Ahmadu Bello University
Department of Mathematics
First Semester Examinations – April 2015
COSC211: Object Oriented Programming I

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

1. (*a*) Create an `Employee` class having the following.

   (*i*) Two *private final instance* fields: `payrollNo` and `employeeName`. They store the employee's file number and name, respectively.
   (*ii*) One *private instance* field: `salary`. This stores the employee's monthly salary.
   (*iii*) A single *constructor* with parameters that set the values of the three fields.
   (*iv*) *Getters* for these three fields.
   (*v*) A *setter* for the `salary` field.
   (*vi*) A *method*, `getNetSalary()`, that returns the employee's salary less tax. The taxation rate is 10%.
   (*vii*) A suitable `toString()` method that will display the `Employee` fields and the net salary.

   (*b*) Write a program that will create three `Employee` instances using the class you defined in part (*a*) above. The program should then display these three employees.

2. (*a*) The value of an investment of P naira after t years at an interest rate of r% compounded yearly is given by $P(1 + r/100)^t$. Write a program that will ask the user to input P, t and r, and will calculate and display the value of the investment. This should be done in a *sentinel*-controlled loop so that many such calculations can be performed.

   (*b*) The first term of a *GP* is 1.5, and its common ratio is 2. Write a program that will calculate the sum of ten terms. Do **not** use a formula.

3. (*a*) Here is a simple example of some source code.

```
public class Welcome{
   public static void main(String[] args){
      System.out.println("Welcome to Java " +
         "programming!");
   }
}
```

(*i*) Explain what is meant by the term *source code*.
(*ii*) What should be the name of the file that stores this code?
(*iii*) Explain how you would *compile* this code from the command line to produce a *class file*.
(*iv*) What would be the name of the resulting class file?
(*v*) Explain how you would *run* the class file from the command line. What would the output be?

(*b*) List the **seven** *numeric primitive data types* and state how many bytes of memory each requires.

Here is a statement from a Java program.

```
int num = 10;
```

Explain ***exactly*** what effect this statement has in terms of memory storage.

4. A trader sells an item for x100.00. However she allows a 5% discount when the quantity ordered is 500 or more.

Examine the following class definition that she uses in her order management system and answer the questions that follow.

```
1. public class Order{
2.    //class constants
3.    private static final double
         UNIT_PRICE = 100.0;
4.    private static final double
         DISCOUNT_RATE = 0.05;
5.    private static final int
         DISCOUNT_LEVEL = 500;
```

```
 9. public void displayMean(int a, int b){
10.    System.out.printf("The mean is %f\n",
          getMean(a, b);
11. }
```

In what follows assume that `myObject` is an instance of the class in which the methods have been declared.

(*i*) What would be returned by `myObject.getMean(2, 3)`?
(*ii*) What would be returned by `myObject.getMean(5.5, 6.5)`?
(*iii*) The first two methods have the same name. Explain how the correct method is called by a statement like the following.

```
double z = myObject.getMean(x, y);
```

What is this technique called?
(*iv*) Explain the significance of the `double` keyword on line 2. What would be the result of the call to `myObject.getMean(2, 3)` if the `double` keyword were missing from line 2?
(*v*) Explain the significance of the `void` keyword on line 9.
(*vi*) Explain which `getMean()` method is called by line 10.


```
 6.
 7.    //instance variables
 8.    private int orderNo;
 9.    private int quantity;
10.
11.    //class variable
12.    private static int numOrders;
13.
14.    //constructor
15.    public Order(int quantity){
16.       this.quantity = quantity;
17.       numOrders = numOrders + 1;
18.       orderNo = numOrders;
19.    }//end of constructor
20.
21.    //setter
22.    public void setQuantity(int quantity){
23.       this.quantity = quantity;
24.    }
25.
26.    //getters
27.    public int getOrderNo(){
28.       return orderNo;
29.    }
30.
31.    public int getQuantity(){
32.       return quantity;
33.    }
34.
35.    public static int getNumOrders(){
36.       return numOrders;
37.    }
38.
39.    public double getCost(){
40.       double cost = quantity * UNIT_PRICE;
41.
42.       //deduct discount if applicable
43.       if(quantity >= DISCOUNT_LEVEL)
44.          cost = cost - cost * DISCOUNT_RATE;
45.    }//end of getCost()
```

```
46.
47.    public String toString(){
48.       return String.format("Order No: %04d\n" +
             "Quantity: %d\nCost: %.2f",
             orderNo, quantity, getCost());
50.    }//end of toString()
51. }//end of class Order
```

(*a*) Explain the difference between an *instance variable* (line 8) and a *class variable* (line 12).

(*b*) Explain why the method `getNumOrders()` (line 35) has to be *static*.

(*c*) Explain the effect and purpose of lines 17 and 18 in the constructor.

(*d*) Explain the use of the `this` keyword on line 23.

(*e*) Write a program that will create and display two `Order` *instances*. One of them should be large enough to attract a discount. The number of orders placed should also be displayed.

5. (a) Examine the program code below and answer the questions that follow.

```
 1. //Grade.java
 2.
 3. import java.util.Scanner;
 4.
 5. public class Grade{
 6.    public static void main(String[] args){
 7.       Scanner input = new Scanner(system.in);
 8.
 9.       System.out.print("Enter your mark " +
             "(integer 0 to 100): ");
10.       int mark = input.nextDouble();
11.
12.       if(mark >= 70)
13.          grade = "A";
14.       else (mark >= 60)
15.          grade = 'B';
16.       else if(mark >= 50)
```



```
17.          grade = 'C';
18.       else if(mark >= 45)
19.          grade = 'D';
20.       else if(mark >= 40)
21.          grade = 'E';
22.       else
23.          grade = 'F'
24.
25.       System.out.println("The grade is " grade);
26.
27. }//end of class Grade
```

(*i*) There are eight errors in this code. Give the line number of each statement that has an error in it and correct the statement.
(*ii*) Explain what the program will do when all the errors have been corrected.

(*b*) Examine the following code snippet.

```
1. int i = 0
2. int sum = 0;
3.
4. while(i < 3){
5.    i++;
6.    sum += i * i;
7. }
8.
9. System.out.printf("Sum = %d\n", sum);
```

Construct a *trace* of this code. What is its final output?

6. The following methods are defined within some class.

```
1. public double getMean(int a, int b){
2.    return ((double)a + b) / 2;
3. }
4.
5. public double getMean(double a, double b){
6.    return (a + b) / 2;
7. }
8.
```