Object Oriented Programming Language I

COSC 211

MULTIPLE QUESIONS

Section 1.2 What is a Computer?

1.1 ________ is the physical aspect of the computer that can be seen.

[✓ A. Hardware]
B. Software
C. Operating system
D. Application program

1.2 __________ is the brain of a computer.

A. Hardware
[✓ B. CPU]
C. Memory
D. Disk

1.3 The speed of the CPU may be measured in __________.

A. megabytes
B. gigabytes
C. megahertz
[✓ D. gigahertz]

1.4 Why do computers use zeros and ones?

A. because combinations of zeros and ones can represent any numbers and characters.
B. because digital devices have two stable states and it is natural to use one state for 0 and the other for 1.
C. because binary numbers are simplest.
D. because binary numbers are the bases upon which all other number systems are built.

1.5 One byte has ________ bits.

A. 4
[✓ B. 8]
C. 12
D. 16

1.6 Which of the following are storage devices?

A. floppy disk
B. hard disk
[✓ C. flash stick]
D. CD-ROM

1.7 ____________ is a device to connect a computer to a local area network (LAN).

A. Regular modem
B. DSL
C. Cable modem
D. NIC

Section 1.3 Program Languages

1.8 ____________ are instructions to the computer.

A. Hardware
B. Software
[✓ C. Programs]
D. Keyboards

1.9 Computer can execute the code in ____________.

[✓ A. machine language]
B. assembly language
C. high-level language
D. none of the above

1.10 ____________ translates high-level language program into machine language program.

A. An assembler
[✓ B. A compiler]
C. CPU
D. The operating system

Section 1.4 Operating Systems

1.11 ____________ is an operating system.

A. Java
B. C++
[✓ C. Windows XP]
D. Visual Basic
E. Ada

1.12 ______________ is a program that runs on a computer to manage and control a computer's activities.

[✓ A. Operating system]
B. Java
C. Modem
D. Interpreter
E. Compiler

Section 1.5 Java, World Wide Web, and Beyond
1.13 Java was developed by _____________.

[✓ A. Sun Microsystems]
B. Microsoft
C. Oracle
D. IBM
E. Cisco Systems

1.14 Java ___________ can run from a Web browser.

[✓ A. applications]
B. applets
C. servlets
D. Micro Edition programs

1.15 _________ is an object-oriented programming language.

A. Java
B. C++
C. C
D. C#
E. Python

1.16 ________ is interpreted.

A. Java
B. C++
C. C
D. Ada
E. Pascal

1.17 ________ is Architecture-Neutral.

A. Java
B. C++
C. C
D. Ada
E. Pascal

Section 1.6 The Java Language Specification, API, JDK, and IDE
1.18 ________ is a technical definition of the language that includes the syntax and semantics of the Java programming language.

A. Java language specification
B. Java API
C. Java JDK
D. Java IDE

1.19 ________ contains predefined classes and interfaces for developing Java programs.

A. Java language specification
B. Java API
C. Java JDK
D. Java IDE

1.20 ________ consists of a set of separate programs for developing and testing Java programs, each of which is invoked from a command line.

A. Java language specification
B. Java API
C. Java JDK
D. Java IDE

1.21 ________ provides an integrated development environment (IDE) for rapidly developing Java programs. Editing, compiling, building, debugging, and online help are integrated in one graphical user interface.

A. Java language specification
B. Java API
C. Java JDK

B. .obj
C. .class
D. .exe

1.32 Which of the following lines is not a Java comment?

A. /** comments */
B. // comments
C. -- comments
D. /* comments */
E. ** comments **

1.33 Which of the following are the reserved words?

A. public
B. static
C. void
D. class

1.34 Every statement in Java ends with ________.

A. a semicolon (;)
B. a comma (,)
C. a period (.)
D. an asterisk (*)

1.35 A block is enclosed inside __________.

A. parentheses
B. braces
C. brackets
D. quotes

Section 1.9 Programming Style and Documentation

1.36 Programming style is important, because ______________.

A. a program may not compile if it has a bad style
B. good programming style can make a program run faster
C. good programming style makes a program more readable
D. good programming style helps reduce programming errors

1.37 Analyze the following code.

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

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

A. Both I and II can compile and run and displa
Welcome to Java, but the code in II has a bette
style than I.
B. Only the code in I can compile and run and display Welcome to Java.
C. Only the code in II can compile and run and display Welcome to Java.
D. Both I and II can compile and run and displa
Welcome to Java, but the code in I has a bette
style than II.

1.38 Which of the following code has the bes
style?

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

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

```
III:
public class Test {
```

D. Java IDE

Section 1.7 A Simple Java Program

1.22 The main method header is written as:

A. public static void main(string[] args)
B. public static void Main(String[] args)
C. public static void main(String[] args)
D. public static main(String[] args)
E. public void main(String[] args)

1.23 Which of the following statements is correct?

A. Every line in a program must end with a semicolon.
B. Every statement in a program must end with a semicolon.
C. Every comment line must end with a semicolon.
D. Every method must end with a semicolon.
E. Every class must end with a semicolon.

1.24 Which of the following statements is correct to display Welcome to Java on the console?

A. System.out.println('Welcome to Java');
B. System.out.println("Welcome to Java");
C. System.println('Welcome to Java');
D. System.out.print('Welcome to Java');
E. System.out.print("Welcome to Java");

Section 1.8 Creating, Compiling, and Executing a Java Program

1.25 The JDK command to compile a class in the file Test.java is

A. java Test
B. java Test.java
C. javac Test.java
D. javac Test
E. JAVAC Test.java

1.26 Which JDK command is correct to run a Java application in ByteCode.class?

A. java ByteCode
B. java ByteCode.class
C. javac ByteCode.java
D. javac ByteCode
E. JAVAC ByteCode

1.27 Java compiler translates Java source code into __________.

A. Java bytecode
B. machine code
C. assembly code
D. another high-level language code

1.28 __________ is a software that interprets Java bytecode.

A. Java virtual machine
B. Java compiler
C. Java debugger
D. Java API

1.29 Suppose you define a Java class as follows:

```
public class Test {

}
```

In order to compile this program, the source code should be stored in a file named

A. Test.class
B. Test.doc
C. Test.txt
D. Test.java
E. Any name with extension .java

1.30 The extension name of a Java bytecode file is

A. .java
B. .obj
C. .class
D. .exe

1.31 The extension name of a Java source code file is

A. .java

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

IV:

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

A. I
B. II
C. III
D. IV

Section 1.10 Programming Errors

1.39 if a program compiles fine, but it produces an incorrect result, then the program suffers from ___________.

A. a compilation error
B. a runtime error
C. a logic error
D. a syntax error

1.40 If you forget to put a closing quotation mark on a string, what kind error will be raised?
A. a compilation error
B. a runtime error
C. a logic error

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu1. Create an `Employee` class that might be used to manage employee records. It should have three fields: `FileNo` (`int`), `Name` (`String`), `Salary` (`double`)). It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the three fields.

The setters for `FileNo` and `Salary` should ensure that they are not negative.

It should have two methods: `calculateTax()` that returns a `double`, `calculateNetSalary()` that returns a `double`.

Assume that the tax rate is 10%. Tax is deducted from the salary to give the net salary.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Question 1
//Employee.java

public class Employee{
    private int fileNo;
    private String name;
    private float salary;

    //getters
    public int getFileNo(){
        return fileNo;
    }//end of getFileNo()

    public String getName(){
        return name;
    }//end of getName()

    public float getSalary(){
        return salary;
    }//end of getSalary()

    //setters
    public void setFileNo(int fileNo){
        if(fileNo < 0) fileNo = 0;
        this.fileNo = fileNo;
    }//end of setFileNo

    public void setName(String name){
        this.name = name;
    }//end of setName()
    public void setSalary(float salary){
        if(salary < 0.0f) salary = 0.0f;
        this.salary = salary;
    }//end of setSalary()

    //constructor
    public Employee(int fileNo, String name, float salary){
        setFileNo(fileNo);
        setName(name);
        setSalary(salary);
    }//end of constructor

    //no-args constructor
    public Employee(){
        this(0, "noname", 0.0f);
    }//end of no-args constructor

    public double calculateTax(){
        return salary * 0.1;
    }//end of calculateTax()

    public double calculateNetSalary(){
        return salary - calculateTax();
    }//end of calculateNetTSalary()

    public static void main(String[] args){
        Employee employee1 = new Employee(1, "Aliyu", 100000.0f);
        Employee employee2 = new Employee(-9, "Hamza", -100.0f);
        Employee employee3 = new Employee();
        employee3.setFileNo(2);
        employee3.setName("Gloria");
        employee3.setSalary(100000.0f);

        System.out.printf("File No: %d\nName: %s\nSalary: %.2f\n" +
                          "Tax: %.2f\nNet Salary: %.2f\n",
                          employee1.fileNo, employee1.name,
                          employee1.salary, employee1.calculateTax(),
                          employee1.calculateNetSalary());
        System.out.printf("File No: %d\nName: %s\nSalary: %.2f\n" +
                          "Tax: %.2f\nNet Salary: %.2f\n",
                          employee2.fileNo, employee2.name,
                          employee2.salary, employee2.calculateTax(),
                          employee2.calculateNetSalary());
        System.out.printf("File No: %d\nName: %s\nSalary: %.2f\n" +
                          "Tax: %.2f\nNet Salary: %.2f\n",
                          employee3.fileNo, employee3.name,
                          employee3.salary, employee3.calculateTax(),
                          employee3.calculateNetSalary());
    }//end of main()
}//end of class Employee
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu2. Create a `Triangle` class that has three `double` fields: `sideA`, `sideB`, `sideC`. It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the three fields.

The setters should ensure that the three sides have lengths that are not negative. There should be two methods: `calculateArea()` that returns a `double`, `calculateAngleA()` that returns a `double`. [The area of a triangle is given by $\sqrt{s(s-a)(s-b)(s-c)}$ where $s=(a+b+c)/2$. Use the cosine formula to calculate angle A.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu2
//Triangle.java

public class Triangle{
    double sideA;
    double sideB;
    double sideC;

    //getters
    public double getSideA(){
        return sideA;
    }//end of getSideA()

    public double getSideB(){
        return sideB;
    }//end of getSideB()

    public double getSideC(){
        return sideC;
    }//end of getSideC()

    //setters
    public void setSideA(double sideA){
        if(sideA < 0) sideA = 0;
        this.sideA = sideA;
    }//end of setSideA()

    public void setSideB(double sideB){
        if(sideB < 0) sideB = 0;
        this.sideB = sideB;
    }//end of setSideB()

    public void setSideC(double sideC){
        if(sideC < 0) sideC = 0;
        this.sideC = sideC;
    }//end of setSideC()

    //constructor
    public Triangle(double sideA, double sideB, double sideC){
        setSideA(sideA);
        setSideB(sideB);
        setSideC(sideC);
    }//end of constructor

    //no-args constructor
    public Triangle(){
        this(1, 1, 1);
    }//end of no-args constructor

    public double calculateArea(){
        double s = (sideA + sideB + sideC) / 2.0;
        return Math.sqrt(s * (s - sideA) * (s - sideB) * (s - sideC));
    }//end of calculateArea

    public double calculateAngleA(){
        double cosA = (sideB * sideB + sideC * sideC - sideA * sideA) /
                      (2 * sideB * sideC);
        return Math.acos(cosA);
    }//end of calculateAngleA

    public static void main(String[] args){
        Triangle triangle1 = new Triangle(3.5, 4.6, 5.7);
        Triangle triangle2 = new Triangle(5.0, 4.0, 3.0);
        Triangle triangle3 = new Triangle();

        System.out.printf("Area: %.2f\nA: %.3f rad\n",
            triangle1.calculateArea(), triangle1.calculateAngleA());
        System.out.printf("Area: %.2f\nA: %.3f rad\n",
            triangle2.calculateArea(), triangle2.calculateAngleA());
        System.out.printf("Area: %.2f\nA: %.3f rad\n",
            triangle3.calculateArea(), triangle3.calculateAngleA());
    }//end of main()
}//end of class Traiangle
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu3. Create a `Parallelogram` class that has three `double` fields: `side1`, `side2`, `angle`, where `side1` and `side2` are adjacent sides and `angle` is the angle between them. It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the three fields.

The setter for `angle` should ensure that its value is greater than 0° and less than 180°. The setter for the sides should ensure that they are not negative.

There should be two methods: `calculateArea()` that returns a `double`, `calculatePerimeter()` that returns a `double`. [The area of a parallelogram is given by `side1 * side2 * sin(angle)`.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu3
//Parallelogram.java

public class Parallelogram{
    private double side1;
    private double side2;
    private double angle;

    //getters
    public double getSide1(){
        return side1;
    }//end of getSide1()

    public double getSide2(){
        return side2;
    }//end of getSide2()

    public double getAngle(){
        return angle;
    }//end of getAngle()

    //setters
    public void setSide1(double side1){
        if (side1 < 0) side1 = 0;
        this.side1 = side1;
    }//end of setSide1()

    public void setSide2(double side2){
        if (side2 < 0) side2 = 0;
        this.side2 = side2;
    }//end of setSide2()
    public void setAngle(double angle){
        if (angle < 0) angle = 0;
        if (angle > 180) angle = 180;
        this.angle = angle;
    }//end of setAngle()

    //constructor
    public Parallelogram(double side1, double side2, double angle){
        setSide1(side1);
        setSide2(side2);
        setAngle(angle);
    }//end of constructor

    //no-args constructor
    public Parallelogram(){
        this(1.0, 1.0, 60.0);
    }//end of no-args constructor

    public double calculateArea(){
        return side1 * side2 * Math.sin(angle * Math.PI / 180);
    }//end of calculateArea()

    public double calculatePerimeter(){
        return 2 * (side1 + side2);
    }//end of calculatePerimeter()

    public static void main(String[] args){
        Parallelogram parallelogram1 = new Parallelogram(4.5, 6.5, 34.5);
        Parallelogram parallelogram2 = new Parallelogram(7.5, 9.5, 45.5);
        Parallelogram parallelogram3 = new Parallelogram();

        System.out.printf("Area: %.2f\nPerimeter: %.2f\n",
                          parallelogram1.calculateArea(),
                          parallelogram1.calculatePerimeter());
        System.out.printf("Area: %.2f\nPerimeter: %.2f\n",
                          parallelogram2.calculateArea(),
                          parallelogram2.calculatePerimeter());
        System.out.printf("Area: %.2f\nPerimeter: %.2f\n",
                          parallelogram3.calculateArea(),
                          parallelogram3.calculatePerimeter());
    }//end of main()
}//end of class Parallelogram
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu4. Create a `Student` class that has fields `name` (`String`), `matriculationNo` (`String`) and three integer fields that store the marks of three continuous assessment tests. It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two `String` fields.

The setters for the marks should ensure that each mark lies in the closed interval from 0 to 100.

There should be a method that returns the average of the three marks (as a `double`).

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods. In each case the marks should be entered by the user from the keyboard.

```
//Qu4
//Student.java

public class Student{
    private String name;
    private String matriculationNo;
    private int mark1;
    private int mark2;
    private int mark3;

    //getters
    public String getName(){
        return name;
    }//end of getName()

    public String getMatriculationNo(){
        return matriculationNo;
    }//end of getMatriculationNo()

    public int getMark1(){
        return mark1;
    }//end of getMark1()

    public int getMark2(){
        return mark2;
    }//end of getMark2()

    public int getMark3(){
        return mark3;
    }//end of getMark3()

    //setters
    public void setName(String name){
        this.name = name;
    }//end of setName()

    public void setMatriculationNo(String matriculationNo){
        this.matriculationNo = matriculationNo;
    }//end of setMatriculationNo()

    public void setMark1(int mark1){
        if (mark1 < 0) mark1 = 0;
        if (mark1 > 100) mark1 = 100;
        this.mark1 = mark1;
    }//end of setMark1()

    public void setMark2(int mark2){
        if (mark2 < 0) mark2 = 0;
        if (mark2 > 100) mark2 = 100;
        this.mark2 = mark2;
    }//end of setMark2()

    public void setMark3(int mark3){
        if (mark3 < 0) mark3 = 0;
        if (mark3 > 100) mark3 = 100;
        this.mark3 = mark3;
    }//end of setMark3()

    //constructor
    public Student(String name, String matriculationNo){
        setName(name);
        setMatriculationNo(matriculationNo);
    }//end of constructor

    //no-arg constructor
    public Student(){
        this("noname", "nonum");
    }//end of no-args constructor

    public double averageCAMark(){
        return (double)(mark1 + mark2 + mark3) / 3.0;
    }//end of averageCAMark()

    public static void main(String[] args){
        Student student1 = new Student("Bashir", "U12CS1234");
        Student student2 = new Student("Mariam", "U13CS4321");
        Student student3 = new Student();

        student1.setMark1(50);
        student1.setMark2(60);
        student1.setMark3(40);
        student2.setMark1(60);
        student2.setMark2(70);
        student2.setMark3(80);
        student3.setName("Marcus");
        student3.setMatriculationNo("U11MT4567");
        student3.setMark1(30);
        student3.setMark2(40);
        student3.setMark3(50);

        System.out.printf("Name: %s\nMatriculation No: “ +
                ”%s\nAverage Mark: %.1f\n", student1.name,
                student1.matriculationNo, student1.averageCAMark());
        System.out.printf("Name: %s\nMatriculation No: “ +
                ”%s\nAverage Mark: %.1f\n", student2.name,
                student2.matriculationNo, student2.averageCAMark());
        System.out.printf("Name: %s\nMatriculation No: “ +
                ”%s\nAverage Mark: %.1f\n", student3.name,
                student3.matriculationNo, student3.averageCAMark());
    }//end of main()
}//end of class Student
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu5. Create an `AP` (*arithmetical progression*) class. Its fields will be the *first term* and the *common difference*, both `double`, and the *number of terms* – an `int`. It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the three fields.

There should be a method that returns the sum of the AP (using a loop), and a method that returns the value of the *n*th term. [The *n*th term is given by $a + (n - 1)d$.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods. In each case the values of the fields should be entered by the user from the keyboard.

```
//Qu5
//AP.java

public class AP{
    private double a;         //first term
    private double d;         //common difference
    private int n;            //number of terms

    //getters
    public double getA(){
        return a;
    }//end of getA()

    public double getD(){
        return d;
    }//end of getD()

    public int getN(){
        return n;
    }//end of getN()

    //setters
    public void setA(double a){
        this.a = a;
    }//end of setA()

    public void setD(double d){
        this.d = d;
    }//end of setD()

    public void setN(int n){
        if (n < 2) n = 2;
        this.n = n;
    }//end of setN()
    //constructor
    public AP(double a, double d, int n){
        setA(a);
        setD(d);
        setN(n);
    }//end of constructor

    //no-args constructor
    public AP(){
        this(0, 0, 1);
    }//end of no-args constructor

    public double getSum(){
        double sum = 0;
        double term = a;
        for(int i = 0; i < n; i++){
            sum = sum + term;
            term = term + d;
        }
        return sum;
    }//end of getSum()

    public double nthTerm(){
        return a + (n - 1) * d;
    }//end of nthTerm

    public static void main(String[] args){
        AP ap1 = new AP(1.0, 1.0, 100);
        AP ap2 = new AP(2.5, 3.5, 50);
        AP ap3 = new AP();

        ap3.a = 1.0;
        ap3.d = 2.0;
        ap3.n = 10;

        System.out.printf("Sum: %.0f\tTerm: %.0f\n",
                ap1.getSum(), ap1.nthTerm());
        System.out.printf("Sum: %.1f\tTerm: %.1f\n",
                ap2.getSum(), ap2.nthTerm());
        System.out.printf("Sum: %.0f\tTerm: %.0f\n",
                ap3.getSum(), ap3.nthTerm());
    }//end of main()
}//end of class AP
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu6. Create a `GP` (*geometical progression*) class. Its fields will be the *first term* and the *common ratio*, both `double`, and the *number of terms* - an `int`. It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the three fields.

There should be a method that returns the sum of the GP (using a loop), and a method that returns the value of the *n*th term. [The *n*th term is given by $ar^{(n-1)}$.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods. In each case the values of the fields should be entered by the user from the keyboard.

```
//Qu6
//GP.java

public class GP{
    private double a;        //first term
    private double r;        //common ratio
    private int n;           //number of terms

    //getters
    public double getA(){
        return a;
    }//end of getA()

    public double getR(){
        return r;
    }//end of getR()

    public int getN(){
        return n;
    }//end of getN()

    //setters
    public void setA(double a){
        this.a = a;
    }//end of setA()

    public void setR(double r){
        this.r = r;
    }//end of setR()

    public void setN(int n){
        if (n < 2) n = 2;
        this.n = n;
    }//end of setN()
    //constructor
    public GP(double a, double r, int n){
        setA(a);
        setR(r);
        setN(n);
    }//end of constructor

    //no-args constructor
    public GP(){
        this(0, 0, 1);
    }//end of no-args constructor

    public double getSum(){
        double sum = 0;
        double term = a;
        for(int i = 0; i < n; i++){
            sum = sum + term;
            term = term * r;
        }
        return sum;
    }//end of getSum()

    public double nthTerm(){
        return a * Math.pow(r, (n - 1));
    }//end of nthTerm

    public static void main(String[] args){
        GP gp1 = new GP(1.0, 2.0, 10);
        GP gp2 = new GP(0.5, 1.5, 50);
        GP gp3 = new GP();

        gp3.a = 1.0;
        gp3.r = 0.5;
        gp3.n = 10;

        System.out.printf("Sum: %f\tTerm: %f\n",
                gp1.getSum(), gp1.nthTerm());
        System.out.printf("Sum: %f\tTerm: %f\n",
                gp2.getSum(), gp2.nthTerm());
        System.out.printf("Sum: %f\tTerm: %f\n",
                gp3.getSum(), gp3.nthTerm());
    }//end of main()
}//end of class GP
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu7. Create a `Student` class that has fields `name` (`String`), `matriculationNo` (`String`) and `mark` (`int`). It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the three fields.

The `mark` setter should ensure that the mark lies in the interval 0 to 100, inclusive.

There should be a method that determines the student `grade` (`char`) that returns the following: 'A' if 70 <= mark, 'B' if 60 <= mark < 70, 'C' if 50 <= mark < 60, 'D' if 45 <= mark < 50, 'E' if 40 <= mark < 45, and 'F' if mark < 40.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods. In each case the mark should be entered by the user from the keyboard and the student's grade should be displayed.

```
//Qu07
//Student.java

import java.util.Scanner;

public class Student{
    private String name;
    private String matriculationNo;
    private int mark;

    //getters
    public String getName(){
        return name;
    }//end of getName()

    public String getMatriculationNo(){
        return matriculationNo;
    }//end of getMatriculationNo()

    public int getMark(){
        return mark;
    }//end of getMark()

    //setters
    public void setName(String name){
        this.name = name;
    }//end of setName()

    public void setMatriculationNo(String matriculationNo){
        this.matriculationNo = matriculationNo;
    }//end of setMatriculationNo()
    public void setMark(int mark){
        if (mark < 0) mark = 0;
        if (mark > 100) mark = 100;
        this.mark = mark;
    }//end of setMark()

    //constructor
    public Student(String name, String matriculationNo, int mark){
        setName(name);
        setMatriculationNo(matriculationNo);
        setMark(mark);
    }//end of constructor

    //no-args constructor
    public Student(){
        this("noname", "nonum", 0);
    }//end of no-args constructor

    public char getGrade(){
        if (mark >= 70) return 'A';
        if (mark >= 60) return 'B';
        if (mark >= 50) return 'C';
        if (mark >= 45) return 'D';
        if (mark >= 40) return 'E';
        return 'F';
    }//end of getGrade()

    public static void main(String[] args){
        Student student = new Student();
        Scanner input = new Scanner(System.in);
        int count = 0;

        while (count++ < 3){
            System.out.print("Enter the student's name: ");
            student.setName(input.nextLine());

            System.out.print("Enter the student's matriculation number: ");
            student.setMatriculationNo(input.nextLine());

            System.out.print("Enter the student's mark: ");
            student.setMark(input.nextInt());
            input.nextLine();

            System.out.printf("Grade: %C\n", student.getGrade());
            System.out.println();
        }//end of loop
    }//end of main()
}//end of class Student
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu8. Create a `Pyramid` class for a *right pyramid standing on a square base*. It will have fields `baseSide` (the length of the side of the base) and `height` (the perpendicular distance from the vertex to the base). Both should be `double`. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields.

The setters should ensure that the dimensions are not negative.

There should be two methods that return doubles. One should return the *volume* of the pyramid and the other return its *slant height*. [The volume of the pyramid is given by $\frac{1}{3}Ah$, where A is the area of the base and h is its height. The slant height is given by $\sqrt{x^2/2 + h^2}$, where x is the length of the side of the base.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu08
//Pyramid.java

public class Pyramid{
    private double baseSide;
    private double height;

    //getters
    public double getBaseSide(){
        return baseSide;
    }//end of getBaseSide()

    public double getHeight(){
        return height;
    }//end of getHeight()

    //setters
    public void setBaseSide(double baseSide){
        if (baseSide < 0) baseSide = 0;
        this.baseSide = baseSide;
    }//end of setBaseSide()

    public void setHeight(double height){
        if (height < 0) height = 0;
        this.height = height;
    }//end of setHeight()

    //constructor
    public Pyramid(double baseSide, double height){
        setBaseSide(baseSide);
        setHeight(height);
    }//end of constructor

    //no-args constructor
    public Pyramid(){
        this(0.0, 0.0);
    }//end of no-args constructor

    public double calculateVolume(){
        return baseSide * baseSide * height / 3;
    }//end of calculateVolume()

    public double calculateSlantHeight(){
        return Math.sqrt(baseSide * baseSide / 2 + height * height);
    }//end of calculateSlantHeight()

    public static void main(String[] args){
        Pyramid pyramid1 = new Pyramid(4.0, 9.0);
        Pyramid pyramid2 = new Pyramid(1.0, 3.0);
        Pyramid pyramid3 = new Pyramid();

        pyramid3.baseSide = 3.6;
        pyramid3.height = 4.7;

        System.out.printf("Volume: %.2f\tSlant height: %.2f\n",
                pyramid1.calculateVolume(),
                pyramid1.calculateSlantHeight());
        System.out.printf("Volume: %.2f\tSlant height: %.2f\n",
                pyramid2.calculateVolume(),
                pyramid2.calculateSlantHeight());
        System.out.printf("Volume: %.2f\tSlant height: %.2f\n",
                pyramid3.calculateVolume(),
                pyramid3.calculateSlantHeight());
    }//end of main()
}//end of class Pyramid
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu9. Create a `Prism` class for a *right prism standing on a regular hexagonal* (6-sided) *base*. It will have fields `baseSide` (the length of the side of the base) and `height`. Both should be `double`. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields.

The setters should ensure that the dimensions are not negative.

There should be two methods that return `doubles`. One should return the *volume* of the prism and the other return its *surface area*. [The volume of the prism is given by *Ah*, where *A* is the area of the base and *h* is its height. The area of a regular hexagon of side *x* is $3x^2\cos 30°$]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu9
//Prism.java

public class Prism{
    private double baseSide;
    private double height;

    //getters
    public double getBaseSide(){
        return baseSide;
    }//end of getBaseSide()

    public double getHeight(){
        return height;
    }//end of getHeight()

    //setters
    public void setBaseSide(double baseSide){
        if (baseSide < 0) baseSide = 0;
        this.baseSide = baseSide;
    }//end of setBaseSide()

    public void setHeight(double height){
        if (height < 0) height = 0;
        this.height = height;
    }//end of setHeight()

    //constructure
    public Prism(double baseSide, double height){
        setBaseSide(baseSide);
        setHeight(height);
    }//end of constructor
    //no-args constructure
    public Prism(){
        this(0.0, 0.0);
    }//end of no-args constructure

    public double calculateVolume(){
        return 3 * baseSide * baseSide * Math.cos(30 * Math.PI / 180)
            * height;
    }//end of calculateVolume()

    public double calculateSurfaceArea(){
        return 6 * baseSide * baseSide * Math.cos(30 * Math.PI / 180) +
               6 * baseSide * height;
    }//end of calculateSurfaceArea()

    public static void main(String[] args){
        Prism prism1 = new Prism(5.6, 4.7);
        Prism prism2 = new Prism(1.0, 1.0);
        Prism prism3 = new Prism();

        prism3.setBaseSide(10.0);
        prism3.setHeight(10.0);

        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                prism1.calculateVolume(),
                prism1.calculateSurfaceArea());
        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                prism2.calculateVolume(),
                prism2.calculateSurfaceArea());
        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                prism3.calculateVolume(),
                prism3.calculateSurfaceArea());
    }//end of main()
}//end of class Prism
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu10 Create a `CompoundInterest` class having fields `principal`, `rate` and `time` all being `double`. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the three fields.

The setters should ensure that the fields are all nonnegative.

There should be one method that returns the value of an investment of the amount `principal` at the given `rate` for the given `time`. [Its value is given by $P(1+r)^t$ where *P* is the principal amount invested, *t* is the time period in years and *r* is the annual rate of interest expressed as a fraction.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its method.

```
//Qu10
//CompoundInterest.java

public class CompoundInterest{
    private double principal;
    private double rate;
    private double time;

    //getters
    public double getPrincipal(){
        return principal;
    }//end of getPrincial()

    public double getRate(){
        return rate;
    }//end of getRate()

    public double getTime(){
        return time;
    }//end of getTime()

    //setters
    public void setPrincipal(double principal){
        if (principal < 0) principal = 0;
        this.principal = principal;
    }//end of setPrincipal()

    public void setRate(double rate){
        if (rate < 0) rate = 0;
        this.rate = rate;
    }//end of setRate()
    public void setTime(double time){
        if (time < 0) time = 0;
        this.time = time;
    }//end of setTime()

    //constructor
    public CompoundInterest(double principal, double rate, double time){
        setPrincipal(principal);
        setRate(rate);
        setTime(time);
    }//end of constructor

    //no-args constructor
    public CompoundInterest(){
        this(0.0, 0.0, 0.0);
    }//end of no-args constructor

    public double calculateValue(){
        return principal * Math.pow(1 + rate, time);
    }//end of calculateValue()

    public static void main(String[] args){
        CompoundInterest interest1 =
                new CompoundInterest(1234.56, 0.05, 10.0);
        CompoundInterest interest2 =
                new CompoundInterest(10000.00, 0.10, 20.0);
        CompoundInterest interest3 = new CompoundInterest();

        interest3.setPrincipal(1000000.00);
        interest3.setRate(0.075);
        interest3.setTime(50.0);

        System.out.printf("The value of the investment will be %,14.2f\n",
                           interest1.calculateValue());
        System.out.printf("The value of the investment will be %,14.2f\n",
                           interest2.calculateValue());
        System.out.printf("The value of the investment will be %,14.2f\n",
                           interest3.calculateValue());
    }//end of main()
}//end of class CompoundInterest
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

11. T-shirt are available in three sizes: small ('S'), medium ('M') and large ('L'). They may also be white or coloured. The white T-shirts are priced at ₦2000, ₦2200 and ₦2500 each respectively. The coloured T-shirts cost 10% more. Create a `TShirt` class having a `char` field `size`, and a `boolean` field `isColoured`. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields.

There should be one method that returns the price of the shirt.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its method.

```
//Qu11
//TShirt.java

public class TShirt{
    private char size;
    private boolean isColoured;

    //getters
    public char getSize(){
        return size;
    }//end of getSize()

    public boolean getIsColoured(){
        return isColoured;
    }//end of getIsColoured()

    //setters
    public void setSize(char size){
        this.size = size;
    }//end of setSize()

    public void setIsColoured(boolean isColoured){
        this.isColoured = isColoured;
    }//end of setIsColoured

    //constructor
    public TShirt(char size, boolean isColoured){
        setSize(size);
        setIsColoured(isColoured);
    }//end of constructor

    //no-args constructor
    public TShirt(){
        this('S', false);
    }//end of no-args constructor

    public double getPrice(){
        double price = 0;
        switch (size){
            case 'S':
                price = 2000.0;
                break;
            case 'M':
                price = 2200.0;
                break;
            case 'L':
                price = 2500.0;
                break;
        }//end of switch

        if (isColoured) price = 1.10 * price;

        return price;
    }//end of getPrice()

    public static void main(String[] args){
        TShirt shirt1 = new TShirt('M', true);
        TShirt shirt2 = new TShirt('L', false);
        TShirt shirt3 = new TShirt();

        shirt3.setSize('S');
        shirt3.setIsColoured(true);

        System.out.printf("The price is %.2f\n", shirt1.getPrice());
        System.out.printf("The price is %.2f\n", shirt2.getPrice());
        System.out.printf("The price is %.2f\n", shirt3.getPrice());
    }//end of main()
}//end of class TShirt
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

12. A wooden box has no lid. Create a `Box` class having four `double` fields: `length`, `breadth`, `width`, which are the *external* dimensions of the box; and `thicknes`, being the uniform thickness of the wooden sides and base. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the four fields.

The setters should ensure that the fields are nonnegative.

There should be two methods that return a `double`. The first, `calculateWoodVolume()` should return the volume of the wood that makes up the box. The second, `calculateSurfaceArea()` should calculate the *external* surface area of the box, ignoring the open top. [The wood volume can be found by subtracting the interior volume from the total volume.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its method.

```
//Qu12
//Box.java

public class Box{
    private double length;
    private double breadth;
    private double width;
    private double thickness;

    //getters
    public double getLength(){
        return length;
    }//end of getLength()

    public double getBreadth(){
        return breadth;
    }//end of getBreadth()

    public double getWidth(){
        return width;
    }//end of getWidth()

    public double getThickness(){
        return thickness;
    }//end of getThickness()

    //setters
    public void setLength(double length){
        if (length < 0) length = 0;
        this.length = length;
    }//end of setLength()

    public void setBreadth(double breadth){
        if (breadth < 0) breadth = 0;
        this.breadth = breadth;
    }//end of setBreadth()

    public void setWidth(double width){
        if (width < 0) width = 0;
        this.width = width;
    }//end of setWidth()

    public void setThickness(double thickness){
        if (thickness < 0) thickness = 0;
        this.thickness = thickness;
    }//end of seTthickness()

    //constructor
   public Box(double length, double breadth, double width, double thickness){
        setLength(length);
        setBreadth(breadth);
        setWidth(width);
        setThickness(thickness);
    }//end of constructor

    //no-args constructor
    public Box(){
        this(1.0, 1.0, 1.0, 0.0);
    }//end of no-args construcor

    public double getWoodVolume(){
        double outerVol = length * breadth * width;
        double innerVol = (length - 2 * thickness) *
                          (breadth - 2 * thickness) *
                          (width - thickness);
        return outerVol - innerVol;
    }//end of getWoodVolume

    public double getSurfaceArea(){
        return 2 * length * width + 2 * breadth * width + length * breadth;
    }//end od getSurfaceArea()

    public static void main(String[] args){
        Box box1 = new Box(2.0, 2.0, 2.0, 0.1);
        Box box2 = new Box(2.0, 1.0, 0.5, 0.02);
        Box box3 = new Box(1.7, 3.1, 1.2, 0.05);

        System.out.printf("Wood volume: %.3f\tSurface area: %.3f\n",
                box1.getWoodVolume(), box1.getSurfaceArea());
        System.out.printf("Wood volume: %.3f\tSurface area: %.3f\n",
                box2.getWoodVolume(), box2.getSurfaceArea());
        System.out.printf("Wood volume: %.3f\tSurface area: %.3f\n",
                box3.getWoodVolume(), box3.getSurfaceArea());
    }//end of main()
}//end of class box
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

13. Create a class, `OddSum`, that can demonstrate that the sum of the first *n* positive odd integers is n^2. The class will have `n` as its only (`int`) field. The setter should ensure that `n` is greater than 1. There should be one constructor that sets the value of `n`.

There should be one method that forms the sum:

$$1+3+5+\cdots+(2n-1).$$

Write test code that uses a loop so that the sum can be tested for many valus of `n`.

```
//Qu13
//OddSum.java

public class OddSum{
    private int n;

    //getter
    public int getN(){
        return n;
    }//end of getN()

    //setter
    public void setN(int n){
        if (n <= 1) n = 2;
        this.n = n;
    }//end of setN()

    //constructor
    public OddSum(int n){
        setN(n);
    }//end of constructor

    public int getSum(){
        int sum = 0;
        for(int i = 0; i < n; i++)
            sum += 2 * i + 1;
        return sum;
    }//end of getSum()

    public static void main(String[] args){
        for(int i = 2; i <= 25; i++){
            OddSum os = new OddSum(i);
            System.out.printf("%3d %10d\n", i, os.getSum());
        }//end of loop
    }//end of main()
}//end of class OddSum
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

14. A hemisphere sits on top of a cylinder. The base radius of the cylinder is the same as the radius of the hemisphere.

Create a `Solid` class that will describe this object. It will have fields `radius` (the common radius just described) and `height` (the height of the cylinder). Both should be `double`. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields.

The setters should ensure that the dimensions are not negative.

There should be two methods that return `doubles`. One should return the *volume* of the object and the other return its *surface area*. [The volume of a sphere is given by $\frac{4}{3}$® r^3. The surface area of a sphere is $4\pi r^2$.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu14
//Solid.class

public class Solid{
    private double radius;  //the common radius of the
                            //hemisphere and the cylinder
    private double height;  //the height of the cylinder

    //getters
    public double getRadius(){
        return radius;
    }//end of getRadius()

    public double getHeight(){
        return height;
    }//end of getHeight()

    //setters
    public void setRadius(double radius){
        if (radius < 0) radius = 0;
        this.radius = radius;
    }//end of setRadius()

    public void setHeight(double height){
        if (height < 0) height = 0;
        this.height = height;
    }//end of setHeight()

    //constructor
    public Solid(double radius, double height){
        setRadius(radius);
        setHeight(height);
    }//end of constructor

    //no-args constructor
    public Solid(){
        this(0.0, 0.0);
    }//end of no-args constructor

    public double getVolume(){
        return (2.0 / 3.0) * Math.PI * radius * radius * radius +
               Math.PI * radius * radius * height;
    }//end of getVolume()

    public double getSurfaceArea(){
        return 2 * Math.PI * radius * radius +
                2 * Math.PI * radius * height +
                Math.PI * radius * radius;
    }//end of getSurfaceArea()

    public static void main(String[] args){
        Solid solid1 = new Solid(2.0, 3.0);
        Solid solid2 = new Solid(10.5, 4.7);
        Solid solid3 = new Solid(1.0, 1.0);

        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                          solid1.getVolume(), solid1.getSurfaceArea());
        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                          solid2.getVolume(), solid2.getSurfaceArea());
        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                          solid3.getVolume(), solid3.getSurfaceArea());
    }//end of main()
}//end of class Solid
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

15. A regular tetrahedron is a pyramid on a triangular base in which all the edges are equal in length. Its four faces are all equilateral triangles. Create a `Tetrahedron` class that will model this solid shape. It will have one field called `edge`. The class should have the necessary getter and setter, and two constructors – a no-args constructor and one that sets the value of the single field.

The setter should ensure that the dimension is not negative.

There should be two methods that return doubles. One should return the volume of the tetrahedron and the other return its surface area. [The volume of the tetrahedron is given by ⅓*Ah*, where *A* is the area of the base and *h* is its height. The height *h* is $\sqrt{2/3}x$, where *x* is the length of the side.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu15
//Tetrahedron.java

public class Tetrahedron{
    private double edge;

    //getter
    public double getEdge(){
        return edge;
    }//end of getEdge()

    //setter
    public void setEdge(double edge){
        if (edge < 0) edge = 0;
        this.edge = edge;
    }//end of setEdge()

    //constructor
    public Tetrahedron(double edge){
        setEdge(edge);
    }//end of constructor

    //no-args constructor
    public Tetrahedron(){
        this(1.0);
    }//end of no-args constructor

    private double faceArea(){
        return 0.5 * edge * edge * Math.sin(60 * Math.PI /180);
    }//end of faceArea()
    public double getVolume(){
        return Math.sqrt(2.0 / 3.0) * edge * faceArea();
    }//end of getVolume()

    public double getSurfaceArea(){
        return 4 * faceArea();
    }//end of getSurfaceArea()

    public static void main(String[] args){
        Tetrahedron tet1 = new Tetrahedron(1.0);
        Tetrahedron tet2 = new Tetrahedron(10.5);
        Tetrahedron tet3 = new Tetrahedron(0.05);

        System.out.printf("Volume: %.3f\tSurface area: %.3f\n",
                tet1.getVolume(), tet1.getSurfaceArea());
        System.out.printf("Volume: %.3f\tSurface area: %.3f\n",
                tet2.getVolume(), tet2.getSurfaceArea());
        System.out.printf("Volume: %.3f\tSurface area: %.3f\n",
                tet3.getVolume(), tet3.getSurfaceArea());
    }//end of main()
}//end of class Tetrahedron
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu16. Create a `Triangle` class that has three `double` fields: `sideA`, `sideB`, `angleC`. It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the three fields.

The setters should ensure that the two sides have lengths that are not negative and that the angle lies between 0° and 180°.

There should be two methods: `calculateArea()` that returns a `double`, `calculateSideC()` that returns a `double`.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu16
//Triangle.java

public class Triangle{
    private double sideA;
    private double sideB;
    private double angleC;

    //getters
    public double getSideA(){
        return sideA;
    }//end of getSideA

    public double getSideB(){
        return sideB;
    }//end of getSideB

    public double getAngleC(){
        return angleC;
    }//end of getAngleC

    //setters
    public void setSideA(double sideA){
        if (sideA < 0) sideA = 0;
        this.sideA = sideA;
    }//end of setSideA()

    public void setSideB(double sideB){
        if (sideB < 0) sideB = 0;
        this.sideB = sideB;
    }//end of setSideB()

    public void setAngleC(double angleC){
        if (angleC < 0) angleC = 0;
        if (angleC > 180) angleC = 180;
        this.angleC = angleC;
    }//end of setAngleC()

    //constructor
    public Triangle(double sideA, double sideB, double angleC){
        setSideA(sideA);
        setSideB(sideB);
        setAngleC(angleC);
    }//end of constructor

    //no-args constructor
    public Triangle(){
        this(1.0, 1.0, 60.0);
    }//end of no-args constructor

    public double calculateArea(){
        return 0.5 * sideA * sideB * Math.sin(angleC * Math.PI / 180);
    }//end of calculateArea()

    public double calculateSideC(){
        return Math.sqrt(sideA * sideA + sideB * sideB -
               2 * sideA * sideB * Math.cos(angleC * Math.PI / 180));
    }//end of calculateSideC()

    public static void main(String[] args){
        Triangle tri1 = new Triangle(3.0, 4.0, 90.0);
        Triangle tri2 = new Triangle(5.0, 5.0, 60.0);
        Triangle tri3 = new Triangle(5.9, 2.4, 156.7);

        System.out.printf("Area: %.2f\tSide C: %.2f\n",
                tri1.calculateArea(), tri1.calculateSideC());
        System.out.printf("Area: %.2f\tSide C: %.2f\n",
                tri2.calculateArea(), tri2.calculateSideC());
        System.out.printf("Area: %.2f\tSide C: %.2f\n",
                tri3.calculateArea(), tri3.calculateSideC());
    }//end of main()
}//end of class Triangle
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu17. Create a `Parallelogram` class that has three `double` fields: `side1`, `side2`, `angle`, where `side1` and `side2` are adjacent sides and `angle` is the angle between them. It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the three fields.

The setter for `angle` should ensure that its value is greater than 0° and less than 180°. The setter for the sides should ensure that they are not negative.

There should be two methods: `calculateArea()` that returns a `double`, `calculateDiagonal()` that returns a `double`. The latter finds the length of the diagonal through the vertex at `angle`. [The area of a parallelogram is given by `side1 * side2 * sin(angle)`.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu17
//Parallelogram.java

public class Parallelogram{
    private double side1;
    private double side2;
    private double angle;

    //getters
    public double getSide1(){
        return side1;
    }//end of getSide1()

    public double getSide2(){
        return side2;
    }//end of getSide2()

    public double getAngle(){
        return angle;
    }//end of getAngle()

    //setters
    public void setSide1(double side1){
        if (side1 < 0) side1 = 0;
        this.side1 = side1;
    }//end of setSide1()

    public void setSide2(double side2){
        if (side2 < 0) side2 = 0;
        this.side2 = side2;
    }//end of setSide2()
```

```java
    public void setAngle(double angle){
        if (angle < 0) angle = 0;
        if (angle > 180) angle = 180;
        this.angle = angle;
    }//end of setAngle()

    //constructor
    public Parallelogram(double side1, double side2, double angle){
        setSide1(side1);
        setSide2(side2);
        setAngle(angle);
    }//end of constructor

    //no-args constructor
    public Parallelogram(){
        this(1.0, 1.0, 90.0);
    }//end of no-args constructor

    public double calculateArea(){
        return 0.5 * side1 * side2 * Math.sin(angle * Math.PI / 180);
    }//end of calculateArea()

    public double calculateDiagonal(){
        return Math.sqrt(side1 * side1 + side2 * side2 +
                2 * side1 * side2 * Math.cos(angle * Math.PI / 180));
    }//end of calculateDiagonal()

    public static void main(String[] args){
        Parallelogram par1 = new Parallelogram(5.0, 4.0, 45.0);
        Parallelogram par2 = new Parallelogram(5.0, 12.0, 90.0);
        Parallelogram par3 = new Parallelogram(10.0, 10.0, 120.0);

        System.out.printf("Area: %.3f\tDiagonal: %.3f\n",
                par1.calculateArea(), par1.calculateDiagonal());
        System.out.printf("Area: %.3f\tDiagonal: %.3f\n",
                par2.calculateArea(), par2.calculateDiagonal());
        System.out.printf("Area: %.3f\tDiagonal: %.3f\n",
                par3.calculateArea(), par3.calculateDiagonal());
    }//end of main()
}//end of class Parallelogram
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu18. Track suits are available in three sizes: small ('S'), medium ('M') and large ('L'). They may also be white or coloured. The white track suits are priced at ₦5000, ₦5500 and ₦6000 each respectively. The coloured track suits cost 20% more. Create a `TrackSuit` class having a `char` field `size`, and a `boolean` field `isColoured`. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields.

There should be one method that returns the price of the suit.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its method.

```
//Qu18
//TrackSuit.java

public class TrackSuit{
    private char size;
    private boolean isColoured;

    //getters
    public char getSize(){
        return size;
    }//end of getSize()

    public boolean getIsColoured(){
        return isColoured;
    }//end of getIsColoured()

    //setters
    public void setSize(char size){
        this.size = size;
    }//end of setSize()

    public void setIsColoured(boolean isColoured){
        this.isColoured = isColoured;
    }//end of setIsColoured

    //constructor
    public TrackSuit(char size, boolean isColoured){
        setSize(size);
        setIsColoured(isColoured);
    }//end of constructor

    //no-args constructor
    public TrackSuit(){
        this('S', false);
    }//end of no-args constructor

    public double getPrice(){
        double price = 0;
        switch (size){
            case 'S':
                price = 5000.0;
                break;
            case 'M':
                price = 5500.0;
                break;
            case 'L':
                price = 6000.0;
                break;
        }//end of switch

        if (isColoured) price = 1.20 * price;

        return price;
    }//end of getPrice()

    public static void main(String[] args){
        TrackSuit tSuit1 = new TrackSuit('M', true);
        TrackSuit tSuit2 = new TrackSuit('L', false);
        TrackSuit tSuit3 = new TrackSuit();

        tSuit3.setSize('S');
        tSuit3.setIsColoured(true);

        System.out.printf("The price is %.2f\n", tSuit1.getPrice());
        System.out.printf("The price is %.2f\n", tSuit2.getPrice());
        System.out.printf("The price is %.2f\n", tSuit3.getPrice());
    }//end of main()
}//end of class TrackSuit
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu19. A hemisphere sits on top of an inverted cone. The base radius of the cone is the same as the radius of the hemisphere. Create a `Solid` class that will describe this object. It will have fields `radius` (the common radius just described) and `height` (the height of the cone). Both should be `double`. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields. The setters should ensure that the dimensions are not negative.

There should be two methods that return `doubles`. One should return the *volume* of the object and the other return its *surface area*. [The volume of a sphere is given by $\frac{4}{3}$® r^3 . The surface area of a sppere is $4\pi r^2$. The volume of a cone is $\frac{1}{3}\pi r^2 h$. The surface area of the curved surface of a cone is given by πrl, where l is the slant height.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu19
//Solid.class

public class Solid{
   private double radius;  //the common radius of the hemisphere and the cone
    private double height;  //the height of the cone

    //getters
    public double getRadius(){
        return radius;
    }//end of getRadius()

    public double getHeight(){
        return height;
    }//end of getHeight()

    //setters
    public void setRadius(double radius){
        if (radius < 0) radius = 0;
        this.radius = radius;
    }//end of setRadius()

    public void setHeight(double height){
        if (height < 0) height = 0;
        this.height = height;
    }//end of setHeight()

    //constructor
    public Solid(double radius, double height){
        setRadius(radius);
        setHeight(height);
    }//end of constructor

    //no-args constructor
    public Solid(){
        this(0.0, 0.0);
    }//end of no-args constructor

    public double getVolume(){
        return (2.0 / 3.0) * Math.PI * radius * radius * radius +
               (1.0 / 3.0) * Math.PI * radius * radius * height;
    }//end of getVolume()

    public double getSurfaceArea(){
        return 2 * Math.PI * radius * radius +
                2 * Math.PI * radius * Math.sqrt(height * height + radius *
radius);
    }//end of getSurfaceArea()

    public static void main(String[] args){
        Solid solid1 = new Solid(2.0, 3.0);
        Solid solid2 = new Solid(10.5, 4.7);
        Solid solid3 = new Solid(1.0, 1.0);

        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                          solid1.getVolume(), solid1.getSurfaceArea());
        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                          solid2.getVolume(), solid2.getSurfaceArea());
        System.out.printf("Volume: %.2f\tSurface area: %.2f\n",
                          solid3.getVolume(), solid3.getSurfaceArea());
    }//end of main()
}//end of class Solid
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu20. Create a `Triangle` class that has three `double` fields: `sideA`, `sideB`, `angleC`. It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the three fields.

The setters should ensure that the two sides have lengths that are not negative and that the angle lies between 0° and 180°.

There should be two methods: `calculateArea()` that returns a `double`, `calculateSideC()` that returns a `double`.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu16
//Triangle.java

public class Triangle{
    private double sideA;
    private double sideB;
    private double angleC;

    //getters
    public double getSideA(){
        return sideA;
    }//end of getSideA

    public double getSideB(){
        return sideB;
    }//end of getSideB

    public double getAngleC(){
        return angleC;
    }//end of getAngleC

    //setters
    public void setSideA(double sideA){
        if (sideA < 0) sideA = 0;
        this.sideA = sideA;
    }//end of setSideA()

    public void setSideB(double sideB){
        if (sideB < 0) sideB = 0;
        this.sideB = sideB;
    }//end of setSideB()

    public void setAngleC(double angleC){
        if (angleC < 0) angleC = 0;
        if (angleC > 180) angleC = 180;
        this.angleC = angleC;
    }//end of setAngleC()

    //constructor
    public Triangle(double sideA, double sideB, double angleC){
        setSideA(sideA);
        setSideB(sideB);
        setAngleC(angleC);
    }//end of constructor

    //no-args constructor
    public Triangle(){
        this(1.0, 1.0, 60.0);
    }//end of no-args constructor

    public double calculateArea(){
        return 0.5 * sideA * sideB * Math.sin(angleC * Math.PI / 180);
    }//end of calculateArea()

    public double calculateSideC(){
        return Math.sqrt(sideA * sideA + sideB * sideB -
               2 * sideA * sideB * Math.cos(angleC * Math.PI / 180));
    }//end of calculateSideC()

    public static void main(String[] args){
        Triangle tri1 = new Triangle(3.0, 4.0, 90.0);
        Triangle tri2 = new Triangle(5.0, 5.0, 60.0);
        Triangle tri3 = new Triangle(5.9, 2.4, 156.7);

        System.out.printf("Area: %.2f\tSide C: %.2f\n",
                tri1.calculateArea(), tri1.calculateSideC());
        System.out.printf("Area: %.2f\tSide C: %.2f\n",
                tri2.calculateArea(), tri2.calculateSideC());
        System.out.printf("Area: %.2f\tSide C: %.2f\n",
                tri3.calculateArea(), tri3.calculateSideC());
    }//end of main()
}//end of class Triangle
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu21. Create a `Patient` class that could be used for the management of hospital patients. It should have three fields: `name` (`String`), `height` (`double`) and `weight` (`double`). It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the three fields.

The setters should ensure that the numerical data are not negative.

There should be two methods: `getBMI()` that returns a `double`, `isObese()` that returns a `boolean`. The first returns the BMI (body mass index) of a patient where *BMI* = `weight`/`height`^{2}, the weight being in kilograms and the height in metres. The second returns `true` if the BMI is greater than or equal to 25.0 and otherwise `false`.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu21
//Patient.java

public class Patient{
    private String name;
    private double height;
    private double weight;

    //getters
    public String getName(){
        return name;
    }//end of getName()

    public double getHeight(){
        return height;
    }//end of getHeight()

    public double getWeight(){
        return weight;
    }//end of getWeight()

    //setters
    public void setName(String name){
        this.name = name;
    }//end of setName()

    public void setHeight(double height){
        if (height < 0.0) height = 0.0;
        this.height = height;
    }//end of setHeight()

    public void setWeight(double weight){
        if (weight < 0.0) weight = 0.0;
        this.weight = weight;
    }//end of setWeight()

    //constructor
    public Patient(double height, double weight){
        setHeight(height);
        setWeight(weight);
    }//end of constructor

    //no-args constructor
    public Patient(){
        this(0.0, 0.0);
    }//end of no-args constructor

    public double getBMI(){
        return weight / (height * height);
    }//end of getBMI()

    public boolean isObese(){
        if (getBMI() >= 25.0) return true;
        return false;
    }//end of isObese()

    public static void main(String[] args){
        Patient patient1 = new Patient(1.85, 79.0);
        Patient patient2 = new Patient(1.56, 63.4);
        Patient patient3 = new Patient(1.20, 55.8);

        System.out.printf("BMI: %.1f\n", patient1.getBMI());
        if (patient1.isObese()) System.out.println("Patient is obese");
        else System.out.println("Patient is not obese");

        System.out.printf("BMI: %.1f\n", patient2.getBMI());
        if (patient2.isObese()) System.out.println("Patient is obese");
        else System.out.println("Patient is not obese");

        System.out.printf("BMI: %.1f\n", patient3.getBMI());
        if (patient3.isObese()) System.out.println("Patient is obese");
        else System.out.println("Patient is not obese");
    }//end of main()
}//end of class Patient
```

Qu22. Create a Car class that has two String fields – make and model, (eg ‘Honda’ and ‘Accord’) – and one double field – performance, which is the time in seconds for the car to accelerate from zero to 100 kph. It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the three fields.
The setter should ensure that the `performance` is at least 5.

There should be two methods: `getMeanAcceleration()` that returns a `double`, and `getDistance()` that also returns a `double`. The first returns the acceleration of the vehicle when it moves from 0 to 100 km/h, given by 250/9*p* m/s^2. The second returns the distance travelled, given by 125*p*/9. In each case *p* is the `performance`.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods. For many cars you can take `performance` to be around 10 seconds.

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu22. Create a `Car` class that has two `String` fields – `make` and `model`, (eg ‘Honda’ and ‘Accord’) – and one `double` field – `performance`, which is the time in seconds for the car to accelerate from zero to 100 kph. It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the three fields.

The setter should ensure that the `performance` is at least 5.

There should be two methods: `getMeanAcceleration()` that returns a `double`, and `getDistance()` that also returns a `double`. The first returns the acceleration of the vehicle when it moves from 0 to 100 km/h, given by 250/9*p* m/s^2. The second returns the distance travelled, given by 125*p*/9. In each case *p* is the `performance`.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods. For many cars you can take `performance` to be around 10 seconds.

```
//Qu22
//Car.java

public class Car{
    private String make;
    private String model;
    private double performance;

    //getters
    public String getMake(){
        return make;
    }//end of getMake()

    public String getModel(){
        return model;
    }//end of getModel()

    public double getPerformance(){
        return performance;
    }//end of getPerformance()

    //setters
    public void setMake(String make){
        this.make = make;
    }//end of setMake()

    public void setModel(String model){
        this.model = model;
    }//end of setModel()
    public void setPerformance(double performance){
        if (performance < 5.0) performance = 5.0;
        this.performance = performance;
    }//end of setPerformance()

    //constructor
    public Car(String make, String model, double performance){
        setMake(make);
        setModel(model);
        setPerformance(performance);
    }//end of constructor

    //no-args constructor
    public Car(){
        this("Unknown make", "Unknown model", 5.0);
    }//end of no-args constructor

    public double getMeanAcceleration(){
        return 250 / (9 * performance);
    }//end of getMeanAcceleration

    public double getDistance(){
        return 125 * performance / 9;
    }//end of getDistance()

    public static void main(String[] args){
        Car car1 = new Car("Honda", "Accord", 10.0);
        Car car2 = new Car("Toyota", "Avensis", 9.0);
        Car car3 = new Car("Peugeot", "720", 10.5);

        System.out.printf("Acceleration: %.2f m/s2\tDistance: %.1f m\n",
                car1.getMeanAcceleration(), car1.getDistance());
        System.out.printf("Acceleration: %.2f m/s2\tDistance: %.1f m\n",
                car2.getMeanAcceleration(), car2.getDistance());
        System.out.printf("Acceleration: %.2f m/s2\tDistance: %.1f m\n",
                car3.getMeanAcceleration(), car3.getDistance());
    }//end of main()
}//end of class Car
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu23. Create an `Purchase` class that is used to describe a purchase made at a store. It should have one `double` field: `unitPrice`, being the price of a single item. It should have one `int` field: `quantity`, being the number of items purchased. It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields.

The setters should ensure that the two fields are not negative.

There should be one method: `calculateCost()` that returns a `double`, being the cost of a purchase. The cost is calculated based upon the store owner offer of a 10% discount if the quantity purchased is equal to or more than 100.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu23
//Purchase.java

public class Purchase{
    private double unitPrice;
    private int quantity;

    //getters
    public double getUnitPrice(){
        return unitPrice;
    }//end of getUnitPrice()

    public int getQuantity(){
        return quantity;
    }//end of getQuantity()

    //setters
    public void setUnitPrice(double unitPrice){
        if (unitPrice < 0) unitPrice = 0;
        this.unitPrice = unitPrice;
    }//end of setUnitPrice()

    public void setQuantity(int quantity){
        if (quantity < 0) quantity = 0;
        this.quantity = quantity;
    }//end of setQuantity()

    //constructor
    public Purchase(double unitPrice, int quantity){
        setUnitPrice(unitPrice);
        setQuantity(quantity);
    }//end of constructor

    //no-args constructor
    public Purchase(){
        this(0.0, 0);
    }//end of no-args constructor

    public double calculateCost(){
        double cost = unitPrice * quantity;

        if (quantity >= 100)
            cost -= cost * 0.1;  //10% discount

        return cost;
    }//end of calculateCost()

    public static void main(String[] args){
        Purchase purchase1 = new Purchase(50.0, 80);
        Purchase purchase2 = new Purchase(60.0, 100);
        Purchase purchase3 = new Purchase(70.0, 150);

        System.out.printf("The cost of %3d @ %8.2f is %10.2f\n",
                           purchase1.quantity, purchase1.unitPrice,
purchase1.calculateCost());
        System.out.printf("The cost of %3d @ %8.2f is %10.2f\n",
                           purchase2.quantity, purchase2.unitPrice,
purchase2.calculateCost());
        System.out.printf("The cost of %3d @ %8.2f is %10.2f\n",
                           purchase3.quantity, purchase3.unitPrice,
purchase3.calculateCost());
    }//end of main()
}//end of class Purchase
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu24. Create a `Conversion` class that can be used to convert one unit into another by multiplying by a scale factor. For example a length expressed in inches can be converted to centimetres by multiplying it by 2.54. The class should have four fields: the magnitude to be converted (`double`), the name of the source unit (`String`), the name of the unit to be converted to (`String`) and the scale factor (`double`). It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the four fields.

There should be one method: `convert()` that returns a `double` – the result of the conversion.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods. The output should be something like “`10 in is equal to 25.4 cm`”.

Here are some scale factors that you might use: 1 in = **2.54** cm; 1 kg = **2.2045** lb; 1 h = **3600** s; 1 m = **3.2808** ft.

```
//Qu24
//Conversion.java

public class Conversion{
    private double magnitude;
    private String fromUnit;
    private String toUnit;
    private double scaleFactor;

    //getters
    public double getMagnitude(){
        return magnitude;
    }//end of getMagnitude()

    public String getFromUnit(){
        return fromUnit;
    }//end of getFromUnit()

    public String getToUnit(){
        return toUnit;
    }//end of getToUnit()

    public double getScaleFactor(){
        return scaleFactor;
    }//end of getScaleFactor()

    //setters
    public void setMagnitude(double magnitude){
        this.magnitude = magnitude;
    }//end of setMagnitude()
    public void setFromUnit(String fromUnit){
        this.fromUnit = fromUnit;
    }//end of setFromUnit()

    public void setToUnit(String toUnit){
        this.toUnit = toUnit;
    }//end of setToUnit()

    public void setScaleFactor(double scaleFactor){
        this.scaleFactor = scaleFactor;
    }//end of setScaleFactor()

    //constructor
    public Conversion(double magnitude, String fromUnit, String toUnit, double
scaleFactor){
        setMagnitude(magnitude);
        setFromUnit(fromUnit);
        setToUnit(toUnit);
        setScaleFactor(scaleFactor);
    }//end of constructor

    //no-args constructor
    public Conversion(){
        this(0.0, "noname", "noname", 0.0);
    }//end of no-args constructor

    public double convert(){
        return magnitude * scaleFactor;
    }//end of convert()

    public static void main(String[] args){
        Conversion conversion1 = new Conversion(36.0, "in", "cm", 2.54);
        Conversion conversion2 = new Conversion(80, "kg", "lb", 2.204622622);
        Conversion conversion3 = new Conversion(24, "h", "s", 3600.0);

        System.out.printf("%f %s is equivalent to %f %s\n",
                conversion1.magnitude, conversion1.fromUnit,
                conversion1.convert(), conversion1.toUnit);
        System.out.printf("%f %s is equivalent to %f %s\n",
                conversion2.magnitude, conversion2.fromUnit,
                conversion2.convert(), conversion2.toUnit);
        System.out.printf("%f %s is equivalent to %f %s\n",
                conversion3.magnitude, conversion3.fromUnit,
                conversion3.convert(), conversion3.toUnit);
    }//end of main()
}//end of class Conversion
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu25. Create a `RegularPolygon` class that has two fields: `n` and `x`. The first, an `int`, is the number of sides; the second, a `double`, is the length of each side. The class should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields.

The setters should ensure that the two fields are not negative and that the number of sides is greater than two.

There should be two methods: `calculateArea()` that returns a `double`, `calculatePerimeter()` that also returns a `double`. [The area of a regular *n*-sided polygon of side *x* is $\frac{n}{4}x^2\cot\left(\frac{180^{\circ}}{n}\right)$.]

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu25
//RegularPolygon.java

public class RegularPolygon{
    private int n;          //number of sides
    private double x;       //length of a side

    //getters
    public int getN(){
        return n;
    }//end of getN()

    public double getX(){
        return x;
    }//end of getX()

    //setters
    public void setN(int n){
        if (n < 3) n = 3;
        this.n = n;
    }//end of setN()

    public void setX(double x){
        if (x < 0) x = 0;
        this.x = x;
    }//end of setX()

    //constructor
    public RegularPolygon(int n, double x){
        setN(n);
        setX(x);
    }//end of constructor

    //no-args constructor
    public RegularPolygon(){
        this(3, 0.0);
    }//end of no-args constructor

    public double calculateArea(){
        return 0.25 * n * x * x / Math.tan(Math.PI/n);
    }//end of calculateArea()

    public double calculatePerimeter(){
        return n * x;
    }//end of calculatePerimeter()

    public static void main(String[] args){
        RegularPolygon regPoly1 = new RegularPolygon(6, 2.5);
        RegularPolygon regPoly2 = new RegularPolygon(4, 10.0);
        RegularPolygon regPoly3 = new RegularPolygon(100, 1.0);

        System.out.printf("Area: %.6f\tPerimeter: %.2f\n",
                regPoly1.calculateArea(), regPoly1.calculatePerimeter());
        System.out.printf("Area: %.6f\tPerimeter: %.2f\n",
                regPoly2.calculateArea(), regPoly2.calculatePerimeter());
        System.out.printf("Area: %.6f\tPerimeter: %.2f\n",
                regPoly3.calculateArea(), regPoly3.calculatePerimeter());
    }//end of main()
}//end of class RegularPolygon
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu26. Create a `Ballistic` class that describes the motion of a projectile. It should have two `double` fields: `speed`, and `elevation`. It should have the necessary getters and setters, and two constructors – a no-args constructor and one that sets the values of the two fields.

The setters should ensure that `speed` is not negative and that the `elevation` lies between 0° and 90°.

The method `getRange()` returns a `double` given by (V^2/g) sin2¯; the method `getMaxHeight()` returns a `double` given by ($V^2/2g$) $\sin^2$¯, where V is `speed`, ¯ is `elevation` and $g = 9.8$ m/s^2.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu26
//Ballistic.java

public class Ballistic{
    private double speed;
    private double elevation;

    //getters
    public double getSpeed(){
        return speed;
    }//end of getSpeed()

    public double getElevation(){
        return elevation;
    }//end of getElevation()

    //setters
    public void setSpeed(double speed){
        if (speed < 0) speed = 0;
        this.speed = speed;
    }//end of setSpeed()

    public void setElevation(double elevation){
        if (elevation < 0) elevation = 0;
        if (elevation > 90) elevation = 90;
        this.elevation = elevation;
    }//end of setElevation()

    //constructor
    public Ballistic(double speed, double elevation){
        setSpeed(speed);
        setElevation(elevation);
    }//end of constructor
    //no-args constructor
    public Ballistic(){
        this(0.0, 0.0);
    }//end of no-args constructor

    public double getRange(){
        double g = 9.8;
        return (speed * speed / g) * Math.sin(2 * elevation * Math.PI / 180);
    }//end of getRange()

    public double getMaxHeight(){
        double g = 9.8;
         return (0.5 * speed * speed / g) * Math.pow(Math.sin(elevation *
Math.PI / 180), 2.0);
    }//end of getMaxHeight()

    public static void main(String[] args){
        Ballistic ballistic1 = new Ballistic(100.0, 45.0);
        Ballistic ballistic2 = new Ballistic(100.0, 60.0);
        Ballistic ballistic3 = new Ballistic(100.0, 30.0);

        System.out.printf("Range: %.1f\tHeight: %.1f\n",
                ballistic1.getRange(), ballistic1.getMaxHeight());
        System.out.printf("Range: %.1f\tHeight: %.1f\n",
                ballistic2.getRange(), ballistic2.getMaxHeight());
        System.out.printf("Range: %.1f\tHeight: %.1f\n",
                ballistic3.getRange(), ballistic3.getMaxHeight());
    }//end of main()
}//end of class Ballistic
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu27. Create a `LeaguePoints` class that has three `int` fields: `won`, `lost`, `drawn`. It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the three fields.

The setters should ensure that the three fields are not negative.

There should be one method: `calculatePoints()` that returns an `int`. The team earns three points for each win and one point for each draw.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu27
//LeaguePoints.java

public class LeaguePoints{
    private int won;
    private int lost;
    private int drawn;

    //getters
    public int getWon(){
        return won;
    }//end of getWon()

    public int getLost(){
        return lost;
    }//end of getLost()

    public int getDrawn(){
        return drawn;
    }//end of getDrawn()

    //setters
    public void setWon(int won){
        if (won < 0) won = 0;
        this.won = won;
    }//end of setWon()

    public void setLost(int lost){
        if (lost < 0) lost = 0;
        this.lost = lost;
    }//end of setLost()

    public void setDrawn(int drawn){
        if (drawn < 0) drawn = 0;
        this.drawn = drawn;
    }//end of setDrawn()

    //constructor
    public LeaguePoints(int won, int lost, int drawn){
        setWon(won);
        setLost(lost);
        setDrawn(drawn);
    }//end of constructor

    //no-args constructor
    public LeaguePoints(){
        this(0, 0, 0);
    }//end of no-args constructor

    public int calculatePoints(){
        return 3 * won + drawn;
    }//end of calculatePoints()

    public static void main(String[] args){
        LeaguePoints points1 = new LeaguePoints(18, 0, 2);
        LeaguePoints points2 = new LeaguePoints(10, 5, 5);
        LeaguePoints points3 = new LeaguePoints(4, 8, 8);

        System.out.printf("Won: %2d\tLost: %2d\tDrawn: %2d\tPoints: %3d\n",
                           points1.won, points1.lost, points1.drawn,
points1.calculatePoints());
        System.out.printf("Won: %2d\tLost: %2d\tDrawn: %2d\tPoints: %3d\n",
                           points2.won, points2.lost, points2.drawn,
points2.calculatePoints());
        System.out.printf("Won: %2d\tLost: %2d\tDrawn: %2d\tPoints: %3d\n",
                           points3.won, points3.lost, points3.drawn,
points3.calculatePoints());
    }//end of main()
}//end of class LeaguePoints
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu28. Create a `Account` class that has three fields: `accountNo` (`int`), `name` (`String`) and `balance` (`double`). It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the three fields.

There should be two methods `makeDeposit()`and `makeWithdrawal()` both returning `void`. The first takes a double parameter – the amount to be deposited in the account. The second also takes a double parameter – the amount to be withdrawn from the account. There are no limits to the amounts that can be withdrawn or deposited but they cannot be negative.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu28
//Account.java

public class Account{
    private int accountNo;
    private String name;
    private double balance;

    //getters
    public int getAccountNo(){
        return accountNo;
    }//end of getAccountNo()

    public String getName(){
        return name;
    }//end of getName()

    public double getBalance(){
        return balance;
    }//end of getBalance()

    //setters
    public void setAccountNo(int accountNo){
        this.accountNo = accountNo;
    }//end of setAccountNo()

    public void setName(String name){
        this.name = name;
    }//end of setName()

    public void setBalance(double balance){
        this.balance = balance;
    }//end of setBalance()
    //constructor
    public Account(int accountNo, String name, double balance){
        setAccountNo(accountNo);
        setName(name);
        setBalance(balance);
    }//end of constructor

    //no-args constructor
    public Account(){
        this(0, "noname", 0.0);
    }//end of no-args constructor

    public void makeDeposit(double amount){
        balance += amount;
    }//end of makeDeposit()

    public void makeWithdrawal(double amount){
        balance -= amount;
    }//end of makeWithdrawal

    public static void main(String[] args){
        Account account1 = new Account(1, "Aliyu", 2000.0);
        Account account2 = new Account(2, "Hauwa", 4000.0);
        Account account3 = new Account(3, "Gideon", 6000.0);

        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account1.getAccountNo(), account1.getBalance());
        account1.makeDeposit(1000.0);
        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account1.getAccountNo(), account1.getBalance());
        account1.makeWithdrawal(2000.0);
        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account1.getAccountNo(), account1.getBalance());


        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account2.getAccountNo(), account2.getBalance());
        account2.makeDeposit(1000.0);
        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account2.getAccountNo(), account2.getBalance());
        account2.makeWithdrawal(5000.0);
        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account2.getAccountNo(), account2.getBalance());

        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account3.getAccountNo(), account3.getBalance());
        account3.makeDeposit(3000.0);
        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account3.getAccountNo(), account3.getBalance());
        account3.makeWithdrawal(10000.0);
        System.out.printf("A/c No: %03d\tBalance: %10.2f\n",
                account3.getAccountNo(), account3.getBalance());
    }//end of main()
}//end of class Account
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu29. Create a `Rectangle` class that has two `double` fields: `sideA` and `sideB`. It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the two fields.

The setters should ensure that the two sides have lengths that are not negative.

There should be three methods: `calculateArea()` that returns a `double`, `calculatePerimeter()` that returns a `double` and `calculateDiagonal()` that also returns a `double`.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu29
//Rectangle.java

public class Rectangle{
    private double sideA;
    private double sideB;

    //getters
    public double getSideA(){
        return sideA;
    }//end of getSideA()

    public double getSideB(){
        return sideB;
    }//end of getSideB

    //setters
    public void setSideA(double sideA){
        if (sideA < 0) sideA = 0;
        this.sideA = sideA;
    }//end of setSideA()

    public void setSideB(double sideB){
        if (sideB < 0) sideB = 0;
        this.sideB = sideB;
    }//end of setSideB()

    //constructor
    public Rectangle(double sideA, double sideB){
        setSideA(sideA);
        setSideB(sideB);
    }//end of constructor

    //no-args constructor
    public Rectangle(){
        this(0.0, 0.0);
    }//end of no-args constructor

    public double calculateArea(){
        return sideA * sideB;
    }//end of calculateArea()

    public double calculatePerimeter(){
        return 2 * (sideA + sideB);
    }//end of calculatePerimeter()

    public double calculateDiagonal(){
        return Math.sqrt(sideA * sideA + sideB * sideB);
    }//end of calculateDiagonal()

    public static void main(String[] args){
        Rectangle rect1 = new Rectangle(2.5, 5.6);
        Rectangle rect2 = new Rectangle(5.0, 12.0);
        Rectangle rect3 = new Rectangle(4.5, 10.6);

        System.out.printf("Area: %.2f\tPerimeter: %.2f\tDiagonal: %.2f\n",
                        rect1.calculateArea(),  rect1.calculatePerimeter(),
rect1.calculateDiagonal());
        System.out.printf("Area: %.2f\tPerimeter: %.2f\tDiagonal: %.2f\n",
                        rect2.calculateArea(),  rect2.calculatePerimeter(),
rect2.calculateDiagonal());
        System.out.printf("Area: %.2f\tPerimeter: %.2f\tDiagonal: %.2f\n",
                        rect3.calculateArea(),  rect3.calculatePerimeter(),
rect3.calculateDiagonal());
    }//end of main()
}//end of class rectangle
```

Ahmadu Bello University
Department of Mathematics
COSC211: Practical Test: June 2014

Instructions

A. Answer the single question below. Time allowed: 90 min.

B. Your solution should be submitted in electronic form as Java source files. They should be stored in a directory whose name is your matriculation number.

C. At the beginning of every file that you submit should be the following identifying marks.

```
//Your matriculation number
//Your full name
//The number of your practical group
//The number of the question assigned to you (shown below)
//The name you have given your file
... your code follows here ...
```

Qu30. Create a `RightTriangle` class that has two `double` fields: `sideA` and `sideB`, `being the sides that include the right-angle`. It should have the necessary getters and setters, and two constructors - a no-args constructor and one that sets the values of the three fields.

The setters should ensure that the two sides have lengths that are not negative.

There should be two methods: `calculateHypotenuse()` that returns a `double`, `calculatePerimeter()` that also returns a `double`.

Write test code that will instantiate at least three objects from this class and demonstrate the use of its methods.

```
//Qu30
//RightTriangle.java

public class RightTriangle{
    private double sideA;
    private double sideB;

    //getters
    public double getSideA(){
        return sideA;
    }//end of getSideA()

    public double getSideB(){
        return sideB;
    }//end of getSideB

    //setters
    public void setSideA(double sideA){
        if (sideA < 0) sideA = 0;
        this.sideA = sideA;
    }//end of setSideA()

    public void setSideB(double sideB){
        if (sideB < 0) sideB = 0;
        this.sideB = sideB;
    }//end of setSideB()

    //constructor
    public RightTriangle(double sideA, double sideB){
        setSideA(sideA);
        setSideB(sideB);
    }//end of constructor
    //no-args constructor
    public RightTriangle(){
        this(0.0, 0.0);
    }//end of no-args constructor

    public double calculateHypotenuse(){
        return Math.sqrt(sideA * sideA + sideB * sideB);
    }//end of calculateHypotenuse();

    public double calculatePerimeter(){
        return sideA + sideB + calculateHypotenuse();
    }//end calculatePerimeter()

    public static void main(String[] args){
        RightTriangle triangle1 = new RightTriangle(5.6, 7.9);
        RightTriangle triangle2 = new RightTriangle(4.0, 3.0);
        RightTriangle triangle3 = new RightTriangle(5.0, 12.0);

        System.out.printf("Hypotenuse: %.2f\tPerimeter: %.2f\n",
                triangle1.calculateHypotenuse(),
                triangle1.calculatePerimeter());
        System.out.printf("Hypotenuse: %.2f\tPerimeter: %.2f\n",
                triangle2.calculateHypotenuse(),
                triangle2.calculatePerimeter());
        System.out.printf("Hypotenuse: %.2f\tPerimeter: %.2f\n",
                triangle3.calculateHypotenuse(),
                triangle3.calculatePerimeter());
    }//end of main()
}//end of class RightTriangle
```