Section, Name and ID#:

AMERICAN UNIVERSITY OF ARMENIA
College of Science and Engineering
CS 120 Introduction to Object-Oriented Programming
MIDTERM EXAM

2

Date / Time: Friday, March 17 2017 at 17:30
Duration: 2 hours
Attention: **ANY TYPE OF COMMUNICATION IS STRICTLY PROHIBITED**
Write down your section, name and ID# at the top of all used pages

Participation:

Problem 1: Consider below a C++ function ***float kahan(float num1, float num2, float& compensation)*** that implements the ***Kahan Summation Algorithm*** for high-precision compensated summation of two float arguments ***float num1*** and ***float num2***:

```
float kahan(float num1, float num2, float &compensation)
{
    float result;
    num2 -= compensation;
    result = num1 + num2;
    compensation = (result - num1) - num2;
    return result;
}
```

Using this function, write a C++ function ***float e(int n)*** that computes the value ***e*** by the following formula:

$$e = \sum_{k=0}^{n} \frac{1}{k!} = \frac{1}{1} + \frac{1}{1} + \frac{1}{1*2} + \frac{1}{1*2*3} + \cdots$$

Recall that the factorial of non-positive numbers equals to *1* by definition.

The initial value of ***float compensation*** is ***0.0***.

```
float(Nt a){ float sum2 1 , float product =1
for (int i=2 ; i<=a; i++){ sum2Lahan(sum;1/
product2product*i ; }                    /(product*i)
}
return seem;
}
```

0

See AB, 2A

Use the backside, if needed

OOP.MI.150317.L043

Section, Name and ID#:

Problem 2: Write a Java method ***public static double[] mean(double[] data)*** that takes as its argument an array of data points ***double[] data***, and returns a two-element array – the first element being the mean value of the data points and the second element being the standard deviation. The standard deviation σ of n numbers a_i is computed as:

$$\sigma = \sqrt{\frac{\sum_{i=0}^{n-1}(a_i - mean)^2}{n}}$$

```
public static double[] mean(double[] data){
    double mean = 0;
    [illegible]
    double standard deviation;
    double[] mean = new double[2]
    for(int i = 0; i < double.length; i++){
        double sum += data[i]
    mean = sum / data length;
    }
    double[1] = mean
        for(int j = 0, j < data.length-1, j ++){
            int summation; summation += Math.pow(data[i]-
                                                 - mean){
            summation = summation/data.length
    standard deviation = Math.sqrt(summation),
    mean[2] = standard deviation;
        return mean[2];
    }
```

SeAR

Use the backside, if needed

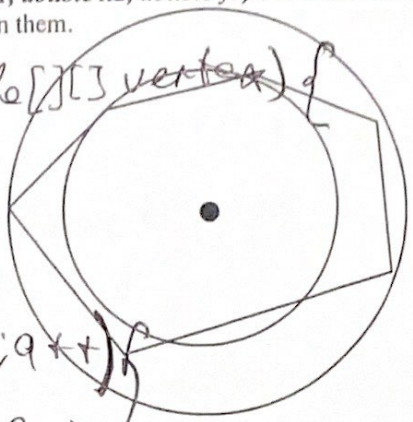
OOP.MT.CJ03TJ.L043

Section, Name and ID#:

Problem 3: Write a Java function ***public static double thickness(double[][] vertex)*** that takes as its argument a ***2-by-n*** array of polygon's vertex coordinates ***double[][] vertex*** – the ***x*** coordinates in the first row and ***y*** coordinates in the second row. It returns polygon's boundary thickness as follows:

1. Computes the center – the mean ***x*** and ***y*** vertex coordinates;
2. Returns the difference between the maximal and minimal distances from the center to the vertices.

You may assume and use a method ***double dist(double x1, double y1, double x2, double y2)*** that takes as its arguments coordinates of two points and returns the distance between them.



```
public static double thickness(double[][] vertex){
    double sum1 = 0
    double sum2 = 0
    for( int a = 0; a < vertex[0].length; a++){
        sum1 = sum1 + vertex[0][a]
        sum2 = sum2 + vertex[1][a] };

    double meanx = sum1/vertex[0].length;
    double meany = sum2/vertex[1].length;

        double herukot = 0;
        double mobikKot = 0;
    for( int b = 0; b < vertex[0].length; b++){
        if (kot < vertex[0][n], vertex[1][n]
```

2

Use the backside, if needed

Problem 3 of 4

OOP.MT.1J031J.2043

Section, Name and ID#:

Problem 4: Implement the following Java methods that swap element values between two 2D integer arrays of the same size ***int[][] a*** and ***int[][] b***:

1. ***public static void swap(int[][] a, int[][] b, int row, int col)*** – swaps element values from the specified row ***int row*** and column ***int col***;
2. ***public static void swapCol(int[][] a, int[][] b, int col)*** – swaps all element values from the specified column ***int col***;
3. ***public static void swapRow(int[][] a, int[][] b, int row)*** – swaps all element values from the specified row ***int row***. Get s bonus, if ***swapRow()*** performs faster than ***swapCol()***.

1)
```
public static void swap(int [][] a, int [][] b,
                        int row, int col)
{ int art ; art = a[row][col];
  a[row][col] = b[row][col];
  b[row][col] = art; }
```

2)
```
public static void swapCol(int[][] a, int[][] b, int col){
int z;                  for(z = 0; z < a.length; z++){
int w;
int l;
                        l = a[z][col];
                        a[z][col] = b[z][col];
                        b[z][col] = w; }
```

3 →

Use the backside, if needed

OOP.MT.17317.L043

int row){

int y, g; ant.

for (y=0; y< a[0].length, y++){

ant = a[row][y];

a[row][y] = b[2066][7];

b[row][y] = g;

g

0

see AB