Ahmadu Bello University
Department of Mathematics
First Semester Examinations – April 2016
COSC211: Object Oriented Programming I – Marking Sceme

Attempt ***Four*** questions — Time: 120 mins

1. (*a*) Design a class called `Author` that contains the following.

   (*i*) Three *private instance variables*: `name` (`String`), `email` (`String`), and `gender` (`char` of either `'m'` or `'f'`).
   (*ii*) One *constructor* to initialize `name`, `email` and `gender` with given values.
   (*iii*) Public *getters* and *setters*: `getName()`, `getEmail()`, `setEmail()`, and `getGender()`. There are no setters for `name` and `gender`, as these attributes cannot be changed.
   (*iv*) A `toString()` method that returns `"author-name (gender) at email"`, eg, `"Aliyu Garba (m) at galiyu@abu.edu.ng"`.

```
class Author { // 1 mark
    private String name; // 2 marks
    private String email;
    private char gender;

    //Constructor // 3 marks
    public Author(String name, String email, char gender) {
        this.name = name;
        this.email = email;
        this.gender = gender;
    }

    //getters
    public String getName() { // 2 marks
        return name;
    }

    public String getEmail() { // 2 marks
        return email;
    }

    public char getGender() { // 2 marks
        return gender;
    }

    // setter
    public void setEmail(String newEmail) { // 2 marks
        email = newEmail;
    }

    public String toString() { // 2 marks
        return String.format("%s (%c) at %s",name,gender,email);
    }
}
```

(*b*) Write a test program called `TestAuthor` to test the constructor and the public methods. Try changing the email of an author.

```
public class TestAuthor {
    public static void main(String[] args) {
        Author author = new Author("Uthman Garba Aliyu",
            "uthhmangarba@gmail.com",'M'); // 2 marks
        System.out.println(author);
        author.setEmail("uthman@gmail.com"); // 1 mark
        System.out.println(author); // 1 mark
    }
}
```

2. (*a*) Write a complete program that will calculate and display the sum of the squares of the first *n* positive integers: $1^2 + 2^2 + 3^2 + ... + n^2$. Use a `for()` loop.

```
import java.util.Scanner;  // 12 marks

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

        System.out.print("Enter the number of terms: ");
        int n = input.nextInt();

        int sum = 0;
        for(int i = 1; i <= n; i++)
            sum += i * i;

        System.out.printf("The sum is %,d\n", sum);
    }
}
```

(*b*) Examine the following code.

```
do
     System.out.print("Enter a number (1 to 9): ");
     number = input.nextInt();
while (number < 1 || number > 9);
```

What is the purpose of this code? Explain how it works. Assume that `input` has been declared as a `Scanner` object.

The purpose of the code is to ensure that the user enters a number between one and nine inclusive. If the number is not in that range then he or she is prompted again. // 4 marks

The `do ... while` loop examines the input at the end of the loop and returns to the loop if `number` is outside the range, and prompts again for input. Otherwise the code exits from the loop. // 4 marks

3. Write a program called `GradesStatistics`, which reads in an array of `n` grades (of `int` between 0 and 100, inclusive) and displays the average, the minimum, the maximum, and the standard deviation. Your program shall check for valid input. You should keep the grades in an `int[]` and use a method for each of the computations. Your output shall look like the following.

```
Enter the number of students: 4
Enter the grade for student 1: 50
Enter the grade for student 2: 51
Enter the grade for student 3: 56
Enter the grade for student 4: 53
The average is 52.5
The minimum is 50
The maximum is 56
The standard deviation is 2.29128784747792
```

Hint: The formula for calculating standard deviation is: $\sqrt{\frac{1}{n}\sum(x-\mu)^2}$,

where μ is the mean of the marks x.

```
import java.util.Scanner; // 1 mark

class GradeStatistics{   // 1 mark
    public static int[] grades;   // 2 marks
    public static int n;   // 1 mark

    public static void readGrades(){   // 4 marks
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        n = input.nextInt();
        System.out.println();

        grades = new int[n];
        for(int i = 0; i < n; i++){
            System.out.print("Enter grade for student " + (i + 1) + ": ");
            int grd = input.nextInt();
            if(grd < 0 || grd > 100){
                System.out.println("Invalid grade!");
                i--;
            } else {
                grades[i] = grd;
                System.out.println();
            }
        }
    }

    public static double getAverage(){ // 2 marks
        int sum = 0;
        for(int i = 0; i < grades.length; i++){
            sum += grades[i];
        }

        return sum / grades.length;
    }
    public static double getMin() {  // 2 marks
        int min = grades[0];
        for(int i =0; i<grades.length; i++) {
            if(grades[i] < min)
                min = grades[i];
        }

        return min;
    }

    public static double getMax(){  // 2 marks
        int max = grades[0];
        for(int i =0; i<grades.length; i++) {
            if(grades[i] > max)
                max = grades[i];
        }

        return max;
    }

    public static double getStdDev(){  // 2 marks
        int summation = 0;
        for(int i = 0; i<grades.length; i++) {
            summation += (grades[i] * grades[i] - getAverage() *
                getAverage());
        }

        return Math.sqrt(summation/n);
    }

    public static void main(String[] args){
        readGrades(); // 1 mark
        // 2 marks
        System.out.printf("The average is %.2f\n",getAverage());
        System.out.printf("The minimum is %.2f\n",getMin());
        System.out.printf("The maximum is %.2f\n",getMax());
        System.out.printf("The standard deviation is %.4f\n",getStdDev());
    }
}
```

4. (*a*) Examine the following code and answer the questions that follow.

```
 1. // Record.java
 2.
 3. import java.io.*;
 4.
 5. public class Record {
 6.     public static void main(String[] args)
 7.             throws IOException {
 8.         File txtFle = new File("record.txt");
 9.         PrintWriter output = new
10.             PrintWriter(txtFle);
11.         output.println("Mohammed Ali");
12.         output.println("Heavyweight boxer");
13.         output.println("Olympic Medalist");
14.         output.close();
15.     }// end of main()
16. }// end of class Record
```

(*i*) Explain the need of the statement on line seven.
(*ii*) Explain the effect of lines nine and ten.
(*iii*) If, after running this program, the file 'record.txt' was opened in a text editor, what would we see?

(*i*) The program performs output to a file and thus might throw an `IOException`. Line seven this exception. // 3 marks
(*ii*) Lines nine and ten create an object called '`output`' of type `PrintWriter` that will create and open a file called '`record.txt`' using the file object `txtFle`. // 3 marks
(*iii*) The output will be

```
Mohammed Ali
Heavyweight boxer
Olympic Medalist
```
// 3 marks

(*b*) Write a program that will prompt the user to enter the size of a circle's radius, and will display its area.

// 11 marks

```
import java.util.Scanner;

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

        System.out.print("Enter the radius: ");
        double radius = input.nextDouble();

        double area = Math.PI * radius * radius;
        System.out.printf("The area is %f\n", area);
    }
}
```

5. (*a*) Explain the meaning of *method overloading* with an example.

Method Overloading is a feature that allows a class to have two or more methods having same name if their argument lists are different // 4 marks

Example // 4 marks

```
class DisplayOverloading {
    public void disp(char c) {
        System.out.println(c);
    }

    public void disp(char c, int num) {
        System.out.println(c + " " + num);
    }
}
```

(*b*) What will be the output of each of the two programs given below

```
public class DemonstrateMethod {
    static int methodOne(int i) {
        return methodTwo(i *= 11);
    }

    static int methodTwo(int i) {
        return methodThree(i /= 11);
    }

    static int methodThree(int i) {
        return methodFour(i -= 11);
    }

    static int methodFour(int i) {
        return i += 11;
    }

    public static void main(String[] args) {
        System.out.println(methodOne(11));
    }
}// end of class DemonstrateMethod

public class MethodOverloading {
    public static int average(int n1, int n2) {
        return (n1+n2)/2;
    }
    public static double average(double n1,
        double n2) {
        return (n1+n2)/2;
    }
    public static int average(int n1, int n2,
            int n3) {
        return (n1+n2+n3)/3;
    }

    public static void main(String[] args) {
        System.out.println(average(1, 2));
        System.out.println(average(1.0, 2.0));
        System.out.println(average(1, 2, 3));
```

```
            System.out.println(average(1.0, 2));
        }// end main
    }// end class
```

```
11      // 6 marks

1       // 2 marks
1.5     // 1 mark
2       // 1 mark
1.5     // 2 marks
```

6. (*a*) T-shirts are available in three sizes; small, medium and large. They may be white or coloured. White T-shirts cost ₦1000, ₦1100 and ₦1200 for the small, medium and large sizes respectively. Coloured T-shirts cost 10% more in each category.

Write a program that will ask for the size of a T-shirt (S/M/L) and its colour (W/C), and display the cost.

```
import java.util.Scanner;   // 12 marks

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

        System.out.print("Enter the size (S/M/L): ");
        char size = input.nextLine().toUpperCase().charAt(0);
        System.out.print("Enter the colour (W/C): ");
        char colour = input.nextLine().toUpperCase().charAt(0);

        double cost = 0.0;
        if (size == 'S' && colour == 'W') cost = 1000.0;
        else if (size == 'S' && colour == 'C') cost = 1000.0 * 1.1;
        else if (size == 'M' && colour == 'W') cost = 1100.0;
        else if (size == 'M' && colour == 'C') cost = 1100.0 * 1.1;
        else if (size == 'L' && colour == 'W') cost = 1200.0;
        else if (size == 'L' && colour == 'C') cost = 1200.0 * 1.1;
        else System.out.println("Invalid input");

        System.out.printf("The cost is %.2f\n", cost);
    }
}
```

(*b*) Examine the following code and answer the questions that follow.

```
switch (number) {
case 2:
    System.out.println("It is an even prime");
    break;
case 3:
case 5:
case 7:
    System.out.println("It is an odd prime");
    break;
case 4:
case 9:
    System.out.println("It is a perfect square");
    break;
case 6:
case 8:
    System.out.println("It is even");
    break;
default:
    System.out.println("It is outside the range");
}
```

(*i*) Explain the use and purpose of the `break` statement.
(*ii*) Explain, with an example, the meaning of the term *fall through* in the context of the `switch` structure.
(*iii*) Explain the effect of the above code when `number` is assigned the value 5.

(*i*) The `break` statement forces a jump out of the `switch` structure to the first following statement. This prevents the next `case` item from being processed. // 3 marks

(*ii*) When there is no `break` statement terminating a `case` the program flow *falls through* to the next `case` statement. This happens with `case 4` in the given code. The `case 4` has no `break` statement so that the program *falls through* to `case 9`. // 3 marks

(*iii*) '`It is an odd prime`' is displayed. // 3 marks