Πανεπιστήμιο Κρήτης, Τμήμα Επιστήμης Υπολογιστών
HY252 – Αντικειμενοστρεφής Προγραμματισμός
Εξάμηνο: **Χειμερινό 2019-2020**
Διδάσκων: **Γιάννης Τζίτζικας**
Βοηθοί: Μ.Ε. Παπαδάκη (1η και 2η), Α. Σαββόπουλος (3η και 5η), Μιχάλης Παναγιώτου (4η συν βοήθεια στα QA των 1 και 2)

1η Σειρά Ασκήσεων

Ανάθεση: 2 Οκτωβρίου 2019
Παράδοση: 22 Οκτωβρίου 2019

Εκπαιδευτικοί Στόχοι

Η 1η άσκηση (40 μονάδες) θα σας επιτρέψει να έχετε μια πρώτη επαφή με τη Java, τη ροή ελέγχου, την είσοδο/έξοδο από κονσόλα και από απλά παράθυρα διαλόγου.
Η 2η άσκηση (20 μονάδες) θα σας επιτρέψει να εξοικειωθείτε με τους πίνακες στην Java (ένας πίνακας είναι αντικείμενο) και με τις στατικές μεθόδους κλάσεων.
Η 3η άσκηση (20 μονάδες) θα σας επιτρέψει να εξοικειωθείτε με τη διαχείριση αρχείων και με τη χρήση κώδικα που σας δίνεται
Η 4η άσκηση (20 μονάδες) θα σας επιτρέψει να εξοικειωθείτε με τη χρήση και εμπλουτισμό εξωτερικού κώδικα, την υλοποίηση αλγορίθμων και τη χρονομέτρηση της εκτέλεσής τους.

Σημειώσεις

1. **Απορίες** σχετικά με την Α1 θα απαντώνται μόνο μέσω του forum της Α1 στο moodle.
2. Την πρώτη άσκηση πρέπει να μπορείτε να τις κάνετε compile και να τις τρέξετε από **γραμμή εντολών**, χωρίς δηλαδή τη χρήση κάποιου **IDE** (έτσι θα γίνει και η εξέταση). Τις επόμενες τρεις (2η, 3η και 4η) συστήνεται να τις κάνετε χρησιμοποιώντας ένα IDE (Eclipse ή NetBeans ή IntelliJ). Σχετικό φροντιστήριο έχει προγραμματιστεί.
3. **Ημερομηνία παράδοσης** και **bonus**: Όποιος παραδώσει **δύο μέρες πριν την προθεσμία** παράδοσης θα έχει βαθμολογικό bonus 10%.
4. **Οδηγίες Παράδοσης**. **ΠΡΟΣΟΧΗ: Ακολουθήστε επακριβώς τις οδηγίες παράδοσης που αναγράφονται στο έγγραφο «Οδηγίες Παράδοσης Ασκήσεων». Διαφορετικά η διόρθωση είναι πολύ χρονοβόρα για τους μεταπτυχιακούς οι οποίοι έχουν και αυτοί μαθήματα και εργασία να κάνουν. Αν παραδώστε κάτι που δεν είναι σύμφωνο με τους κανόνες παράδοσης τότε ίσως υπάρξει μείωση βαθμού και ίσως δεν βαθμολογηθεί.**
5. Η **εξέταση** της Α1 θα γίνει μέσω μίας υποχρεωτικής εργαστηριακής άσκησης (**Α1Ε**). Θα είναι μια απλή και μικρή άσκηση που θα πρέπει να κάνετε στο εργαστήριο/αναγνωστήριο υπό την εποπτεία των βοηθών του μαθήματος (σε περίπτωση απουσίας ή αποτυχίας δεν θα προσμετρηθούν οι βαθμοί της Α1).

Πριν αρχίσετε, διαβάστε τις οδηγίες παράδοσης (θα σας διευκολύνουν)

Άσκηση 1 **[40 μονάδες]** *Ζωγραφίζοντας το γράμμα Η*

Εκπαιδευτικοί Στόχοι: Πρώτη επαφή με Java, ροή ελέγχου, είσοδος/έξοδος από κονσόλα, διαχείριση συμβολοσειρών, παράθυρα διαλόγου, ζωγραφική.

Στόχος είναι να γράψετε μια κλάση DrawH (σε αρχείο DrawH.java) η οποία θα μπορεί να ζωγραφίσει το γράμμα Η με διάφορους τρόπους και σε διάφορα μεγέθη.

(i) [5 μονάδες] – Διαχείριση Εισόδου

Η μέθοδος main πρέπει να μπορεί να διαβάσει δύο παραμέτρους από την γραμμή εντολών (command line arguments), ας τις ονομάσουμε M και L.
Οι αποδεκτές τιμές για τη μεταβλητή M είναι οι 'c' (από το console), 'w' (από το window), 'f' (από το file), 'g' (από το graphics). Οι αποδεκτές τιμές για τη μεταβλητή L είναι οποιοσδήποτε θετικός ακέραιος από το 3 έως το 20.

Το πρόγραμμα σας πρέπει να εκτυπώνει ένα μήνυμα λάθους και να τερματίζει η λειτουργία του αν δεν έχουν δοθεί δύο παράμετροι ή αν οι τιμές αυτών των παραμέτρων δεν είναι οι αποδεκτές.
Δείτε τις σχετικές υποδείξεις.

(ii) [5 μονάδες] - Ζωγραφική σε **Κονσόλα**

Αν M=c τότε το πρόγραμμα σας πρέπει να τυπώνει το γράμμα Η στην κονσόλα με αστεράκια και το ύψος του να είναι L. Για παράδειγμα με java DrawH c 5 πρέπει να εκτυπώνεται το:

```
*   *
*   *
****
*   *
*   *
```

Το βασικό είναι να χρησιμοποιούνται πάντα L γραμμές και το αποτέλεσμα να μοιάζει με το γράμμα Η (να μην μπορεί δηλαδή κάποιος να το μπερδέψει με κάποιο άλλο γράμμα).
Μπορεί η main να καλεί μια στατική μέθοδο της κλάσης σας για να το κάνει.

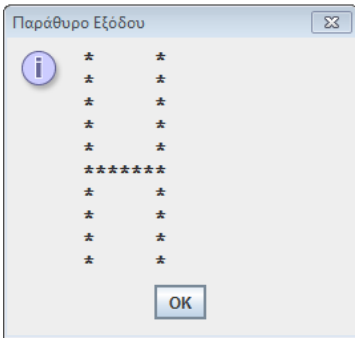
(iii) [5 μονάδες] - Ζωγραφική σε **Παράθυρο Διαλόγου**

Αν M=w τότε το πρόγραμμα σας πρέπει να τυπώνει το γράμμα Η σε ένα παράθυρο εξόδου. Για να εκτυπώσετε μια συμβολοσειρά σε παράθυρο εξόδου μπορείτε να χρησιμοποιήσετε την εντολή

```
JOptionPane.showMessageDialog(null,
    "Ακολουθουν 2 αστεράκια τυπωμένα σε δύο γραμμές \n*\n* ",
    "Παράθυρο Εξόδου",
     JOptionPane.INFORMATION_MESSAGE);
```

Έχοντας κάνει
import javax.swing.JOptionPane;

Για παράδειγμα με java DrawH w 10 το γράμμα H πρέπει να εμφανίζεται σε ένα παράθυρο διαλόγου όπως φαίνεται στο παρακάτω παράδειγμα.

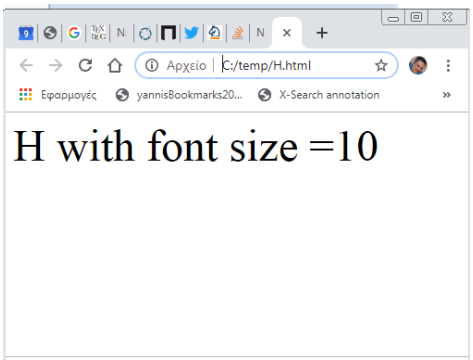


Αν η στοίχιση δεν είναι σωστή, λόγω της γραμματοσειράς, προσθέστε στον κώδικα σας την εντολή
UIManager.put("OptionPane.messageFont", new Font("Lucida Console", Font.BOLD, 14));
έχοντας κάνει
import java.awt.Font;
import javax.swing.JOptionPane;
import javax.swing.UIManager;

(iv) [5 μονάδες] - Εγγραφή σε **Αρχείο**
Αν M=f τότε το πρόγραμμα σας πρέπει να δημιουργεί ένα αρχείο και μέσα του να γράψει HTML κώδικα. Συγκεκριμένα με java DrawH f 10 να δημιουργεί ένα html αρχείο με όνομα H.html όπως το ακόλουθο. Στο παράδειγμα που ακολουθεί η τιμή του L που έδωσε ο χρήστης ήταν το 10.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="content-type" content="text/html;charset=utf-8"/>
</head>
<body><font size="10">H with font size =10</font></body>
</html>
```

Το αρχείο αυτό θα μπορείτε να το ανοίξετε με ένα ιστοπλοηγητή, και θα εμφανιστεί κάτι της μορφής:



Για τη δημιουργία αρχείων και για γράψιμο σε αρχεία δείτε το υλικό από το σχετικό φροντιστήριο που θα γίνει. Ένα απλό παράδειγμα ακολουθεί:

```
PrintWriter writer;
try {
      writer = new PrintWriter("C:\\temp\\AAA.txt", "UTF-8");
      writer.println("The first line");
      writer.println("The second line");
      writer.close();
} catch (Exception e) {
      System.out.println("Πρόβλημα: "+e);
}
```

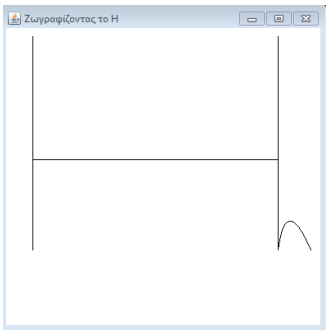
(v) [5 μονάδες] - Ζωγραφική σε Παράθυρο Γραφικών
Αν M=g τότε το πρόγραμμα σας πρέπει να τυπώνει το γράμμα Η σε ένα παράθυρο τύπου γραφικών με κάπως .. καλλιγραφικό τρόπο. Συγκεκριμένα προσθέστε στην κλάση σας την εξής μέθοδο:

```
static void drawHgraphics(int L) {
            Frame f = new Frame("Ζωγραφίζοντας το Η") {
                  public void paint (Graphics g) {
                        Graphics2D g2 = (Graphics2D) g;
                         g2.draw(new Line2D.Double(50,  300, 200, 50)); // a  line
                        // συμπληρώστε

                  }
            };
            f.setSize(400,400);
            f.setVisible(true);
      }
```

Έχοντας προσθέσει τα εξής:

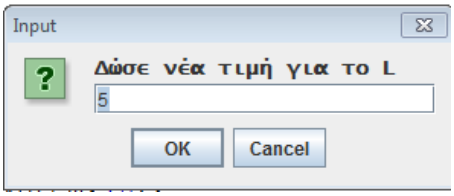
```
import java.awt.Frame;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.geom.Line2D;
import java.awt.geom.QuadCurve2D;
```

Εσείς πρέπει να την καλείτε από τη main και να συμπληρώσετε την paint. Δείτε τι επιλογές δίνονται στο https://docs.oracle.com/javase/tutorial/2d/geometry/primitives.html και συμπληρώστε τη μέθοδο paint ώστε να ζωγραφιστεί ένα κάπως.. καλλιγραφικό «Η» της αρεσκείας σας, όπως για παράδειγμα το ακόλουθο:



(vi) [5 μονάδες] - Ροή Ελέγχου

Εμπλουτίστε το πρόγραμμά σας ώστε το τέλος να ζητάει μια νέα είσοδο για το L. Αυτό να το κάνει από την κονσόλα, εκτός αν η M=w όπου τότε πρέπει να ζητάει την επιλογή μέσω ενός παράθυρου διαλόγου, όπως το ακόλουθο:



- Για να διαβάζετε την είσοδο stdin :

```
Scanner in = new Scanner(System.in);
int aNumber = in.nextInt();
// String nextLine = in.next(); // Αν θέλατε να διαβάσετε String
```

Στην αρχή του αρχείου θα πρέπει να έχετε κάνει:

```
import java.util.Scanner;
```

- Για να πάρετε είσοδο από παράθυρο διαλόγου χρειάζεται να έχετε κάνει

```
import javax.swing.JOptionPane;
```

και να χρησιμοποιήσετε μια εντολή της μορφής:

```
int      inputk = Integer.parseInt(JOptionPane.showInputDialog(
                  "Give me a number ",4)); // το 4 είναι η default τιμή
```

Αφού διαβάσει την νέα τιμή L θα ζωγραφίζει το Η και αυτό θα γίνεται έως ότου ο χρήστης δώσει ένα L που δεν είναι στο επιτρεπτό εύρος τιμών.

(vii) [10 μονάδες] - Αναδρομική Κλήση της main

Η main μιας κλάσης είναι μια μέθοδος που μπορεί να κληθεί όπως όλες οι μέθοδοι. Εμπλουτίστε το πρόγραμμα σας ως εξής: Λίγο πριν ζητήσει νέα τιμή του L (σκέλος vii) να καλείται η main αλλά με διαφορετικές παραμέτρους: συγκεκριμένα με ένα L μειωμένο κατά ένα. Με αυτόν τον τρόπο αν η αρχική

κλήση ήταν με L=10, τότε το πρόγραμμα θα ζωγραφίσει το Η με μέγεθος 10, κατόπιν με 9, κ.ο.κ έως 3 (το ελάχιστο αποδεκτό) και μετά θα ζητήσει νέα τιμή από το χρήστη (στην ουσία λόγω της αναδρομής θα δείτε ότι θα ζητήσει νέα τιμή από το χρήστη τόσες φορές όσες οι κλήσεις της main που έγιναν).

ΣΗΜΕΙΩΣΗ.
Μάθετε να εκτελείτε το πρόγραμμα σας (και να περνάτε τις παραμέτρους που χρειάζονται) με διάφορους τρόπους: (α) από το IDE που χρησιμοποιείτε, (β) από command line, (γ) φτιάχνοντας ένα jar αρχείο.

Υποδείξεις

- Μετατροπή μιας συμβολοσειράς str σε ακέραιο: `int x = Integer.parseInt(str);`
- Για να διαβάζετε την είσοδο stdin :

```
Scanner in = new Scanner(System.in);
int aNumber = in.nextInt();
// String nextLine = in.next(); // Αν θέλατε να διαβάσετε String
```

Στην αρχή του αρχείου θα πρέπει να έχετε κάνει:

```
import java.util.Scanner;
```

- Για να πάρετε είσοδο από παράθυρο διαλόγου χρειάζεται να έχετε κάνει

```
import javax.swing.JOptionPane;
```

και να χρησιμοποιήσετε μια εντολή της μορφής:

```
int      inputk = Integer.parseInt(JOptionPane.showInputDialog(
                  "Give me a number ",4)); // το 4 είναι η default τιμή
```

- Για να εκτυπώσετε μια συμβολοσειρά σε παράθυρο εξόδου

```
JOptionPane.showMessageDialog(null,
    "Ακολουθουν 2 αστεράκια τυπωμένα σε δύο γραμμές \n*\n* ",
    "Παράθυρο Εξόδου",
     JOptionPane.INFORMATION_MESSAGE);
```

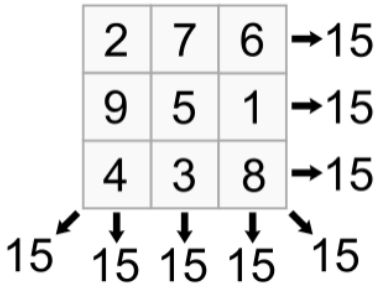
Για παράθυρα άλλου τύπου (προειδοποίησης, λάθους) δείτε τι άλλες τιμές μπορούν να περάσουν στην τελευταία παράμετρο (αντί του `JOptionPane.INFORMATION_MESSAGE`)

- Για τη δημιουργία αρχείων και για γράψιμο σε αρχεία δείτε το υλικό από το σχετικό φροντιστήριο που θα γίνει. Ένα απλό παράδειγμα ακολουθεί:

```
PrintWriter writer;
try {
     writer = new PrintWriter("C:\\temp\\AAA.txt", "UTF-8");
     writer.println("The first line");
     writer.println("The second line");
     writer.close();
} catch (Exception e) {
     System.out.println("Πρόβλημα: "+e);
}
```

Άσκηση 2 – [20 μονάδες] *Πίνακες*

Ένα μαγικό τετράγωνο τάξης n είναι μία τετράγωνη συστοιχία n*n **διακριτών** ακέραιων αριθμών στην οποία το άθροισμα των n αριθμών σε κάθε **στήλη**, **σειρά** και **διαγώνιο** είναι το ίδιο. Η «μαγεία» βρίσκεται στο γεγονός πως οι αριθμοί σε κάθε στήλη, σειρά και διαγώνιο αθροίζονται στον ίδιο αριθμό, ο οποίος ονομάζεται μαγικό στοιχείο. Παρακάτω παρουσιάζεται ένα 3ης τάξης μαγικό τετράγωνο με μαγικό στοιχείο τον αριθμό 15.



Γράψτε μια κλάση **MagicSquareChecker**. Η main να ζητάει από το χρήστη το **n** (ο οποίος **πρέπει** να είναι **ακέραιος** από **2** έως **10**) και κατόπιν να δίνει επαναληπτικά τα **n*n** στοιχεία του πίνακα τα οποία πρέπει να τα αποθηκεύεται σε ένα δισδιάστατο πίνακα ακεραίων. Κατόπιν να ελέγχει εάν πρόκειται για «μαγικό τετράγωνο» και αν ναι να εκτυπώνει το αντίστοιχο μήνυμα και το μαγικό στοιχείο. Συγκεκριμένα η κλάση που θα φτιάξετε πρέπει να προσφέρει την εξής **στατική** μέθοδο

```
public static boolean checkIsMagic(int[][] s)
```

η οποία με τη σειρά της να αξιοποιεί τις ακόλουθες **στατικές** μεθόδους που πρέπει να φτιάξετε:

```
private static int  sumOfRow    (int[][] s, int k)
private static int  sumOfColumn (int[][] s, int k)
private static int  sumOfDiagonal1   (int[][] s)
private static int  sumOfDiagonal2   (int[][] s)
```

Επιπλέον βοηθητικές **στατικές** μεθόδοι:

```
public static int  getMagicNumber(int[][] s)
private static boolean  hasDuplicates(int[][] s)
```

Σημείωση: Το **k** που εμφανίζεται σαν όρισμα στις 2 συναρτήσεις (sumOfRow και sumOfColumn) αναπαριστά τον δείκτη της γραμμής και της στήλης αντίστοιχα. Για να είναι ένα τετράγωνο μαγικό **δεν** πρέπει να εμφανίζεται ένας ακέραιος πάνω από μια φορά. Για τέτοιου είδους ελέγχους βολεύει η έννοια του **συνόλου** (set) και υλοποιήσεις υπάρχουν στις βιβλιοθήκες της java τις οποίες θα δούμε (σε επόμενα μαθήματα). Εσείς έχετε τις εξής επιλογές για έλεγχο διπλοτύπων: Για κάθε νέο αριθμό που δέχεστε ως είσοδο μπορείτε είτε (α) να κάνετε σειριακή αναζήτηση στον πίνακα για να βεβαιωθείτε ότι ο αριθμός δεν έχει ήδη δοθεί, ή (β) να τον βάζετε στην κλάση **MySet** που σας δίνεται (η οποία είναι μια απλή υλοποίηση συνόλου με πίνακα) η οποία προσφέρει έλεγχο διπλοτύπων.

Επίσης μετά την είσοδο αριθμών πρέπει να τυπώνονται οι αριθμοί στη κονσόλα στοιχισμένοι έκαστος σε χώρο 5 χαρακτήρων (δείτε τις επιλογές που σας προσφέρει η printf) και αν το τετράγωνο δεν είναι

μαγικό να τυπώνεται τουλάχιστον ένας λόγος για τον οποίο δεν είναι, όπως φαίνεται στα δύο παρακάτω παραδείγματα:

Παράδειγμα 1:
(μετά την είσοδο των αριθμών από το χρήστη το πρόγραμμα σας πρέπει να τυπώνει):

```
------------------
|    2|    7|    6|
------------------
|    9|    5|    1|
------------------
|    4|    3|    8|
------------------
The square is magic and the magic element is: 15
```

Παράδειγμα 2:
(μετά την είσοδο των αριθμών από το χρήστη το πρόγραμμα σας πρέπει να τυπώνει):

```
------------------------
|   33|  221|  222|    3|
------------------------
|    3|    3|  333|    3|
------------------------
|  333|    2|   22|  222|
------------------------
|   33|   22|   11|   12|
------------------------
The numbers you gave contain duplicates
The square isn't magic!
```

Άσκηση 3 – [20 μονάδες] *Επιλογή, Διάβασμα, Γράψιμο Αρχείων*

Σκοπός αυτής της άσκησης είναι να γράψετε ένα πρόγραμμα το οποίο να μπορεί να διαβάσει ένα τετράγωνο (πίνακα ακεραίων) από ένα αρχείο, όπου κάθε αριθμός διαχωρίζεται από τον επόμενο με ένα **κόμμα**, και να ελέγχει αν είναι μαγικό τετράγωνο, χρησιμοποιώντας και αξιοποιώντας αυτά που έχετε κάνει στην Άσκηση 2. Επίσης θα πρέπει να εμφανίζει τα κατάλληλα μηνύματα και να μπορεί επίσης να γράψει ένα αρχείο. Συγκεκριμένα:

(α) Το πρόγραμμα πρέπει να ζητάει από το χρήστη να διαλέξει ένα αρχείο (.txt) μέσω παραθύρων διαλόγου. Για να επιλέξετε ένα αρχείο από παράθυρο διαλόγου χρειάζεται να έχετε κάνει:

```
import javax.swing.JFileChooser;
```

και να χρησιμοποιήσετε τις παρακάτω εντολές:

```
JFileChooser fileChooser = new JFileChooser();
fileChooser.setDialogTitle("Select a file");
int userSelection = fileChooser.showSaveDialog(null);
if (userSelection == JFileChooser.APPROVE_OPTION) {
   File file = fileChooser.getSelectedFile();
   String filepath = file.getAbsolutePath();
   System.out.println("The path of the selected file is: " + filepath);
}
```

Εναλλακτικά, μπορείτε να χρησιμοποιήσετε τη σχετική κλάση (`GraphicalFileLoaderSaver.java`) που έχει ανέβει στο moodle (συγκεκριμένα την μέθοδο GraphicalFileLoaderSaver.saveFile()).

(β) Το πρόγραμμα σας πρέπει κατόπιν να ανοίγει το αρχείο, να διαβάζει τους ακεραίους (που διαχωρίζονται μεταξύ τους με κόμμα) και να τους βάζει σε έναν πίνακα (όπως στην Άσκηση 2). Αν ένα αρχείο έχει προβληματική δομή (π.χ. η πρώτη γραμμή του αρχείου έχει 4 ακεραίους ενώ η δεύτερη γραμμή 3) τότε να βγάζει μήνυμα ότι η μορφοποίηση δεν είναι η κατάλληλη. Αν τα περιεχόμενα του αρχείου διαβάστηκαν επιτυχώς και γέμισε ο πίνακας τότε, να εκτυπώνεται στην κονσόλα ο πίνακας (όπως στην Άσκηση 2) και εν συνεχεία να γίνεται ο έλεγχος μαγικού τετραγώνου. Εάν είναι, τότε να εμφανίζεται αντίστοιχο μήνυμα και ο μαγικός αριθμός. Αν δεν είναι μαγικό τετράγωνο να εμφανίζεται αντίστοιχο μήνυμα.

(γ) Αν το τετράγωνο ήταν μαγικό να δίνεται η δυνατότητα στο χρήστη να επιλέξει ένα φάκελο (μέσω παραθύρων διαλόγου) όπου να γραφτεί ένα αρχείο το οποίο θα έχει τα περιεχόμενα του πρώτου αρχείου κατόπιν μια κενή γραμμή και μετά μια γραμμή της μορφής «Μαγικό τετράγωνο με μαγικό αριθμό Χ» όπου Χ ο μαγικός αριθμός. **Διευκρίνιση:** Ο χρήστης θα πρέπει να διαλέξει μόνο τον φάκελο που θα γραφτεί το αρχείο. Το αρχείο θα πρέπει να δημιουργηθεί μέσω κώδικα και το όνομα που θα του ανατεθεί δεν έχει σημασία π.χ. MagicSquareSavedFile.txt

(δ) Επεκτείνεται το πρόγραμμα σας ώστε αντί για αρχείο εισόδου ο χρήστης να έχει τη δυνατότητα να δώσει ένα URL (Universal Resource Locator) και το πρόγραμμα σας να κατεβάζει το αντίστοιχο αρχείο από τον Παγκόσμιο Ιστό. Ενδεικτικά αρχεία για τον έλεγχο των μεθόδων σας μπορείτε να βρείτε στο παρακάτω link: https://www.csd.uoc.gr/~savvopoulos/hy252_A13_testFiles/testX Όπου Χ = 1 , 2 ή 3)

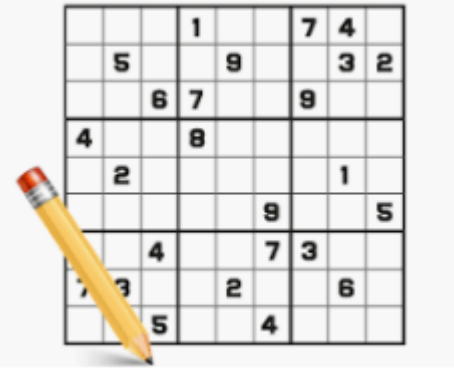
Σας δίνεται ο παρακάτω κώδικας που διαβάζει και εκτυπώνει το περιεχόμενο ενός URL, δηλαδή μιας σελίδας στο διαδίκτυο:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.URL;
import javax.swing.JOptionPane;

public class Downloader {

  static int download(String address) {
     try {
           URL url = new URL(address);
           BufferedReader in = new BufferedReader(
                     new InputStreamReader(url.openStream(), "UTF-8"));
           int c = in.read();
           while (c != -1) {
               System.out.print((char) c);
               c = in.read();
           }
           in.close();
     } catch (Exception e) {
           System.out.println(e);
           return -1;
     }
     return 0;
  }

     public static void main(String[] a) {

           String toDownload =
                JOptionPane.showInputDialog("Δώστε την διεύθυνση ","");
           System.out.println(toDownload);

           download(toDownload);
     }
}
```

Προφανώς για να δουλέψει πρέπει να είστε συνδεδεμένοι στο διαδίκτυο.

Άσκηση 4 – [20 μονάδες] Sudoku

Κατανόηση και εμπλουτισμός κώδικα, χρονομέτρηση.



Η άσκηση αυτή σχετίζεται με το γνωστό παιχνίδι Sudoku, το διασκεδαστικό Γιαπωνέζικο παιχνίδι παζλ που χρησιμοποιεί αριθμούς, αλλά δεν χρειάζεται μαθηματικά.

Αρχικά διαβάστε το άρθρο
https://www.baeldung.com/java-sudoku.
Εν συνεχεία κατεβάστε και τρέξτε την έκδοση με **backtracking** από το αρχείο
`https://github.com/eugenp/tutorials/blob/master/algorithms-miscellaneous-2/src/main/java/com/baeldung/algorithms/sudoku/BacktrackingAlgorithm.java`
Η οποία μπορεί να λύσει ένα παζλ soduku.
ΠΡΟΣΟΧΗ: Για να κάνετε αυτήν την άσκηση **δεν χρειάζεται να καταλάβετε ακριβώς τον αλγόριθμο και τον κώδικα**, απλά τι κάνει η κάθε μέθοδος γενικά (δηλαδή ποιος είναι ο ρόλος της).

Εν συνεχεία εμπλουτίστε τον κώδικα ώστε:

(α) Ο αλγόριθμος προϋποθέτει ότι το αρχικό πάζλ (board) που του δίδεται είναι σωστό, και δεν κάνει κάποιο έλεγχο. Αυτό μπορείτε να το καταλάβετε μόνοι σας αν αλλάξετε την πρώτη γραμμή του παζλ που βρίσκεται μέσα στον κώδικα ως εξής:
από
{8, 0, 0, 0, 0, 0, 0, 0, 0}
σε
{12, 0, 0, 0, 0, 0, 0, 0, 0} // θα λάβουμε runtime exception
ή σε
{8, 8, 0, 0, 0, 0, 0, 0, 0} // θα μας επιστρέψει το board άλυτο
Ή σε
{8, 6, 0, 0, 0, 0, 0, 0, 0} // θα μας επιστρέψει το board άλυτο χωρίς να μας ενημερώσει για κάτι

Για το λόγο αυτό προσθέστε μια μέθοδο **boolean isValidBoard(int[][] brd)** που να ελέγχει αν πράγματι το παζλ είναι σωστό (δηλαδή αν δεν παραβιάζεται κάποιος περιορισμός του sudoku):

- Τα κελιά κάθε γραμμής έχουν κενά ή τιμές από το 1 έως το 9 και όχι διπλότυπα
- Τα κελιά κάθε γραμμής στήλης έχουν κενά ή τιμές από το 1 έως το 9 και όχι διπλότυπα
- Κάθε ένα από τα 9 πλέγματα 3χ3 δεν έχει διπλότυπα

(β) Προσθέστε μια μέθοδο **isSolvableBoard(board)** η οποία να επιστρέφει true αν το board είναι ορθό και επιλύεται (αξιοποιώντας τον κώδικα που κατεβάσατε και αυτά που κάνατε στο προηγούμενο σκέλος).

(γ) Προσθέστε μια μέθοδο που να μπορεί να φτιάξει ένα board το οποίο να έχει **X** κενά κελιά (π.χ. **X**=75) και τα υπόλοιπα να τα γεμίζει τυχαία με αριθμούς.

(δ) Γράψτε κώδικα που να μπορεί να δημιουργεί **N έγκυρα** ζευγάρια Sudoku (άσκηση-λύση). Ο κώδικας σας μπορεί πρώτα να παράγει ένα τυχαίο παζλ (μέσω αυτού που κάνατε στο σκέλος γ) και μετά να ελέγχει αν είναι έγκυρο και επιλύσιμο (σκέλος β). Στο τέλος να τυπώνει τα έγκυρα boards που κατάφερε να φτιάξει και επίσης πόσο χρόνος δαπανήθηκε, και πόσα άκυρα boards φτιάχτηκαν. Προσπαθήστε ώστε ο κώδικας σας να μπορεί να παράξει τουλάχιστον 10 έγκυρα ζευγάρια σε λιγότερο από 10 λεπτά. Η εκτέλεση του προγράμματος σας θα τυπώνει κάτι της μορφής (εδώ):

N=100 X=75	N=10 X=65	N=10 X=54
Board #1 0 0 0 0 0 0 2 0 0 0 0 2 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 7 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Solution of the Board #1 1 4 3 5 7 8 2 6 9 5 7 2 6 1 9 3 4 8 6 8 9 2 3 4 1 5 7 2 1 4 3 8 5 7 9 6 3 5 7 9 2 6 4 8 1 8 9 6 7 4 1 5 2 3 4 2 1 8 6 3 9 7 5 9 3 8 4 5 7 6 1 2 7 6 5 1 9 2 8 3 4 … Board #100 0 0 0 0 0 0 6 0 0 0 7 0 0 0 0 0 0 0 0 0 5 0 0 0 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 Solution of the Board #100 1 2 3 4 5 6 7 8 9 7 4 6 2 8 9 1 3 5 5 8 9 3 1 7 2 4 6 2 6 1 5 3 4 8 9 7 3 5 4 7 9 8 6 1 2 8 9 7 1 6 2 3 5 4	Board #1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 4 0 0 0 9 0 5 0 0 0 0 0 0 0 0 0 0 0 8 0 0 0 0 0 7 0 6 0 0 3 0 0 0 8 0 0 0 7 4 0 0 0 0 0 0 7 0 0 0 5 0 0 0 4 0 0 0 0 0 0 0 0 1 2 0 Solution of the Board #1 2 3 4 6 9 7 5 1 8 1 5 6 2 3 8 7 4 9 7 8 9 1 5 4 3 6 2 5 1 7 9 6 3 2 8 4 9 4 8 5 7 2 6 3 1 3 6 2 4 8 1 9 5 7 4 9 1 3 2 5 8 7 6 8 2 5 7 1 6 4 9 3 6 7 3 8 4 9 1 2 5 ... Board #10 5 0 0 0 0 8 0 0 0 2 0 0 0 4 6 0 0 1 0 0 6 0 0 0 0 0 0 0 0 0 0 0 3 0 9 0 7 0 0 0 0 0 0 0 0 0 0 0 9 0 2 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 0 7 0 8 0 0 0 3 0 0 0 0 0 0 Solution of the Board #10 5 1 4 2 3 8 6 7 9 2 3 9 7 4 6 8 5 1 8 7 6 1 5 9 2 3 4 1 4 2 5 6 3 7 9 8 7 9 5 8 1 4 3 2 6 3 6 8 9 7 2 1 4 5	Board #1 0 0 0 0 0 0 8 0 5 0 0 0 0 9 4 0 0 0 0 3 0 0 1 8 0 0 0 0 0 5 0 0 0 0 0 0 8 0 0 0 0 9 7 2 0 9 6 7 8 0 0 0 5 1 0 0 1 0 0 3 0 0 0 2 0 3 0 0 5 0 9 8 0 0 6 7 0 0 0 0 0 Solution of the Board #1 1 4 9 2 3 7 8 6 5 7 5 8 6 9 4 2 1 3 6 3 2 5 1 8 4 7 9 3 2 5 1 7 6 9 8 4 8 1 4 3 5 9 7 2 6 9 6 7 8 4 2 3 5 1 5 8 1 9 2 3 6 4 7 2 7 3 4 6 5 1 9 8 4 9 6 7 8 1 5 3 2 ... Board #10 5 4 2 0 0 0 0 9 0 7 0 0 0 0 0 0 4 0 0 0 8 5 6 0 0 0 0 4 2 0 0 8 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0 7 3 2 0 0 0 0 3 0 9 2 7 0 0 1 5 0 0 5 0 0 3 0 8 7 0 0 0 6 0 0 0 0 0 Solution of the Board #10 5 4 2 1 3 7 8 9 6 7 3 6 8 9 2 5 4 1 1 9 8 5 6 4 3 7 2 4 2 1 7 8 5 9 6 3 6 5 3 4 1 9 7 2 8 9 8 7 3 2 6 1 5 4

4 1 2 8 7 5 9 6 3 6 7 8 9 4 3 5 2 1 9 3 5 6 2 1 4 7 8 Empty cells per board : 75 Valid boards created : 100 Invalid boards created : 42 Unsolvable boards created: 0 Elapsed time in seconds : 0.426	4 2 7 6 8 5 9 1 3 6 5 1 3 9 7 4 8 2 9 8 3 4 2 1 5 6 7 Empty cells per board : 65 Valid boards created : 10 Invalid boards created : 506 Unsolvable boards created: 1 Elapsed time in seconds : 10.71	3 6 9 2 7 8 4 1 5 2 1 5 9 4 3 6 8 7 8 7 4 6 5 1 2 3 9 Empty cells per board : 54 Valid boards created : 10 Invalid boards created : 64614 Unsolvable boards created: 62 Elapsed time in seconds : 21.555

Σχόλιο: Προφανώς αν κάποιος θέλει να παράξει πολλά παζλ, θα μπορούσε επίσης από τη λύση του κάθε παζλ να παράγει πολλά άλλα, σβήνοντας τυχαία αριθμούς από τη λύση του παζλ. Δεν χρειάζεται εσείς να κάνετε κάτι τέτοιο, απλά το σημειώνουμε για τη δική σας ενημέρωση.

Υποδείξεις

Παραγωγή τυχαίου ακεραίου από MIN_VALUE έως MAX_VALUE:

```
Random randomGenerator = new Random();
int randomInt = randomGenerator.nextInt(MAX_VALUE-MIN_VALUE+1) + MIN_VALUE;
```

Χρονομέτρηση

```
// finding the time before the operation is executed
 long start = System.currentTimeMillis();

<κώδικας που θέλουμε να χρονομετρήσουμε>

// finding the time after the operation is executed
 long end = System.currentTimeMillis();

//finding the time difference and converting it into seconds
float sec = (end - start) / 1000F; System.out.println(sec + " seconds");
```

ΠΡΟΑΙΡΕΤΙΚΗ ΑΣΚΗΣΗ

Για δική σας εξάσκηση. Δεν υπάρχει βαθμολογικό bonus.

Άσκηση 5 – *Κρυπτογράφηση Αρχείων*

File, I/O, Χρήση κλάσης της οποίας ο κώδικας δίδεται

(α)

Γράψτε μια κλάση Java με όνομα encrypt (σε αρχείο `encrypt.java`). Η main της θα λαμβάνει ως παραμέτρους από την γραμμή εντολών το όνομα ενός αρχείου εισόδου (π.χ. myMessage.txt) και έναν φυσικό αριθμό, ας τον ονομάσουμε K, και θα δημιουργεί ένα νέο αρχείο του οποίου το όνομα θα περιέχει τη λέξη Encrypted στην αρχή (π.χ `EncryptedmyMessage.txt`). Το περιεχόμενο του νέου αρχείου θα πρέπει να είναι το περιεχόμενο του αρχείου εισόδου κρυπτογραφημένο.
Συγκεκριμένα η κλάση encrypt πρέπει να έχει την εξής στατική μέθοδο:

(i) `static void Transform(String inputFileName, String outputFileName, int inputK)`

η οποία θα καλείται από την main της κλάσης με τα κατάλληλα ορίσματα. Θα διαβάζει/γράφει το αρχείο εισόδου/εξόδου ανά byte. Εκ τούτου πρέπει να χρησιμοποιεί την FileInputStream και FileOutputStream. Η κρυπτογράφηση που θα προσφέρεται θα γίνεται αντικαθιστώντας το κάθε byte με ένα byte στο οποίο έχει προστεθεί η τιμή inputk (που αντιστοιχεί στην παράμετρο K από τη γραμμή εντολών).

Προφανώς η ίδια κλάση επαρκεί και για την αποκρυπτογράφηση ενός κρυπτογραφημένου αρχείου. Για παράδειγμα αν έχουμε ένα αρχείο myMessage.txt η εντολή

```
java encrypt myMessage.txt 2
```

θα δημιουργήσει το αρχείο EncryptedmyMessage.txt του οποίου το περιεχόμενο θα είναι κρυπτογραφημένο.
Για να το αποκρυπτογραφήσουμε αρκεί η εντολή

```
java encrypt EncryptedmyMessage.txt -2
```

το οποίο θα δημιουργήσει το αρχείο EncryptedEncryptedmyMessage.txt του οποίου το περιεχόμενο πρέπει (αν η υλοποίηση είναι σωστή) να είναι αυτό του αρχικού (ήτοι αυτό του myMessage.txt).

Το πρόγραμμά σας στο τέλος πρέπει να τυπώνει στη κονσόλα πόσο χρόνο πήρε η κρυπτογράφηση.
Δοκιμάστε αυτό που κατασκευάσατε και δείτε:

1. Αν δουλεύει σωστά σε οποιοδήποτε αρχείο, π.χ. .txt, .doc, .exe, .zip
2. Εάν υπάρχει πρόβλημα εάν ο αριθμός K είναι πολύ μεγάλος;
3. Δοκιμάστε να κρυπτογραφήσετε και πολύ μεγάλα αρχεία και σχολιάστε το χρόνο που απαιτήθηκε.

(β)

Για να βεβαιωθείτε ότι δουλεύει σωστά αυτό που κάνατε, προσθέστε στην κλάση μια στατική μέθοδο `boolean test()` η οποία θα κάνει διάφορους ελέγχους ορθότητας. Η μέθοδος θα επιστρέφει true αν όλοι οι έλεγχοι εκτελεστούν επιτυχώς. Για παράδειγμα μετά από μια κλήση του χρήστη:

```
java encrypt.java Lala.exe 7
```

πρέπει η main να καλεί την test(), η οποία θα μπορούσε να καλεί την transform με K = 7 και μετά να μετατρέπει το αποτέλεσμα με K = -7, και στο τέλος να ελέγχει ότι το αρχείο που προκύπτει έχει τα ίδια ακριβώς περιεχόμενα με το αρχικό. Στην περίπτωση που η μέθοδος επιστρέψει false, η κονσόλα πρέπει να τυπώνει ένα αντίστοιχο μήνυμα. Είστε ελεύθεροι να κάνετε όποιο άλλο έλεγχο κρίνετε σκόπιμο.

(γ)

Εμπλουτίστε την κλάση που φτιάξατε και με την εξής μέθοδο:

```
static void Transform(File inputFile, File outputFile,  int inputK)
```

Στην ουσία η παραπάνω απλά λαμβάνει διαφορετικού τύπου παραμέτρους. Αναπαραγοντοποιείστε (refactor) ανάλογα την προηγούμενη μέθοδο (του σκέλους (α)) ώστε να μην υπάρχουν στον κώδικα της κλάσης σας επαναλήψεις (κάτι το οποίο είναι κακή προγραμματιστική πρακτική), αλλά η προηγούμενη μέθοδος (του σκέλους α) να καλεί σε κάποιο σημείο τη νέα (του σκέλους γ). Εν συνεχεία χρησιμοποιήστε τις εντολές που θα βρείτε στις υποδείξεις ώστε η επιλογή του αρχείου προς κρυπτογράφηση (ή αποκρυπτογράφηση) να γίνεται με γραφικό (παραθυρικό) τρόπο.

Εμπλουτίστε τη ροή ελέγχου της main έτσι ώστε η επιλογή του γραφικού τρόπου να ενεργοποιείται εάν ο χρήστης καλέσει τη κλάση encrypt χωρίς καμία παράμετρο εισόδου. Αν ο χρήστης έχει δώσει παραμέτρους εισόδου τότε το πρόγραμμα πρέπει να λειτουργεί όπως στο σκέλος (α).

<u>Προαιρετικά</u>

1/ Να λαμβάνει ως είσοδο το όνομα ενός φακέλου και να κρυπτογραφεί όλα τα περιεχόμενα του.

<u>Υποδείξεις</u>

- Για να μετράτε τον χρόνο σε nano seconds χρησιμοποιείστε:

```
long startTime = System.nanoTime();
… (code)
… (code)
long endTime = System.nanoTime();
long timeInNanosecs = endTime – startTime;
```

- Για να επιλέξετε ένα αρχείο από παράθυρο διαλόγου χρειάζεται να έχετε κάνει:

```
import javax.swing.JFileChooser;
```

και να χρησιμοποιήσετε τις παρακάτω εντολές:

```
JFileChooser fileChooser = new JFileChooser();
fileChooser.setDialogTitle("Select a file");
int userSelection = fileChooser.showSaveDialog(null);
if (userSelection == JFileChooser.APPROVE_OPTION) {
   File file = fileChooser.getSelectedFile();      String filepath =
file.getAbsolutePath();
   System.out.println("The path of the selected file is: " + filepath);
}
```

Εναλλακτικά, μπορείτε να χρησιμοποιήσετε τη σχετική κλάση (`GraphicalFileLoaderSaver.java`) που έχει ανέβει στο moodle (συγκεκριμένα την μέθοδο GraphicalFileLoaderSaver.saveFile()).

Καλή εργασία!