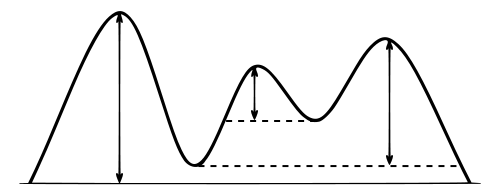


Fig. 1: De lodrette linjer viser den *topografiske prominence* af 3 bjergtoppe på en ø. De stiplede vandrette linjer angiver den laveste højdekurve, som ikke omslutter en højere bjergtop.

* name: Bjergets navn
* height: Højde over havets overflade
* prominence: Relativ højde (fig. 1)
* latitude: Breddegrad i Grader°Minutter’Sekunder”
* longitude: Længdegrad i Grader°Minutter’Sekunder”
* range: Bjergkæde

Som eksempel er her linjen for *Mount Ventoux*:

Mont Ventoux;1909;1148;44°10'26";05°16'42";Alps

I denne opgave skal filen indlæses og oplysningerne indsættes i en datastruktur. I opgaverne a) og b) skal der implementeres en klasse med signaturen public class Mountain implements Comparable. Instanser af klassen kan holde informationerne om ét bjerg. Det udleverede kodeskelet indeholder en main-metode, som kan bruges til at teste klassen med.

a) 5%

I constructoren til Mountain, skal de 6 felter initialiseres og public String toString()overskrives, så testkørsel af programmet giver et output, som ligner følgende:

Mont Ventoux h=1909, pro=1148, lat=44°10'26", lon=05°16'42", ran=Alps

b) 8%

Implementer public int compareTo(Mountain o), så instanser af Mountain kan ordnes efter prominence. Hvis flere bjergtoppe har samme prominence, skal disse ordnes efter height.

Ved at fjerne udkommenteringen af koden i main-metoden, skal der opnås et output som ligner dette:

Usorteret:

[Pica d'Estats h=3143, pro=1281, lat=42°42'43", lon=00°57'23", ran=Pyrenees

, Mont Chaberton h=3131, pro=1281, lat=44°57'53", lon=06°45'06", ran=Alps

, Pic de Bure h=2709, pro=1268, lat=44°37'38", lon=05°56'07", ran=Alps

, Pic du Midi d'Ossau h=2886, pro=1092, lat=42°48'22", lon=-00°25'05", ran=Pyren

ees

]

Sorteret:

[Pic du Midi d'Ossau h=2886, pro=1092, lat=42°48'22", lon=-00°25'05", ran=Pyrene

es

, Pic de Bure h=2709, pro=1268, lat=44°37'38", lon=05°56'07", ran=Alps

, Mont Chaberton h=3131, pro=1281, lat=44°57'53", lon=06°45'06", ran=Alps

, Pica d'Estats h=3143, pro=1281, lat=42°42'43", lon=00°57'23", ran=Pyrenees

]

c) 12%

Implementer en klasse med signaturen public class MountainSet.

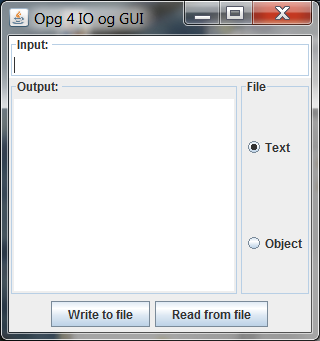
Klassen skal indkapsle en datastruktur af typen Set<Mountain>, en constructor samt en metode med signaturen

public void add(Mountain m), som benyttes til indsætning af ét bjerg ad gangen i settet, som skal holdes sorteret vha. compareTo() metoden i klassen Mountain.

Implementer også public static void main(String[] args)efter denne algoritme:

* erklær og initier en instans af MountainSet
* læs *FranskeBjerge.csv* en linje ad gangen
* for hver linje oprettes en instans af Mountain, som indsættes vha. add(Mountain m)

# Opgave 3: GUI, Polymorphism og lidt File I/O 30%

*Udleveret kode skelet: FileIOGUI.java*

*FileIOInterface.java*

a) 8%

Implementer metoden makeComponents() i FileIOGUI, så der fremkommer en brugerflade som vist på figur 2 indeholdende:

* Øverst et tekstfelt med en titledBorder, visende teksten ”Input”.
* Til højre et panel med en titledBorder, visende teksten ”File”. Panelet skal indeholde 2 JRadioButtons, til valg af hhv. ”Text” eller ”Object”.
* Nederst et panel med 2 knapper, ”Write to file” og ”Read from file”
* Resten af brugerfladen skal indeholde et JTextArea indeholdt i et JScrollPane med en titledBorder, visende teksten ”Output”.

FileIOInterface.java er et interface, som erklærer to metoder void writeToFile(String text) og String readFromFile().

Der skal i de følgende implementeres 2 klasser, hhv. TextFile og ObjFile, som implementerer dette interface:

b) 6%

public class TextFile implements FileIOInterface

Der skal erklæres et statisk felt af typen File, som holder en reference til en fil med navnet "Opg4File.txt". Metoden writeToFile(String text), skal tilføje text efterflugt af ”\n” i filen.

Metoden readFromFile() skal læse hele filen linje for linje, danne en String heraf og returnere denne (husk ”\n” mellem hver linje i resultatet).

c) 6%

public class ObjFile implements FileIOInterface, Serializable

Der skal erklæres et statisk felt af typen File, som holder en reference til en fil med navnet "Opg4File.obj".

Desuden skal klassen have et privat felt String txt, samt en getTxt() metode til returnering af txt.

Metoden writeToFile(String text) skal:

* sætte txt til text
* gemme hele objektet i filen ved navn "Opg4File.obj"

Metoden readFromFile() skal læse et objekt af typen ObjFile og returnere det indlæste vha. metoden getTxt().

d) 10%

FileIOGUI indeholder en indre klasse, ButtonListener implements ActionListener. En instance af denne skal benyttes af de to knapper ”Write to file” og ”Read from file”. Der skal benyttes polymorfi:

* Et privat felt af typen FileIOInterface fileIO er allerede erklæret i den udleverede fil
* implementer public void actionPerformed(ActionEvent e), så fileIO initialiseres til en ny instance af TextFile eller ObjFile, afhængigt af hvilken radioknap der er valgt.
* Hvis der er klikket på ”Write to file” , kaldes fileIO.writeToFile(String text) med teksten fra Input-feltet
* Hvis der er klikket på ”Read from file” , kaldes fileIO.readFromFile(), og den returnerede tekst skrives i Output-vinduet
* Hvis der forsøges læst fra en fil, som ikke er oprettet, skal der skrives en passende meddelelse i Output-vinduet.

# Opgave 4: Find ”*runs*” i arrays 20%

*Definition:* Et *run* er en sekvens af ens værdier i et array. Fx. har arrayet [2, 3, 3, 3, 2, 4, 4, 2, 1, 1] tre *runs*, nemlig (3, 3, 3), (4, 4) og (1, 1).

I denne opgave skal sådanne *runs* findes og udskrives.

a) 5%

Implementer en klasse Die, som kan simulere kast med en ”terning” med et vilkårligt antal sider (dog mindst 2, svarende til kast med en mønt). Klassen skal implementere følgende:

* public Die(int max) // Constructor som tager antal sider på terningen som parameter
* public int throwDie() // returnerer et tilfældigt tal mellem 1 og max, begge inclusive.

b) 5%

Implementer en klasse DieAnalyser, som benytter Die-klassen til at erklære og fylde et array af int:

* public DieAnalyser(int max, int length) // Constructor som tager antal sider og størrelsen på arrayet som parametre
* public void fillArray() // Metode som opretter en instance af Die og udfylder arrayet ved gentagne kald til Die.throwDie().

c) 10%

Overskriv toString() metoden i DieAnalyser, så *runs* i den returnerede String er omgivet af parenteser. Eksempel:

[4, 3, 1, 4, 4, 1, 4, 1, 3, 3]udskrevet med runs: [4, 3, 1, (4, 4), 1, 4, 1, (3, 3)]

*Hint: Følgende pseudokode giver den grundlæggende ide. Hvis den følges, skal der tages højde for ulovlige index og tilføjes kommaer, mellemrum og firkantede parenteser til den dannede streng:*

erklær en tom String r

Set en boolean variabel inRun til false

for hvert index i i array:

hvis inRun:

hvis array[i] er forskelligt fra forrige værdi:

append ”)” til r

set inRun til false

ellers:

hvis array[i] er det samme som den følgende værdi:

append ”(” til r

set inRun til true

append array[i] til r

hvis inRun:

append ”)” til r // Sørger for at afslutte, hvis sidste tal er med i en run.