Number: | | | | | | | Course: _____

Name: _____

INSTITUTO UNIVERSITÁRIO DE LISBOA

Introduction to Programming

1º semester – 2023/2024

1st Exam

Duration: 2 horas (+30 minutes tolerance)

Cotação: 13 val. (Part I — 4 val.; Part II — 9 val.)

Observations and rules:

- The exam has two separate parts, in different pages;
- Pencil can be used to answer the exam;
- Questions should be posed in the first 30 minutes of the exam;
- It is not allowed to exit the room during the exam;
- One can only exit de room after the first hour of the exam;
- It is strictly prohibited to use electrical devices during the exam (e.g. mobile phones). violation of this rule will result in annulment of the exam;
- Functions and procedures, or classes developed in previous questions can and should be used, it is not necessary to use the *import* command.

Please consider that only the following classes are available:

Color

`new Color(r, g, b)` — create a color in RGB format
`getR()` — get value of R
`getG()` — get value of G
`getB()` — get value of B
`getLuminance()` — get value of luminance (brightness), returns a value (int) of equivalent gray intensity to the correspondent color that invoke de function

ColorImage

`new ColorImage(w, h)` — create an image width *w* and height *h*
`getColor(x, y)` — get pixel color (Color) at coordinates *(x, y)*
`setColor(x, y, color)` — changes color of pixel at coordinates *(x, y)* to *color*
`drawText(x, y, text, size, color)` — draws String *text*, left-justified from position *(x, y),* on the image, with letter size *size* and color *color*
`drawCenteredText(x, y, text, size, color)` — draw String *text*, centered at position *(x, y)*
`getWidth()` — get width of the image (number of pixels)
`getHeight()` — get height of the image (number of pixels)

Exceptions

`IllegalArgumentException()` — exceptions related to arguments
`IllegalStateException()` — exceptions related to stat of object
`NullPointerException()` — exceptions related to null value assignments
`FileNotFoundException()` — exceptions related to files

Math

`Math.abs(a)` — returns absolut value of *a*
`Math.max(a, b)` — returns the maximum value between *a* and *b*
`Math.min(a, b)` — returns the minimum value between *a* and *b*
`Math.random()` — returns a random decimal value between [0,0 ; 1,0 [
`Math.sqrt(n)` — returns the square root of *n*
`Math.pow(base, exponent)` — returns $base^{exponent}$

String

`String.valueOf (..)` — returns given argument in String format
`toCharArray()` — returns a vector of characters contained in the String
`length()` — returns the number of characters of the String
`charAt (index)` — returns the character at position *index* in the String

PrintWriter

`new PrintWriter( new File("..") )` — creates a writer
`writer.print(..)` — writes in text mode using writer
`writer.println(..)` — writes in text mode using writer, and terminates line
`writer.close()` — closes writer

Scanner

`new Scanner(...)` or `new Scanner( new File("..") )` — creates scanner
`scanner.hasNextLine()` — returns *true* if a line can be read
`scanner.hasNext()` — returns `true` if a word can be read
`scanner.hasNextInt()` — returns `true` if an integer word can be read
`scanner.nextLine()` — returns a `String` with the read line
`scanner.next()` — returns a `String` with the read word
`scanner.nextInt()` — returns a read integer
`scanner.nextDouble()` — returns a read double
`scanner.close()` — closes scanner

`System.out.print()` ou `System.out.println()`— writes text on console

PART I

Develop a static class `GaloAux` with procedures and/or function to create, manipulate, and visualize the game board of a "Jogo do Galo" (a traditional Portuguese game). The crosses '**X**' should be drawn in green, and the circles '**O**' in red, and the grid should be gray. The grid can be drawn from the top left or centered on image. Empty spaces should be coded as ' '.

Question 1 (1,5 val.)

1.1 Develop an auxiliar static procedure that can be used to draw the gray grid, on a given image, with unit size grid and a given grid size.

```
static void drawGrid (ColorImage img, int sizeGrid){
```

1.2 Develop um procedure to draw the gray grid and the letters '**X**' and '**O**' in the game bord, given the previously filled game matrix (3x3) and the grid size. Empty spaces should be invisible.

```
static void drawBoard (ColorImage img, int sizeGrid, char [][] matrizJogo){
```

Question 2 (0,75 val.)

Develop a procedure that draws a square of size *size* centered on a given position `(x0, y0)`, on a given image, and draws underneath it, centered, a given word, both with given colors.

```
static void drawSquareAndText (ColorImage img, int x0, int y0, Color
colorSquare, Color colorText, int size, String palavra){
```

Question 3 (1,5 val.)

3.1 Develop a static function that returns the first word of a given `String`.

```
e.g. getFirstWord("João Gonçalves Zarco") → "João"

static String getFirstWord (String palavra){
```

3.2 Develop a static function that returns the last word of a given `String`.

```
e.g. getLastWord("João Gonçalves Zarco") → "Zarco"

static String getLastWord (String palavra){
```

Question 4 (0,75 val.)

Develop a function that checks if a given vector is composed only of a single character repeated in all positions, if so, it should return that character, otherwise it should return '_'.

```
static char allEqual (char[] v){
```

Question 5 (0,5 val.)

Develop a function that returns a column of a given matrix in the form of a vector, given the matrix column index.

```
static char[] getColumn (char[][] m, int columnIndex){
```

PART II

Question 1 (4 val.)

Develop/complete a class of objects JogoDoGalo. Assume that "**X**" plays first.

```
class JogoDoGalo {

      private char [][] board;
      private int playNumber;
      private ColorImage img;
```

a) (0,75) Create a new game only with the grid and with all positions initially empty, two constructors: (1) given an image and a grid size; (2) given an initial image, and assuming the grid divides the image in equal parts;

b) (0.75) Develop a function to make a play, in a given de the position `(i,j) in the game` matrix, exceptions should be trown apropriatelly if the position is invalid or already occupied; odd plays should be '**X**' and vice-versa.

c) (0.5) Develop a function that verifies if any row is fully filled only with '**X**'s or '**O**'s, depending on winner return '**X**' or '**O**', if that's not the case it should return '_ ';

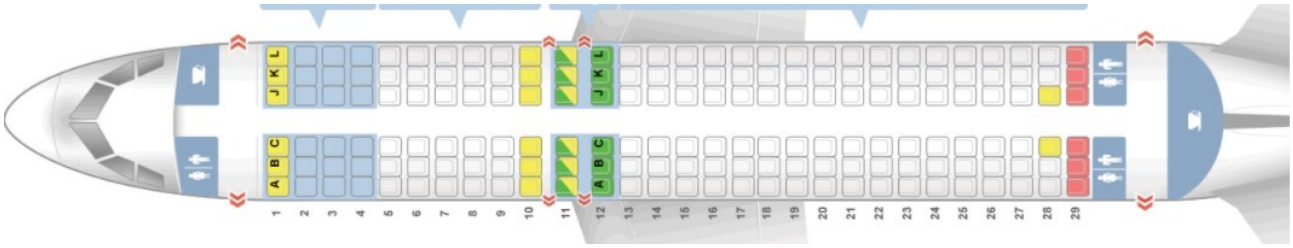
d) (0,5) Develop a function that verifies if any column is fully filled only with '**X**'s or '**O**'s, depending on winner return '**X**' or '**O**', if that's not the case it should return '_ ';

e) (0.5) Develop a function that verifies if any of the two main diagonals is fully filled only with '**X**'s or '**O**'s, depending on winner return '**X**' or '**O**', if that's not the case it should return '_ ';

f) (1,0) Develop a function that verifies if the game is ended (a winner found or board is fully filled), in case of victory it should return '**X**' or '**O**', depending on winner, in case of "draw" - if there is no winner and board if fully filled – it should return '**E**', otherwise, while the game is proceeding it should return '_'; when the game ends there should be a message on console with the winner, or a draw message.

Question 2 (4 val.)

Develop/complete a class of objects ReservasAviao that manages and saves the reservations in file “reservations.dat”.



The objective is to manage the reservation of seats in an airplane, these should be saved in a CSV (comma separated text) text file, (one line per reservation), the reservation line should only contain first and last name of passenger, as shown:

```
“João Zarco,12,B”
“Maria Silva,22,J”
..
```

Please also develop/complete the following classes: `IdPassenger` and `Seat`

```
class Seat {
      private IdPassenger passenger;
      private boolean state;              // true <-> reserved
}

class IdPassenger {
      private String fullName;
      private String reservationCode;
      private String passaportNumber;
..
}
```

Develop:

a) the constructors of classes `IdPassenger` and `Seat`;
b) the attributes and the constructor of class `ReservasAviao` given the number of rows and seats per row on airplane;
c) a method that returns a `String`, consisting of only the first and last according to attribute `fullName`;
d) a method to return the number of occupied seats on a given row index;
e) a method that for a given array of objects of type `IdPassenger`, makes an automated reservation of seats, in a sequential sequence, on the available empty seats;
f) a method to get the total number of empty seats on the airplane;
g) a method to write a file with all the reservation data;