Kursus 02101
Paul Fischer, DTU Compute

Indledende Programmering – Efterår 2021
Hjemmeopgave 2

Dato: 12. 10. 2021, **Slut dato**: 29. 10. 2021, kl.23:59 på DTULearn

Dette er den anden afleveringsopgave i kurset 02101. Husk at disse opgaver er *obligatoriske* og indgår i kursets samlede bedømmelse. Procenttallet for hver opgave indikerer den estimerede tid.

Læs den generele information om afleveringsopgaverne på Learn:
`VejledningAfleveringsopgaver.pdf` og `Samarbejdspolitik.pdf`.
Husk, at programmerne skal køre i *CodeJudge*, hvor der også ligger et par testfiler.
Husk desuden: Ingen mellemrumsymboler før eller efter outputtet.
Der skal laves en rapport kun for den sidste opgave Problem 3.

In general, make your programs robust, for example ensure that they can handle illegal inputs.

Problem 1 [35%]: [Text anlysis] We want to write a program which allows to analyse texts.
Write a class `TextAnalysis` which reads a text file and allows some text mining. Concretely the program has to support

- the number of words.
- the number of different words.
- the number of immediate repetitions. The word "long" in the following examples counts as one and two, respectively, repetitions: *It is a long, long way* and *It is a long, long, long way.*

A *word* is a non-empty string consisting only of letters (a,. . . ,z,A,. . . ,Z), surrounded by blanks, punctuation, hyphenation, line start, or line end. In the analysis, punctuation is ignored. The analysis is non-case-sensitive.
The class has to have a constructor

```
        public TextAnalysis(String sourceFileName, int maxNoOfWords)
```

The argument `sourceFileName` is the name of the source file, if necessary with the path. The parameter `maxNoOfWords` is an upper bound on the number of words in the file which might help in your implementation. In the CodeJudge tests it will be supplied, i.e., it will be given to you.
Class `TextAnalysis` has to implement the following instance methods:

```
public int wordCount() // returns the number of words in the file
public public int getNoOfDifferentWords()
 // retunrs the number different words in the text.
public int getNoOfRepetitions()
// returns the number of repetitions.
```

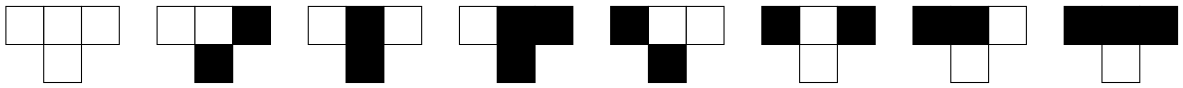
On CodeJudge you find two text files: a simple test text (`testtext01.txt`) for local testing. There also is a longer one, which contains *Hamlet* by William Shakespeare (`Hamlet.txt`) fetched from the Gutenberg project[1].
You should only upload the file `TextAnalysis.java` to CodeJudge, the text files are already there.

Class `TextAnalysis` should not write anything to standard output.
Hint: You might want to consider the following: Given a string `line` which is a line of the text, you can use `line.split("[^a-zA-Z]+");` to break it into the individual words, consider the Java API description for details. Example:

```
String line = "This is, a test 123 Line; or not? b12";
String[] tokens = line.split("[^a-zA-Z]+");
// now the array tokens consits of
// "This" "is" "a" "test" "Line" "or" "not" "b"
```

End of Problem 1

Problem 2 [30%]: [Run a Simulation] The program will simulate the growth of a one-dimensional pattern. We consider a $h \times n$-cell grid, that is, a chess board like layout with h rows and n cells per row, $n \geq 5$. The row and column count starts with 0. Each cell can be empty or filled.
First, seeds are planted in the top row, row 0. This is done by specifying those cells which are filled, all other cells in row 0 are empty.
Then the growing process starts top down. Each row is processed and the cells are filled or left empty depending on the previous row. The content of the j-th cell, for $1 \leq j \leq n-2$, in row r, $r \geq 1$, depends on the three cells in row $r-1$ at positions $j-1$, j, and $j+1$. The cells in columns 0 and $n-1$ are empty.
There are eight possible patterns for these three cells. The patterns are listed below and the value for the cell at (r, j) is also shown. White denotes "empty", black denotes "filled". The top row is $r-1$; cells are numbered left to right.



What you have to do:
Write a class `TrianglePattern.java` with the following:

- The class should have a constructor

  ```
  public TrianglePattern(int n, int h, int[] initial)
  ```

  The parameters n and h specify the grid size as specified above. The array `initial` contains the indices of those columns which are filled in row 0.

[1]`Hamlet.txt` only contains the text of the play. If you are interested, the full text from the Gutenberg repository is available at `www-gutenberg.org/ebooks`

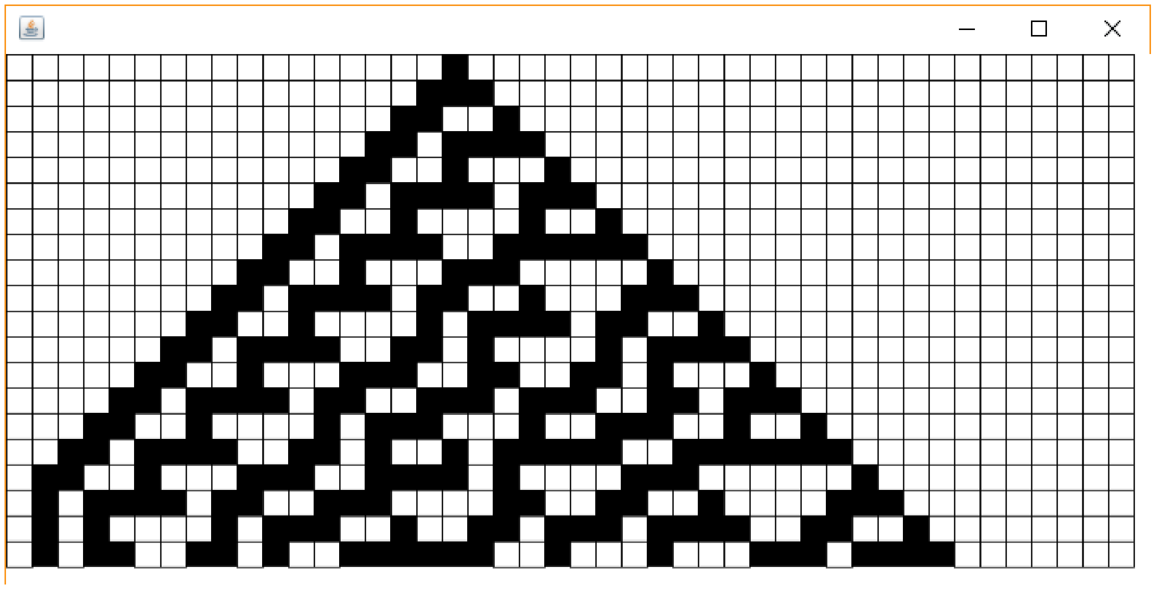


Figure 1: An example output. At left, the figure is limited by reaching column 0. Parameters are $n = 44$, $h = 20$, and $initial = \{17\}$.

- A method
  ```
  public int getValueAt(int r, int c)
  ```
  The method returns 0 if the cell in row r, column c is empty, otherwise it returns 1.
- Getter methods for n and h and $initial$.
  ```
  public int getN()
  public int getH()
  public int[] getInitial()
  ```

Remark: Figure 1 shows examples. On CodeJudge, you find the driver program `TPViewer` which you can use for local testing and which displays the result graphically. Do not try to understand the graphics components used in this program. All test examples on CodeJudge use legal input parameters; you do not have to check this.

End of Problem 2

Problem 3 [35%]: [Design a Class] Design a class `MovingPoint` which represents a point in 2-dimensional space which moves. The movement is described by the direction and speed. The direction is given in degrees in the usual form, i.e., the positive x-axis is $0°$ and angle increases counterclockwise, values are in the range $[0, 360)$. The speed is a non-negative value which is the distance the point moves in one unit of time. There is a maximum speed of 20.0 which applies to all points. The position is specified by the x- and y-coordinates. The class has to contain two constructors:

```
public MovingPoint(); //Constructs a MovingPoint at the origin
                       //with speed 0 and direction 90.
public MovingPoint(double x, double y, double direction, double speed)
//Constructs a MovingPoint at [x,y] with the given direction and speed.
```

The class has to implement instance methods for

- Moving the point for a given number of time units, which might be a real number. In each time unit the point moves in the given direction by a distance equal to the current value of speed. The method has to look like this
  ```
  public void move(double duration)
  ```

- Changing the direction. The method changes the direction by the given angle. A positive value means a turning left a negative one turning right. The method has to look like this
  ```
  public void turnBy(double angle)
  ```

- Changing the speed. The method changes the speed by the given value. A positive value means an increase a negative one a decease (deceleration). If the speed would become negative, it is set to 0.0. If the speed would become larger than the maximum speed, it is set to the maximum speed. The method has to look like this
  ```
  public void accelerateBy(double change)
  ```

- Returning a string which describes the current state by location, direction (d) and speed (s) in the following form. The number of digit should be at maximum precision; CodeJudge will allow for minor rounding errors.
  ```
  [x;y] d s
  [1.02;6.440000000001] 184.3 2.34// example
  ```

  The method has to look like this
  ```
  public String toString()
  ```

End of Problem 3