Name and, if possible, ID#:______

Problem 2

Colors in Java can be represented by objects of type ***Color***. Each such object contains the ***red, green*** and ***blue*** components of the corresponding color as integer values from **0** to **255**. Consider below a Java code that creates and initializes a rectangular array of ***Color*** type:

```
import java.util.Scanner;
import java.awt.Color;

public class Colors {

public static void main(String args[]) {
      Scanner in = new Scanner(System.in);

// Read number of rows and columns and create a Color array of such size
      Color[][] c = new Color[in.nextInt()][in.nextInt()];

// For each element read the red, green and blue components as integers and
// create a Color object by calling Color(int, int, int) constructor
      for (int row = 0; row < c.length; row++)
            for (int col = 0; col < c[0].length; col++)
int z;                c[row][col] = new Color(in.nextInt(), in.nextInt(), in.nextInt());
// TO BE CONTINUED
}
}
```

Continue with a Java code that creates another array ***Color[][] g*** of the same size and fills it with gray equivalents of the colors from the array ***Color[][] c***. To get a grey equivalent of a given color ***c[i][j]***, it is enough to construct a ***Color*** object, whose red, green and blue components all are equal to the calculated average of red, green and blue components of the initial ***c[i][j]***. Use ***int getRed()***, ***int getGreen()*** and ***int getBlue()*** methods of class ***Color***.

```
Color[][] g = new Color ~~[]~~ [c.length][c[0].length];
 for (int x = 0; x < c.length; x++){
       for (int y = 0; y < c[0].length; y++){
~~int~~ z= ~~g[x][y]=~~ c[x][y].get Red() + c[x][y].get Green() + c[x][y]getBlue()/3;
 g[x][y] = new Color(z, z, z);
 }
}
```

6

OOP.MT1-170215.M111

Use the backside, if needed

Name and, if possible, ID#:________

Problem 3

Similar to files, strings also can be related to streams in C++, this time using ***stringstream*** objects. Particularly, it is enough to create an object of type ***istringstream*** to organize formatted reading from a string. Consider, for example, a C++ code below:

```
#include <string>
#include <sstream>
#include <iostream>
using namespace std;

void main()
{
        string text = "Before_increment: 199999999", word;
        int num;
        istringstream tokens(text);

        tokens >> word >> num;
        cout << "After " << word.substr(7) << num + 1 << endl;
}
// After increment:200000000
```

Write a C++ function ***double value(string expression)*** that takes as its argument a string representing an arithmetic expression, evaluates it and returns its value. The expression includes only '+' and '–' operations and double operands, both positive and negative. The operands and operations are delimited by spaces.

For example, ***value("5.1 – -0.7 + 1.2")*** results in ***7.0***.

4

```
x  double value(string expression) {
    istringstream tokens(string)expression)
    int spaces = 0;
    for(int x = 0; x < stringexpression.length; x++)
      { if (~~charAt~~ stringexpression.charAt(x) == " ")
                  spaces ++;
    string words [spaces];
    for (int x = 0; x < spaces ; x++)
        { ~~words[x] =~~
          tokens >> words[x];
        }
    for (int x = 0; x < spaces; x++)
      {   ~~if (words[x] == "+" || words[x] == "-")~~.
          if (words [x] == "+")
         {words[x] = words[x-1] + words [x + 1];
          words [x - 1] = " ";
          words [x + 1] = " ";
         }
          if (words [x] == "-")
        { words [x] = words [x - 1] - words [x + 1];
          words [x - 1] = " ";
          words [x + 1] = " ";
}       }
```