Midterm 2
Introduction to Programming

Oct. 21, 2019

Name (printed):	Solution
Student ID:	

Integrity Pledge

On my honor, I affirm that I have neither given nor received inappropriate aid in the completion of this exercise.

Signed: ______________________________

Instructions: Answer all programming questions in Java. For a given problem, each part is worth the same number of points unless stated otherwise. Show your work to receive partial credit.

Additional notes:

- You have fifty minutes to complete the exam.
- Pay attention to distinctions like int versus double and String versus char.
- Try to complete short problems quickly so you have time to write code fragments and programs.
- Pay attention to whether each programming question is asking for a code fragment or a complete program. Do not write a whole program when you are only asked for a few lines of code.
- Pay attention to whether you need input from a user. If you do not need user input, the problem will say something like "the value of the variable was set elsewhere."
- If you do need user input, assume the user enters all data perfectly unless stated otherwise.
- You do not need to include import statements in your code.
- You may abbreviate System.out.println as S.o.p., and you may abbreviate prompts.

Do not write on the back of any page.

Problem 1 (10 points; 5 per part): Trace the execution of the loops below by filling in the table to the right of each code fragment. If a loop executes infinitely, trace at least three iterations and then write "infinite loop" on the next row. Be sure to store the initial values in the table.

a)
```
final int value = 5;
int count = 1;
int product = 2;
int sum = 0;
while (count < (value – 1))
{
    product = product * value;
    sum = sum + product;
    count = count + 1;
}
```

count	product	sum
1	2	0
2	10	10
3	50	60
4	250	310

b)
```
int bound = 4;
int k = 1;
int total = 1;
while (k <= bound)
{
    total = total * k;
    ++k;
}
```

k	total
1	1
2	1
3	2
4	6
5	24

Problem 2 (4 points; 1 per part): Suppose that an array is declared and constructed with the following line of code:

```
int[] intArray = {6, 2, 19, 3, 21, 14, 12};
```

Answer each of the questions below about the array.

a) What is the index of the element 6?

0

b) What is the value of the field intArray.length?

7

Now suppose that the following line of code is executed:

```
Arrays.sort(intArray);
```

Answer the remaining questions assuming that the line above has finished executing.

c) What is the index of the element 6?

```
2
```

d) What is the value of the field intArray.length?

```
7
```

Problem 3 (16 points; 4 per part): Write the signature for a method that will perform each of the computations described below. **You do not need to write the body of the method.**

Example: Find the sum of two double values.

```
double sum(double x, double y)
```

a) Given two integers, determine if the first is divisible by the second.

```
boolean isDivisible(int first, int second)
```

b) Find the sum of a sequence of double values entered from the keyboard that ends with -1.0. (Note: the Scanner used to read the values is not constructed in the method.)

```
double sumInputs(Scanner keyboard)
```

c) Find the number of times that a given String appears in an array of Strings.

```
int countString(String str, String[] array)
```

d) Return a copy of a given array of boolean values with each true flipped to false and vice versa.

```
boolean[] flip(boolean[] array)
```

Problem 4 (20 points; 10 per part): Trace the execution of each program in the memory diagram. Then complete the message that is output to the console.

Variables declared in the header of a for-loop do not need to be traced, but every other variable should appear in a table. Be sure to include the initial value of each variable. If the contents of a variable change, use a comma, space, or some other delimiter to separate the old and new values.

a)
```
public static void main(String[] args)
{
    int[] array = {5, 9, 11, 13, 10};
    int start = 2;
    int end = array.length - 2;
    updateArray(array, start, end);
    System.out.println("Elements of array: " + Arrays.toString(array));
}

public static void updateArray(int[] array, int start, int end)
{
    for (int j = start; j < end; ++j)
    {
        array[j] = array[j] + 3;
    }
}
```

Console Output

```
Elements of array: [5, 9, 14, 13, 10]
```

main Stack Frame

Identifier	Address	Contents
array	100	1000
start	101	2
end	102	3
	103	
	104	

updateArray Stack Frame

Identifier	Address	Contents
array	200	1000
start	201	2
end	202	3
j	203	2, 3
	204	

Heap

Identifier	Address	Contents
0	1000	5
1	1001	9
2	1002	11, 14
3	1003	13
4	1004	10
length	1005	5
	1006	
	1007	
	1008	
	1009	
	1010	
	1011	
	1012	
	1013	

b)
```
public static void main (String[] args)
{
    int[] array = {5, 9, 11, 13, 10};
    int scale = 2;
    updateArray(array, scale);
    System.out.println("Elements of array: " + Arrays.toString(array));
}

public static int[] updateArray(int[] array, int factor)
{
    int[] temp = new int[array.length];

    for (int idx = 0; idx < array.length; ++idx)
    {
        temp[idx] = array[idx] * factor;
    }

    return temp;
}
```

Console Output

```
Elements of array: [5, 9, 11, 13, 10]
```

main Stack Frame

Identifier	Address	Contents
array	100	1000
scale	101	2
	102	
	103	
	104	

updateArray Stack Frame

Identifier	Address	Contents
array	200	1000
factor	201	2
temp	202	1006
idx	203	0 1 2 3 4 5
	204	

Heap

Identifier	Address	Contents
0	1000	5
1	1001	9
2	1002	11
3	1003	13
4	1004	10
length	1005	5
0	1006	0, 10
1	1007	0, 18
2	1008	0, 22
3	1009	0, 26
4	1010	0, 20
length	1011	5
	1012	
	1013	

Problem 5 (10 points): Write a code fragment that prints the minimum element in an array of doubles.

```
double[] doubleArray;  // The array is constructed and initialized elsewhere.
// Find and print the minimum element.
```

```
double min = doubleArray[0];

for (int idx = 0; idx < doubleArray.length; ++idx)
{
    if (doubleArray[idx] < min)
    {
        min = doubleArray[idx];
    }
}

System.out.println(min);



/* Notes:

1. Here is the logic of the solution: (1) Use a variable to store the smallest element.
   Start by assigning the first element to this variable. (2) Compare the variable to
   every element in the array. (3) Whenever you find an element that is smaller than
   the variable, update the variable with the new element.

2. Some clever students used the sort method of the Arrays class to solve this problem.
   With this strategy, the solution can be written in just two lines:

   Arrays.sort(doubleArray);
   System.out.println(doubleArray[0]);

   After sorting the array, the smallest element is in the first position (index 0).
   Note that the return type of the sort method is void. The method does not make a
   copy of the array; it modifies the array it is given.

3. The for-loop solution is faster than the sort solution; however, efficiency is a
   secondary concern in this class.

4. The question is only asking for a fragment of code, not an entire method. Many
   people wrote code with a return statement, which is incorrect. Make sure you read
   questions like this carefully so you understand the type of answer it is asking for.
*/
```

Problem 6 (40 points): Write a program that collects grades from the user, curves the grades so that the highest is 100, and calculates the average of the new grades. The program prints the highest grade, the amount the grades are curved, the grades with the curve added, and the average.
(Assume each grade is an integer between 0 and 100.)

Below is an example run of the program with the user input underlined and in bold:

```
Enter the grades separated by spaces and terminated by -1.
88 75 91 95 82 -1
Max grade: 95
Curve: 5
Curved grades: [93, 80, 96, 100, 87]
Average: 91.2
```

The program consists of the following five methods:

1. `int readGrades(Scanner keyboard, int[] array)`
2. `int findMax(int[] array, int size)`
3. `int[] curveGrades(int[] array, int size, int value)`
4. `double averageGrades(int[] array)`
5. `void main(String[] args)`

You will need to write the bodies of methods 1, 3, and 5.

Do not write anything on this page—start your code on the next page.

a) (10 points) Write the body of the method readGrades. This method reads a sequence of grades entered by the user on a single line. The grades are separated by spaces and terminated by the value -1. The method stores the grades in a given array and returns the number of grades. The method is also given a Scanner that is connected to the keyboard.

Additional notes:

- The method does not prompt the user to enter the grades. This is done in the main method.
- You may assume that the capacity of the given array is large enough to store all the grades.

```
public static int readGrades(Scanner keyboard, int[] array)
```

```
{
    int numGrades = 0;
    int grade = keyboard.nextInt();

    while (grade != -1)
    {
        array[numGrades] = grade;
        ++numGrades;
        grade = keyboard.nextInt();
    }

    return numGrades;
}



/* Notes:

1. We want to store the grades in consecutive elements of the given oversize array.
   Here are the steps: (1) Declare a variable to store the number of grades and
   initialize it to 0. (2) Perform a priming read by calling nextInt on the Scanner and
   assign the first input value to a variable. (3) Use a while loop to check if the
   input is -1 (the sentinel value). (4) If the input is a grade, assign it to the next
   empty element of the array and increment the number of grades. (Note that the
   current number of grades is equal to the index of the next empty element.) (5) Read
   the next input value at the end of the loop body so the loop condition checks the
   next input.

2. The problem says the given array will be large enough to store all the grades. It is
   easy to modify the code to handle the case where the array is too small. Simply
   change the loop condition to the following:

   grade != -1 && numGrades < array.length

   Now the loop will stop if the number of grades (the size of the oversize array)
   equals the capacity (the length of the array). This prevents the method from trying
    to assign a grade to an element that does not exist.
*/
```

b) (10 points) Write the body of the method curveGrades. This method takes an oversize array of size grades, adds a given value to each grade, and returns the new grades in a perfect size array. (That is, the array returned by the method should contain all the curved grades and no extra space.)

```
public static int[] curveGrades(int[] array, int size, int value)
```

```
{
    int[] curved = new int[size];

    for (int idx = 0; idx < size; ++idx)
    {
        curved[idx] = array[idx] + value;
    }

    return curved;
}
```

c) (20 points) Write the main method of the program. This method should use the two methods you wrote in parts (a) and (b), which have the following signatures:

```
int readGrades(Scanner keyboard, int[] array)
int[] curveGrades(int[] array, int size, int value)
```

The main method should also use the two methods below. (You do not have to write these methods.)

```
int findMax(int[] array, int size)
double averageGrades(int[] array)
```

findMax returns the maximum element in an oversize array of size integers. (Use this method to calculate the amount to curve each grade.) averageGrades returns the average of an array of grades.

Make sure you construct the Scanner and the array that are passed to readGrades. Give the array a capacity of 100.

```
public static void main(String[] args)
```

```
{
    Scanner keyboard = new Scanner(System.in);
    int[] grades = new int[100];

    System.out.println("Enter the grades separated by spaces and terminated by -1.");
    int numGrades = readGrades(keyboard, grades);

    int maxGrade = findMax(grades, numGrades);

    int curve = 100 – maxGrade;

    int[] curvedGrades = curveGrades(grades, numGrades, curve);

    double average = averageGrades(curvedGrades);

    System.out.println("Max grade: " + maxGrade);
    System.out.println("Curve: " + curve);
    System.out.println("Curved grades: " + Arrays.toString(curvedGrades));
    System.out.println("Average: " + average);
}
```