1. (Rectangle Class) Create a class Rectangle with attributes length and width. Provide member functions that calculate the perimeter and the area of the rectangle.

   Program:

Rectangle.java

```java
public class Rectangle{
    private double length;
    private double width;

    public Rectangle(double length, double width){
        this.length = length;
        this.width = width;
    }

    public double getPerimeter(){
        return 2 * (length + width);
    }

    public double getArea(){
        return length * width;
    }

    public static void main(String[] args){
        Rectangle rect = new Rectangle(5.0, 3.0);
        System.out.println("Perimeter: " + rect.getPerimeter());
        System.out.println("Area: " + rect.getArea());
    }
}

```

   Output:

```
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> javac .\Rectangle.java
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> java Rectangle
Perimeter: 16.0
Area: 15.0
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP>
```

2. (Invoice Class) Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four pieces of information as data members a part number (type string), a part description (type string), a quantity of the item being purchased (type int) and a price per item (type int). Your class

should have a constructor that initializes the four data members. In addition, provide a member function named getInvoiceAmount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as an int value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0. Write a test program that demonstrates class Invoice's capabilities.

Program:

Invoice.java

```java
public class Invoice{
    private String partNumber;
    private String partDescription;
    private int quantity;
    private int pricePerItem;

    public Invoice(String partNumber, String partDescription, int quantity, int pricePerItem){
        this.partNumber = partNumber;
        this.partDescription = partDescription;
        if(quantity >= 0){
            this.quantity = quantity;
        }else{
            this.quantity = 0;
        }
        if(pricePerItem >= 0){
            this.pricePerItem = pricePerItem;
        }else{
            this.pricePerItem = 0;
        }
    }

    public int getInvoiceAmount(){
        return quantity * pricePerItem;
    }

    public static void main(String[] args){
        Invoice inv1 = new Invoice("12", "ABDI", 5, 500);
        Invoice inv2 = new Invoice("13", "SAID", 3, 300);

        System.out.println("\nobject1_Invoice Amount: " + inv1.getInvoiceAmount());
        System.out.println("object2_Invoice Amount: " + inv2.getInvoiceAmount()+"\n");
    }
}

```

Output:

```
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> JAVAC .\Invoice.java
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> JAVA Invoice

object1_Invoice Amount: 2500
object2_Invoice Amount: 900

PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP>
```

3. (Account Class) Create a class called Account that a bank might use to represent customers' bank accounts. Your class should include one data member of type int to represent the account balance. Your class should provide a constructor that receives an initial balance and uses it to initialize the data member. The constructor should validate the initial balance to ensure that it is greater than or equal to 0. If not, the balance should be set to 0 and the constructor should display an error message, indicating that the initial balance was invalid. The class should provide three member functions. Member function credit should add an amount to the current balance. Member function debit should withdraw money from the Account and should ensure that the debit amount does not exceed the Account's balance. If it does, the balance should be left unchanged and the function should print a message indicating "Debit amount exceeded account balance." Member function getBalance should return the current balance. Create a program that creates two Account objects and tests the member functions of class Account.

   Program:

Account.java

```java
public class Account{
    private int accountBalance;

    public Account(int initialBalance){
        if(initialBalance >= 0){
            accountBalance = initialBalance;
        }else{
            accountBalance = 0;
            System.out.println("Initial Balance is invalid!");
        }
    }

    public void credit(int amount){
        accountBalance += amount;
    }

    public void debit(int amount){
        if(amount > accountBalance){
            System.out.println("Debit amount exceeded account accountBalance!");
        }else{
            accountBalance -= amount;
        }
    }

    public int getBalance(){
        return accountBalance;
    }

    public static void main(String[] args){
        Account a1 = new Account(1000);
        Account a2 = new Account(-500);

        System.out.println("\nFor object 1:");
        System.out.println("initial balance: " + a1.getBalance());
        a1.credit(200);
        System.out.println("Account Balance after credit: " + a1.getBalance());
        a1.debit(100);
        System.out.println("Account Balance after debit: " + a1.getBalance());

        System.out.println("\nFor object 2:");
        System.out.println("initial balance: " + a2.getBalance());
        a2.credit(200);
        System.out.println("Account Balance after credit: " + a2.getBalance());
        a2.debit(300);
        System.out.println("Account Balance after debit: " + a2.getBalance());
    }
}

```

Output:

```
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> javac .\Account.java
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> java Account
Initial Balance is invalid!

For object 1:
initial balance: 1000
Account Balance after credit: 1200
Account Balance after debit: 1100

For object 2:
initial balance: 0
Account Balance after credit: 200
Debit amount exceeded account accountBalance!
Account Balance after debit: 200
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP>
```

4. (Employee Class) Create a class called Employee that includes three pieces of information as data members a first name (type string), a last name (type string) and a monthly salary (type int). Your class should have a constructor that initializes the three data members. If the monthly salary is not positive, set it to 0. Write a test program that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10 percent raise and display each Employee's yearly salary again.

   Program:

Employee.java

```java
public class Employee{
    private String firstName;
    private String lastName;
    private int monthlySalary;

    public Employee(String firstName, String lastName, int monthlySalary){
        this.firstName = firstName;
        this.lastName = lastName;
        if(monthlySalary >= 0){
            this.monthlySalary =  monthlySalary;
        }else{
            this.monthlySalary = 0;
        }

    }

    public int getYearlySalary(){
        return monthlySalary * 12;
    }

    public void giveRaise(double percentage){
        monthlySalary += monthlySalary * percentage / 100;
    }

    public static void main(String[] args){
        Employee emp1 = new Employee("abdillah", "mohd", 4000);
        Employee emp2 = new Employee("khalid", "khamis", -5000);

        System.out.println("Yearly Salary (Emp1): " + emp1.getYearlySalary());
        System.out.println("Yearly Salary (Emp2): " + emp2.getYearlySalary());

        emp1.giveRaise(10);
        emp2.giveRaise(10);

        System.out.println("Yearly Salary After Raise (Emp1): " + emp1.getYearlySalary());
        System.out.println("Yearly Salary After Raise (Emp2): " + emp2.getYearlySalary());
    }
}

```

Output:

```
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> javac .\Employee.java
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> java Employee
Yearly Salary (Emp1): 48000
Yearly Salary (Emp2): 0
Yearly Salary After Raise (Emp1): 52800
Yearly Salary After Raise (Emp2): 0
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP>
```

5. (Date Class) Create a class called Date that includes three pieces of information as data members a month (type int), a day (type int) and a year (type int). Your class should have a constructor with three parameters that uses the parameters to initialize the three data

members. For the purpose of this exercise, assume that the values provided for the year and day are correct, but ensure that the month value is in the range 1-12; if it is not, set the month to 1. Provide a member function displayDate that displays the month, day and year separated by forward slashes (/). Write a test program that demonstrates class Date's capabilities.

Program:

Date.java

```java
public class Date{
    private int month;
    private int day;
    private int year;

    public Date(int month, int day, int year){
        if (month >= 1 && month <= 12){
            this.month = month;
        }else{
            this.month = 1;
        }

        this.day = day;
        this.year = year;
    }

    public void displayDate(){
        System.out.println(month + "/" + day + "/" + year);
    }

    public static void main(String[] args) {
        Date date1 = new Date(12, 20, 2024);
        Date date2 = new Date(13, 10, 2024);

        System.out.print("\nDATE 1: ");
        date1.displayDate();
        System.out.print("DATE 2: ");
        date2.displayDate();

        System.out.print("\n");

    }
}

```

Output:

```
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> javac .\Date.java
PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP> java Date

DATE 1: 12/20/2024
DATE 2: 1/10/2024

PS C:\Users\Abdillah's PC\Desktop\Java\Exercise Lecture One OOP>
```