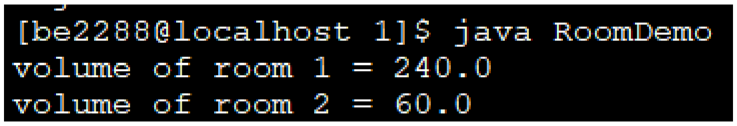
Name : Hrithvik Kondalkar
Roll : 002211001088

1. Create a class "Room" which will hold the "height", "width" and "breadth" of the room in three fields. This class also has a method "volume()" to calculate the volume of this room. Create another class "RoomDemo" which will use the earlier class, create instances of rooms, and display the volume of room.

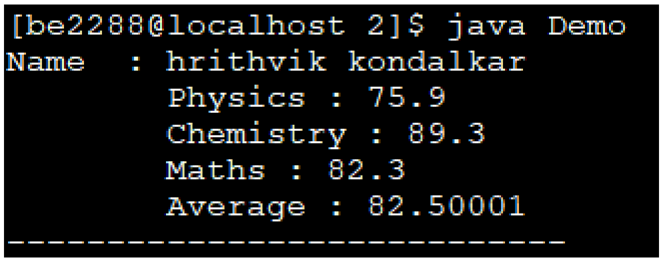
```
class Room{
  private float height, width, breadth;
  Room(float height, float width, float breadth){
    this.height = height;
    this.width = width;
    this.breadth = breadth;
  }
  public float volume(){
    return (height*width*breadth);
  }
}

class RoomDemo{
  public static void main(String[] args){
    Room obj1 = new Room(2,10,12);
    System.out.println("volume of room 1 = "+obj1.volume());
    Room obj2 = new Room(2,5,6);
    System.out.println("volume of room 2 = "+obj2.volume());
  }
}
```



```
[be2288@localhost 1]$ java RoomDemo
volume of room 1 = 240.0
volume of room 2 = 60.0
```

2. Write a program to implement a class "student" with the following members. Name of the student. Marks of the student obtained in three subjects. Function to assign initial values. Function to compute total average. Function to display the student's name and the total marks. Write an appropriate main() function to demonstrate the functioning of the above.

```
class Student{
 String name;
 float marks1, marks2, marks3;
 Student(String name){
  this.name = name;
 }
 public void setmarks(float m1, float m2, float m3){
  marks1 = m1;
  marks2 = m2;
  marks3 = m3;
 }
 private float average(){
  return (marks1+marks2+marks3)/3;
 }
 public void display(){
  System.out.println("Name  : "+name);
  System.out.println("\tPhysics : "+marks1);
  System.out.println("\tChemistry : "+marks2);
  System.out.println("\tMaths : "+marks3);
  System.out.println("\tAverage : "+ average());
  System.out.println("----------------------------");
 }
}
class Demo{
 public static void main(String[] args){
  String temp = new String("hrithvik kondalkar");
  Student s1 = new Student(temp);
  s1.setmarks(75.9f, 89.3f, 82.3f);
  s1.display();
 }
}
```

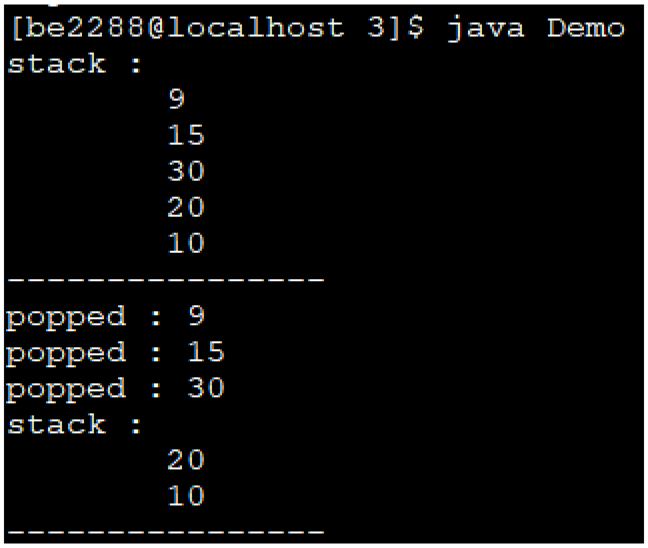


```
[be2288@localhost 2]$ java Demo
Name   : hrithvik kondalkar
        Physics : 75.9
        Chemistry : 89.3
        Maths : 82.3
        Average : 82.50001
----------------------------
```

3. Implement a class for stack of integers using an array. Please note that the operations defined for a stack data structure are as follows: “push”, “pop”, “print”. There should be a constructor to create an array of integers; the size of the array is provided by the user. Write a main() function to (i) create a stack to hold maximum of 30 integers; (ii) push the numbers 10, 20, 30, 15, 9 to the stack; (iii) print the stack; (iii) pop thrice and (iv) print the stack again.

```
class Stack{
 int size;
 int[] array;
 int currenttopindex;

 Stack(int size){
  array = new int[size];
  currenttopindex = 0;
  this.size = size;
 }

 public Stack push(int element){
  if(currenttopindex==size){
   System.out.println("Overflow!");
  }
  else{
   array[currenttopindex++] = element;
  }
  return this;
 }

 public int pop(){
  if(currenttopindex == 0){
   System.out.println("Underflow!");
   return -1;
  }
  else{
   int popped = array[--currenttopindex];
   System.out.println("popped : "+popped);
   return popped;
  }
 }
```

```
  public void print(){
    System.out.println("stack : ");
    for(int i=currenttopindex-1;i>=0;i--){
      System.out.println("\t"+array[i]);
    }
    System.out.println("----------------");
  }
}

class Demo{
  public static void main(String[] args){
    //stack size in commandline argument
    int arg = Integer.parseInt(args[0]);
    Stack mystack = new Stack(arg);
    mystack.push(10);
    mystack.push(20);
    mystack.push(30);
    mystack.push(15).push(9);

    mystack.print();

    mystack.pop();
    mystack.pop();
    mystack.pop();

    mystack.print();

  }
}
```



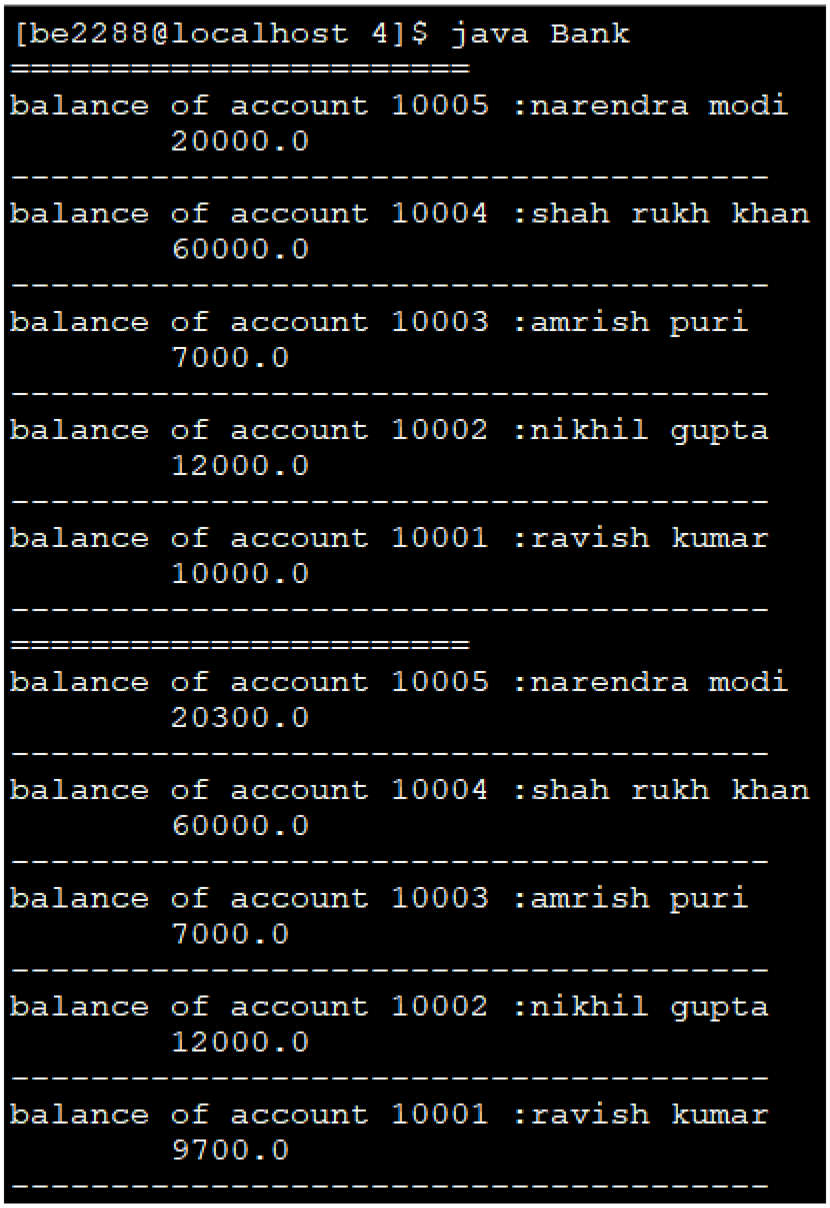
```
[be2288@localhost 3]$ java Demo
stack :
        9
        15
        30
        20
        10
----------------
popped : 9
popped : 15
popped : 30
stack :
        20
        10
----------------
```

4. Write a class “BankAccount” with the following instance variables: AccountNumber (an integer), balance a floating-point number), and “ownerName” (a String). Write proper constructor for this class. Also write methods balance, add (to deposit an amount), and subtract (to withdraw an amount) and implement them. Now create another class “AccountManager” that contains an array of BankAccount. Write methods create (to create an account), delete(to terminate an account), deposit (to deposit an amount to an account) and withdraw (to withdraw an amount from an account). Also write a class “Bank”, add main() function that creates an AccountManager and add 5 accounts. Now print the details of all accounts in this Bank.

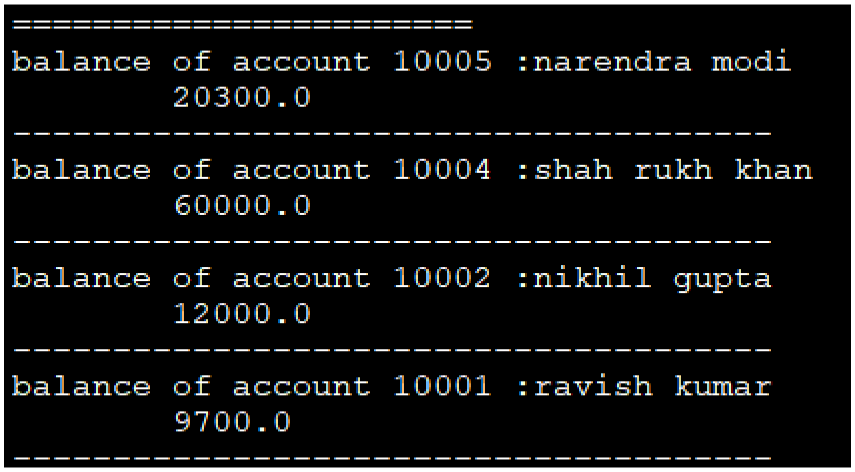
```
class BankAccount{
  private int accountnumber;
  private float balance;
  private String ownername;
  BankAccount(int accountnumber, float balance, String ownername){
    this.accountnumber = accountnumber;
    this.balance = balance;
    this.ownername = ownername;
  }
  public float balance(){
    System.out.println("balance of account "+accountnumber+" :" + ownername);
    System.out.println("\t"+balance);
    System.out.println("--------------------------------------");
    return balance;
  }
  public void add(float amount){
    balance+=amount;
  }
  public void subtract(float amount){
    if(balance<amount){
      System.out.println("Insufficient funds!");
    }
    else{
      balance-=amount;
    }
  }
  public boolean isacnum(int acnum){
    if(acnum==accountnumber){
      return true;
    }
    return false;
  }
}
```

```java
class AccountManger{
 private BankAccount[] accounts;
 private int currentinsertindex;

 AccountManger(){
  accounts = new BankAccount[5];
  currentinsertindex =0;
 }
 public void showall(){
  System.out.println("=====================");
  for (int i = currentinsertindex-1;i>=0;i--){
   accounts[i].balance();
  }
 }
 public void create(int accountnumber, float balance, String ownername){
  if(balance<5000){
   System.out.println("Minimum balance = 5000/- !!");
  }
  else{
   boolean uniqueacnum = true;
   for(int i=currentinsertindex-1;i>=0;i--){
    if(accounts[i].isacnum(accountnumber)){
     uniqueacnum = false;
     break;
    }
   }
   if(uniqueacnum){
    accounts[currentinsertindex++] = new BankAccount(accountnumber,balance,ownername);
   }
   else{
    System.out.println("account already exists !!!");
   }
  }
 }
 public void delete(int accountnumber){
  for(int i = currentinsertindex-1;i>=0;i--){
   if(accounts[i].isacnum(accountnumber)){
    if(i!=currentinsertindex){
     while(i!=currentinsertindex-1){
      accounts[i] = accounts[i+1];
      i++;
     }
     accounts[i] = null;
     currentinsertindex--;
    }
```

```
        else{
          accounts[i] = null;
          currentinsertindex--;
        }
        break;
      }
    }
  }


  public void deposit(int accountnumber,float addbalance){
    for(int i = currentinsertindex-1;i>=0;i--){
      if(accounts[i].isacnum(accountnumber)){
        accounts[i].add(addbalance);
      }
    }
  }
  public void withdraw(int accountnumber, float amount){
    for(int i = currentinsertindex-1;i>=0;i--){
      if(accounts[i].isacnum(accountnumber)){
        accounts[i].subtract(amount);
      }
    }
  }

}

class Bank{
  public static void main(String args[]){
    AccountManger mgr = new AccountManger();
    mgr.create(10001, 10000, "ravish kumar");
    mgr.create(10002, 12000, "nikhil gupta");
    mgr.create(10003, 7000, "amrish puri");
    mgr.create(10004, 60000, "shah rukh khan");
    mgr.create(10005, 20000, "narendra modi" );
    mgr.showall();
    mgr.deposit(10005, 300);
    mgr.withdraw(10001, 300);
    mgr.showall();
    mgr.delete(10003);
    mgr.showall();
  }
}
```



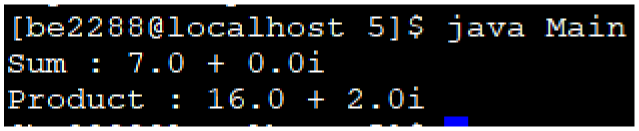
```
[be2288@localhost 4]$ java Bank
========================
balance of account 10005 :narendra modi
        20000.0
----------------------------------------
balance of account 10004 :shah rukh khan
        60000.0
----------------------------------------
balance of account 10003 :amrish puri
        7000.0
----------------------------------------
balance of account 10002 :nikhil gupta
        12000.0
----------------------------------------
balance of account 10001 :ravish kumar
        10000.0
----------------------------------------
========================
balance of account 10005 :narendra modi
        20300.0
----------------------------------------
balance of account 10004 :shah rukh khan
        60000.0
----------------------------------------
balance of account 10003 :amrish puri
        7000.0
----------------------------------------
balance of account 10002 :nikhil gupta
        12000.0
----------------------------------------
balance of account 10001 :ravish kumar
        9700.0
----------------------------------------
```



```
========================
balance of account 10005 :narendra modi
        20300.0
----------------------------------------
balance of account 10004 :shah rukh khan
        60000.0
----------------------------------------
balance of account 10002 :nikhil gupta
        12000.0
----------------------------------------
balance of account 10001 :ravish kumar
        9700.0
----------------------------------------
```

5. Write a class to represent complex numbers with necessary constructors. Write member functions to add, multiply two complex numbers. There should be three constructors: (i) to initialize real and imaginary parts to 0; (ii) to initialize imaginary part to 0 but to initialize the real part to user defined value; (iii) to initialize the real part and the imaginary part to user defined values. Also, write a main() function to (i) create two complex numbers 3+2i and 4-2i; (ii) to print the sum and product of those numbers.

```
class Complex{
 private float real, imaginary;
 Complex(float real, float imaginary){
  this.real = real;
  this.imaginary = imaginary;
 }
 Complex(float real){
  this.real = real;
  this.imaginary = 0.0f;
 }
 Complex(){
  real =0;
  imaginary = 0;
 }
 public Complex add(Complex other){
  Complex ret = new Complex(real+other.real, imaginary+other.imaginary);
  return ret; }
 public Complex multiply(Complex other){
  Complex ret = new Complex(((real*other.real)-(imaginary*other.imaginary)),
((real*other.imaginary)+(imaginary*other.real)));
  return ret;
 }
 public void show(){
  System.out.println(real+" + "+imaginary+"i");}
}
class Main{
 public static void main(String[] arg){
  Complex c1=new Complex(3,2);
  Complex c2=new Complex(4,-2);
  Complex sum = c1.add(c2);
  Complex prod = c1.multiply(c2);
  System.out.print("Sum : ");
  sum.show();
  System.out.print("Product : ");
  prod.show();
 }
}
```



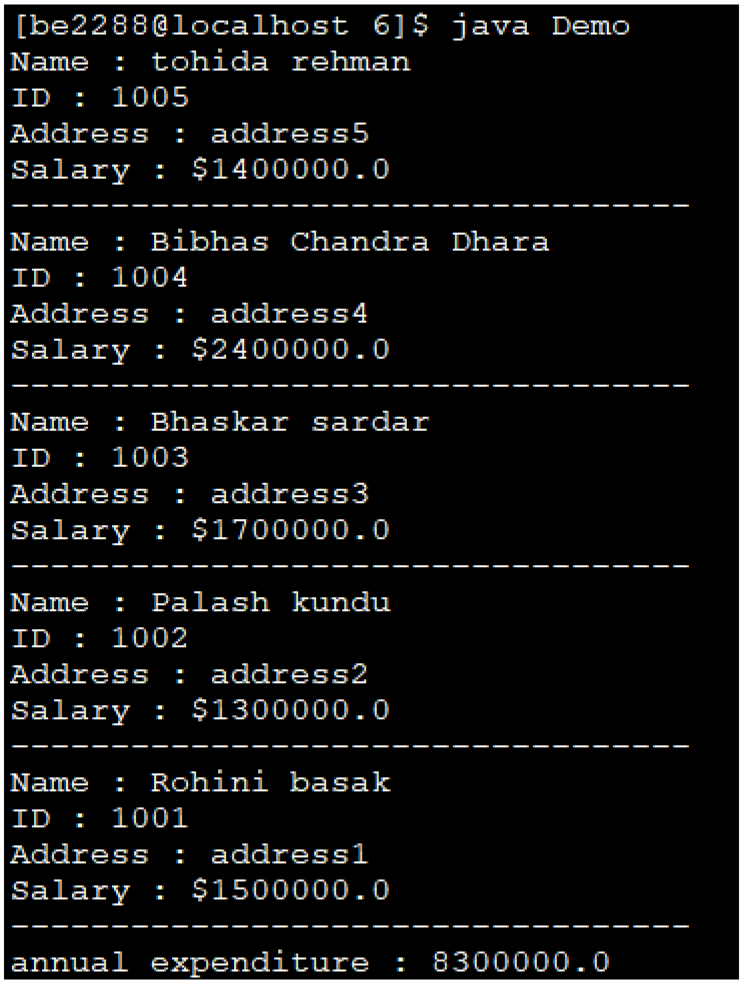
```
[be2288@localhost 5]$ java Main
Sum : 7.0 + 0.0i
Product : 16.0 + 2.0i
```

6. Write a Java class “Employee” containing information name, id, address, salary etc. Write necessary constructor and read/write methods. Create a class “Dept” that has a name, location etc. The “Dept” contains a number of “Employee”. Write methods “add” and “remove” to add and remove an employee to/from this department. Write a main() function and create “Information Technology” department. Add five employees and print yearly expenditure for this department.

```
class Employee{
  private static int currid = 1001;
  private String name;
  public int id;
  private String address;
  private float salary;


  Employee(String name, String address, float salary){
    id = (currid++);
    this.name = name;
    this.address = address;
    this.salary = salary;
  }


  public void show(){
    System.out.println("Name : "+name);
    System.out.println("ID : "+id);
    System.out.println("Address : "+address);
    System.out.println("Salary : $"+salary);
    System.out.println("----------------------------------");
  }
  public float getsal(){
    return salary;
  }
}
```

```
class Dept{
 public Employee employees[];
 public String name;
 public String location;
 private int currentinsertionindex;
 private int maxemployees;

 Dept(String name, int numberofemployees, String location){
  this.name = name;
  employees = new Employee[numberofemployees];
  this.location = location;
  currentinsertionindex =0;
  maxemployees = numberofemployees;
 }
 public void add(String name, String address, float salary){
  if(currentinsertionindex==maxemployees){
   System.out.println("department is full!!");
  }
  else{
   employees[currentinsertionindex++] = new Employee(name, address, salary);
  }
 }
 public void remove(int id){
  if(currentinsertionindex==0){
   System.out.println("no employees in department!!");
  }
  else{
   for(int i = currentinsertionindex-1;i>=0;i--){
    if(employees[i].id == id){
     while(i!=currentinsertionindex-1){
      employees[i] = employees[i+1];
      i++;
     }
     employees[i] = null;
     currentinsertionindex--;
     break;
    } }} }
 public float annualexpenditure(){
  float temp = 0;
  for(int i=currentinsertionindex-1;i>=0;i--){
   temp+=employees[i].getsal();
   employees[i].show();
  }
  return temp;
 }
}
```

```
class Demo{
  public static void main(String[] args){
    Dept it = new Dept("Information Technology", 5, "Jadavpur University Salt Lake Campus");
    it.add("Rohini basak", "address1", 1500000);
    it.add("Palash kundu", "address2", 1300000);
    it.add("Bhaskar sardar","address3", 1700000);
    it.add("Bibhas Chandra Dhara", "address4", 2400000);
    it.add("tohida rehman", "address5", 1400000);
    float temp = it.annualexpenditure();
    System.out.println("annual expenditure : "+temp);
  }
}
```

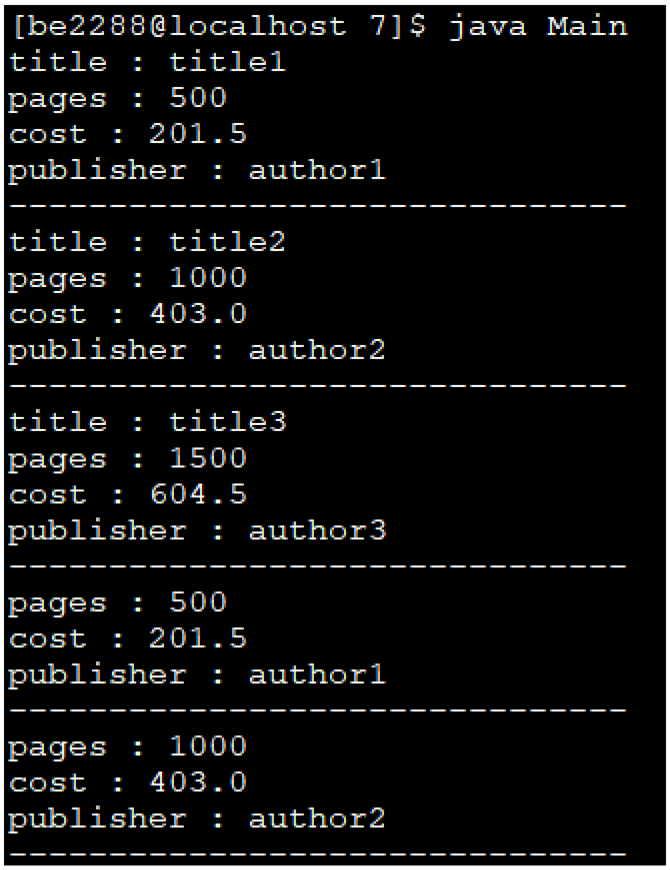


```
[be2288@localhost 6]$ java Demo
Name : tohida rehman
ID : 1005
Address : address5
Salary : $1400000.0
----------------------------------
Name : Bibhas Chandra Dhara
ID : 1004
Address : address4
Salary : $2400000.0
----------------------------------
Name : Bhaskar sardar
ID : 1003
Address : address3
Salary : $1700000.0
----------------------------------
Name : Palash kundu
ID : 1002
Address : address2
Salary : $1300000.0
----------------------------------
Name : Rohini basak
ID : 1001
Address : address1
Salary : $1500000.0
----------------------------------
annual expenditure : 8300000.0
```

7. Create an abstract class “Publication” with data members ‘noOfPages’, ‘price’, ‘publisherName’ etc. with their accessor/modifier functions. Now create two sub-classes “Book” and “Journal”. Create a class Library that contains a list of Publications. Write a main() function and create three Books and two Journals, add them to library and print the details of all publications.

```
abstract class Publication{
        int noOfPages;
        float price;
        String publisherName;
        Publication(int noOfPages, float price, String publisherName){
                this.noOfPages = noOfPages;
                this.price = price;
                this.publisherName = publisherName;
        }
        void show(){
                System.out.println("pages : "+noOfPages);
                System.out.println("cost : "+ price);
                System.out.println("publisher : "+publisherName);
                System.out.println("--------------------------------");
        }
}
class Book extends Publication{
        String title;
        Book(String title ,int noOfPages, float price, String publisherName){
                super(noOfPages, price, publisherName);
                this.title = title;
        }
        @Override void show(){
        System.out.println("title : "+title);
        super.show();
        }
}
class Journal extends Publication{
        Journal(int noOfPages, float price, String publisherName){
                super(noOfPages, price, publisherName);
        }
}
```

```
class Main{
	public static void main(String[] a){
		Publication[] library = new Publication[5];
		for(int i=0;i<3;i++){
			library[i] = new Book("title"+(i+1),500*(i+1),201.5f*(i+1),"author"+(i+1));
		}
		for(int i=0;i<2;i++){
			library[i+3] = new Journal(500*(i+1),201.5f*(i+1),"author"+(i+1));
		}
		for(Publication a1 : library){
			a1.show();
		}
	}
}
```

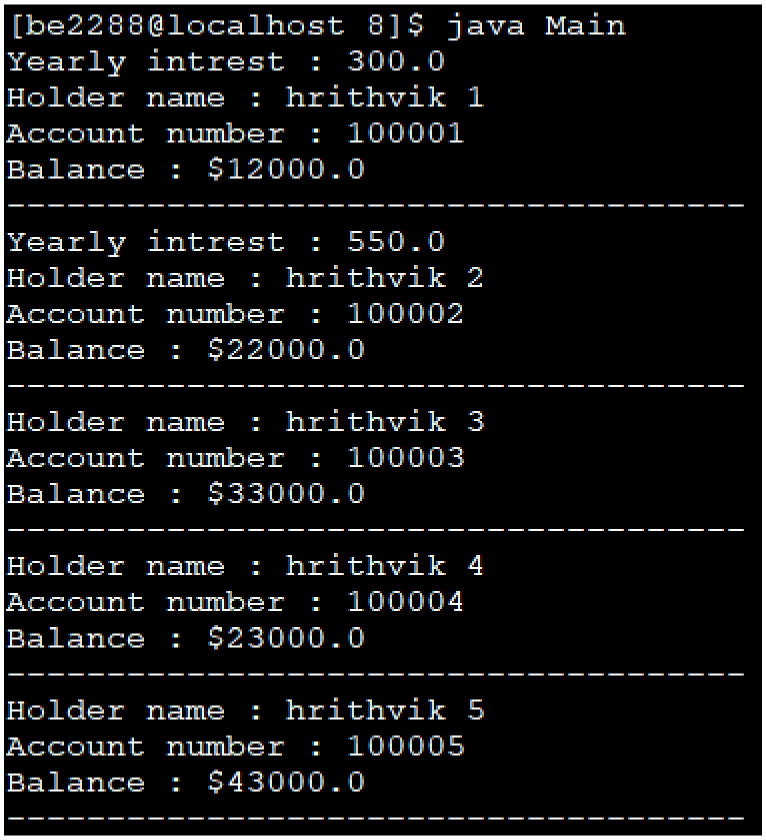


```
[be2288@localhost 7]$ java Main
title : title1
pages : 500
cost : 201.5
publisher : author1
-------------------------------
title : title2
pages : 1000
cost : 403.0
publisher : author2
-------------------------------
title : title3
pages : 1500
cost : 604.5
publisher : author3
-------------------------------
pages : 500
cost : 201.5
publisher : author1
-------------------------------
pages : 1000
cost : 403.0
publisher : author2
-------------------------------
```

8. Write a class for “Account” containing data members ‘accountNumber’, ‘holderName’, ‘balance’ and add constructors and necessary accessor/modifier functions for these data members. Now create two class “SavingsAccount” and “CurrentAccount” extending from this class. SavingsAccount will have a member variable ‘interestRate’ and member function ‘calculateYearlyInterest’. Write another class “Manager” that contains a list Account. Also write a main() function to create an instance of Manager class. Add two SavingsAccount and three CurrentAccount to Manager. Calculate interest of each SavingsAccount. Print the details of all accounts.

```
abstract class Account {
    int accountNumber;
    static int nextaccountnumber = 100001;
    String holderName;
    float balance;

    Account(String name, float balance) {
        accountNumber = nextaccountnumber++;
        holderName = name;
        this.balance = balance;
    }

    void show() {
        System.out.println("Holder name : " + holderName);
        System.out.println("Account number : " + accountNumber);
        System.out.println("Balance : $" + balance);
        System.out.println("-------------------------------------");
    }
}
class SavingsAccount extends Account {
    static float intrestRate = 0.025f;
    float calculateYearlyIntrest() {
        return intrestRate * (super.balance); }
    static void updateIntrest(float newrate) {
        intrestRate = newrate;}
    SavingsAccount(String name, float balance) {
        super(name, balance);}
    @Override
    void show() {
        System.out.println("Yearly intrest : " + calculateYearlyIntrest());
        super.show();}
}
```

```
class CurrentAccount extends Account {
   CurrentAccount(String name, float balance) {
      super(name, balance);
   }
}

class Manager {
   Account[] accounts;
   int nextindex;
   int size;

   Manager(int numberofaccounts) {
      nextindex = 0;
      accounts = new Account[numberofaccounts];
      size = numberofaccounts;
   }

   void addSavingsAc(String name, float balance) {
      if (nextindex < size) {
         accounts[nextindex++] = new SavingsAccount(name, balance);
      }
   }

   void removeAc(int acnumber) {
      for (int i = 0; i < nextindex; i++) {
         if (accounts[i].accountNumber == acnumber) {
            accounts[i] = null;
            while (i < nextindex - 1) {
               accounts[i] = accounts[i + 1];
               i++;
            }
            accounts[i] = null;
            nextindex--;
            break;
         }
      }
   }

   void addCurrentAc(String name, float balance) {
      if (nextindex < size) {
         accounts[nextindex++] = new CurrentAccount(name, balance);
      }
   }
   void showall() {
       for (int i = 0; i < nextindex; i++) {
           accounts[i].show();
       }
    }
}

class Main {
    public static void main(String[] args) {
        Manager m = new Manager(5);
        m.addSavingsAc("hrithvik 1", 12000);
        m.addSavingsAc("hrithvik 2", 22000);
        m.addCurrentAc("hrithvik 3", 33000);
        m.addCurrentAc("hrithvik 4", 23000);
        m.addCurrentAc("hrithvik 5", 43000);
        m.showall();
    }
}
```



```
[be2288@localhost 8]$ java Main
Yearly intrest : 300.0
Holder name : hrithvik 1
Account number : 100001
Balance : $12000.0
-------------------------------------
Yearly intrest : 550.0
Holder name : hrithvik 2
Account number : 100002
Balance : $22000.0
-------------------------------------
Holder name : hrithvik 3
Account number : 100003
Balance : $33000.0
-------------------------------------
Holder name : hrithvik 4
Account number : 100004
Balance : $23000.0
-------------------------------------
Holder name : hrithvik 5
Account number : 100005
Balance : $43000.0
-------------------------------------
```

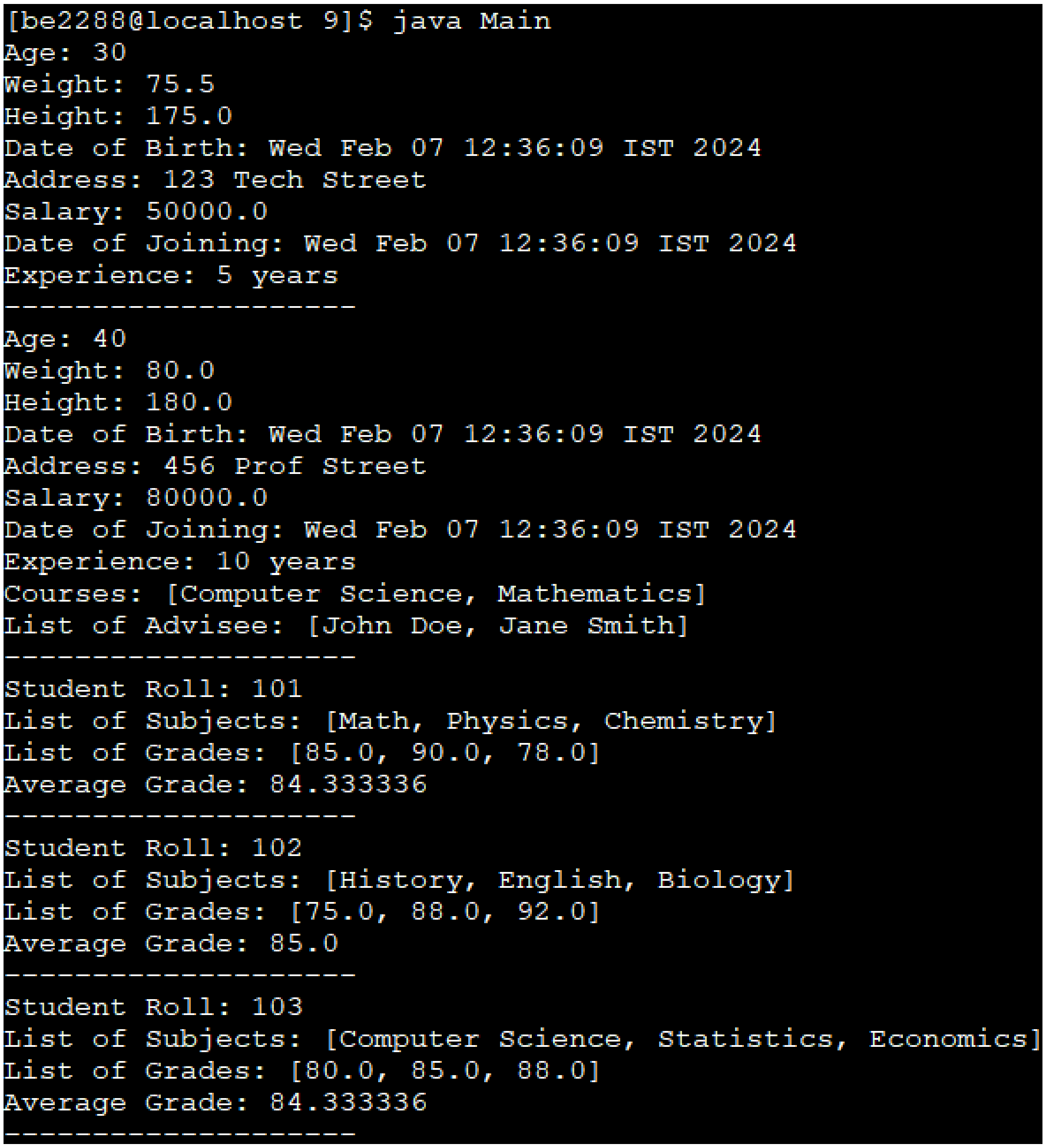
9. Implement a class for a “Person”. Person has data members ‘age’, ’weight’, ‘height’, ‘dateOfBirth’, ‘address’ with proper reader/write methods etc. Now create two subclasses “Employee” and “Student”. Employee will have additional data member ‘salary’, ‘dateOfJoining’, ‘experience’ etc. Student has data members ‘roll’, ‘listOfSubjects’, their marks and methods ‘calculateGrade’. Again create two sub-classes “Technician” and “Professor” from Employee. Professor has data members ‘courses’, ‘listOfAdvisee’ and their add/remove methods. Write a main() function to demonstrate the creation of objects of different classes and their method calls.

```
import java.util.*;

abstract class Person {
   int age;
   float weightKG;
   float heightCM;
   Date dateOfBirth;
   String address;

   Person(int age, float weightKG, float heightCM, Date dateOfBirth, String address) {
      this.age = age;
      this.weightKG = weightKG;
      this.heightCM = heightCM;
      this.dateOfBirth = dateOfBirth;
      this.address = address;
   }
   abstract void display();
}
class Employee extends Person {
   float salary;
   Date dateOfJoining;
   int experience;

   Employee(int age, float weightKG, float heightCM, Date dateOfBirth, String address,
         float salary, Date dateOfJoining, int experience) {
      super(age, weightKG, heightCM, dateOfBirth, address);
      this.salary = salary;
      this.dateOfJoining = dateOfJoining;
      this.experience = experience;
   }
```

```
    void display() {
        System.out.println("Age: " + age);
        System.out.println("Weight: " + weightKG);
        System.out.println("Height: " + heightCM);
        System.out.println("Date of Birth: " + dateOfBirth);
        System.out.println("Address: " + address);
        System.out.println("Salary: " + salary);
        System.out.println("Date of Joining: " + dateOfJoining);
        System.out.println("Experience: " + experience + " years");
    }
}

class Student extends Person {
    int roll;
    String[] listOfSubjects;
    float[] marks;

    Student(int age, float weightKG, float heightCM, Date dateOfBirth, String address,
            int roll, String[] listOfSubjects, float[] marks) {
        super(age, weightKG, heightCM, dateOfBirth, address);
        this.roll = roll;
        this.listOfSubjects = listOfSubjects;
        this.marks = marks;
    }

    float calculateGrade() {
        float sum = 0;
        for (float mark : marks) {
            sum += mark;
        }
        return sum / marks.length;
    }

    void display() {
        System.out.println("Student Roll: " + roll);
        System.out.println("List of Subjects: " + Arrays.toString(listOfSubjects));
        System.out.println("List of Grades: " + Arrays.toString(marks));
        System.out.println("Average Grade: " + calculateGrade());
    }
}
```

```
class Technician extends Employee {
   Technician(int age, float weightKG, float heightCM, Date dateOfBirth, String address,
           float salary, Date dateOfJoining, int experience) {
      super(age, weightKG, heightCM, dateOfBirth, address, salary, dateOfJoining, experience);
   }
}

class Professor extends Employee {
   String[] courses;
   String[] listOfAdvisee;

   Professor(int age, float weightKG, float heightCM, Date dateOfBirth, String address,
           float salary, Date dateOfJoining, int experience,
           String[] courses, String[] listOfAdvisee) {
      super(age, weightKG, heightCM, dateOfBirth, address, salary, dateOfJoining, experience);
      this.courses = courses;
      this.listOfAdvisee = listOfAdvisee;
   }

   @Override
   void display() {
      super.display();
      System.out.println("Courses: " + Arrays.toString(courses));
      System.out.println("List of Advisee: " + Arrays.toString(listOfAdvisee));
   }
}

class Main {
   public static void main(String[] args) {
      Person[] employees = new Person[]{
            new Technician(30, 75.5f, 175.0f, new Date(), "123 Tech Street",
                  50000.0f, new Date(), 5),
            new Professor(40, 80.0f, 180.0f, new Date(), "456 Prof Street",
                  80000.0f, new Date(), 10,
                  new String[]{"Computer Science", "Mathematics"},
                  new String[]{"John Doe", "Jane Smith"})
      };
      Person[] students = new Person[]{
            new Student(20, 65.0f, 170.0f, new Date(), "789 Student Street",
                  101, new String[]{"Math", "Physics", "Chemistry"},
                  new float[]{85.0f, 90.0f, 78.0f}),
            new Student(22, 70.0f, 175.0f, new Date(), "987 Student Street",
                  102, new String[]{"History", "English", "Biology"},
                  new float[]{75.0f, 88.0f, 92.0f}),
            new Student(21, 68.0f, 172.0f, new Date(), "654 Student Street",
                  103, new String[]{"Computer Science", "Statistics", "Economics"},
                  new float[]{80.0f, 85.0f, 88.0f})
      };
```

```
        for (Person employee : employees) {
            employee.display();
            System.out.println("--------------------");
        }

        for (Person student : students) {
            ((Student) student).display();
            System.out.println("--------------------");
        }
    }
}
```

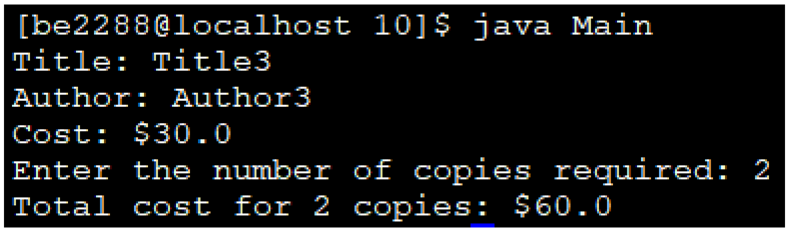


```
[be2288@localhost 9]$ java Main
Age: 30
Weight: 75.5
Height: 175.0
Date of Birth: Wed Feb 07 12:36:09 IST 2024
Address: 123 Tech Street
Salary: 50000.0
Date of Joining: Wed Feb 07 12:36:09 IST 2024
Experience: 5 years
--------------------
Age: 40
Weight: 80.0
Height: 180.0
Date of Birth: Wed Feb 07 12:36:09 IST 2024
Address: 456 Prof Street
Salary: 80000.0
Date of Joining: Wed Feb 07 12:36:09 IST 2024
Experience: 10 years
Courses: [Computer Science, Mathematics]
List of Advisee: [John Doe, Jane Smith]
--------------------
Student Roll: 101
List of Subjects: [Math, Physics, Chemistry]
List of Grades: [85.0, 90.0, 78.0]
Average Grade: 84.333336
--------------------
Student Roll: 102
List of Subjects: [History, English, Biology]
List of Grades: [75.0, 88.0, 92.0]
Average Grade: 85.0
--------------------
Student Roll: 103
List of Subjects: [Computer Science, Statistics, Economics]
List of Grades: [80.0, 85.0, 88.0]
Average Grade: 84.333336
--------------------
```

10. A bookshop maintains the inventory of books that are being sold at the shop. The list includes details such as author, title, publisher, cost and stock position. Whenever a customer wants a book, the sales person inputs the title and author and the system searches the list and displays whether it is available or not. If it is not, an appropriate message is displayed. If it is, then the system displays the book details and details and requests for the number of copies required. If the required copies are available, the total cost of the requested copies is displayed, otherwise the message “requires copies not in stock” is displayed. Design a system using a class called “Book” with suitable member methods and constructors.

```
import java.util.HashMap;
import java.util.Scanner;
class Book {
  String author;
  String title;
  float cost;
  public Book(String author, String title, float cost) {
    this.author = author;
    this.title = title;
    this.cost = cost;
  }
  void display() {
    System.out.println("Title: " + title);
    System.out.println("Author: " + author);
    System.out.println("Cost: $" + cost);
  }
}
class BookStore {
  HashMap<String, Integer> inventory;
  HashMap<String, Book> bookDetails;
  BookStore() {
    inventory = new HashMap<>();
    bookDetails = new HashMap<>();
  }
  void updateInventory(Book book, int newInventoryCount) {
    String key = book.title + " BY : " + book.author;
    inventory.put(key, newInventoryCount);
    bookDetails.put(key, book);
  }
  void searchInventory(String title, String author) {
    String key = title + " BY : " + author;
    if (bookDetails.containsKey(key)) {
      Book book = bookDetails.get(key);
      book.display();
```

```
            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter the number of copies required: ");
            int requiredCopies = scanner.nextInt();
            if (requiredCopies <= inventory.get(key)) {
                float totalCost = requiredCopies * book.cost;
                System.out.println("Total cost for " + requiredCopies + " copies: $" + totalCost);
            } else {
                System.out.println("Required amount not available.");
            }
        } else {
            System.out.println("Book unavailable.");
        }
    }
}
class Main {
    public static void main(String[] args) {
        Book book1 = new Book("Author1", "Title1", 20.0f);
        Book book2 = new Book("Author2", "Title2", 25.0f);
        Book book3 = new Book("Author3", "Title3", 30.0f);
        Book book4 = new Book("Author4", "Title4", 15.0f);
        Book book5 = new Book("Author5", "Title5", 18.0f);
        BookStore bookstore = new BookStore();
        bookstore.updateInventory(book1, 10);
        bookstore.updateInventory(book2, 15);
        bookstore.updateInventory(book3, 8);
        bookstore.updateInventory(book4, 20);
        bookstore.updateInventory(book5, 12);
        bookstore.searchInventory("Title3", "Author3");
    }
}
```

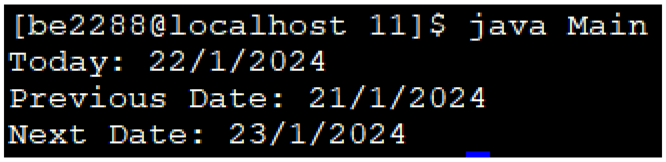


```
[be2288@localhost 10]$ java Main
Title: Title3
Author: Author3
Cost: $30.0
Enter the number of copies required: 2
Total cost for 2 copies: $60.0
```

11. Implement a class for “Date”. Write member functions for (i) getting the previous day, (iv) getting the next day, (iii) printing a day There should be four constructors: (i) day, month and year are initialized to 01, 01, 1970; (ii) day is initialized to user specified value but month and year are initialized to 01, 1970; (iii) day, month are initialized to user specified value but year is initialized to 1970; (iv) day, month and year are initialized to user defined values. Also, write a main() function to (i) create a date object; (ii) print the next and the previous day.

```
class Date {
   public int date = 1;
   public int month = 1;
   public int year = 1970;
   public boolean leap;
   static int[] month30 = {4, 6, 9, 11};
   static int[] month31 = {1, 3, 5, 7, 8, 10, 12};
   Date(int date, int month, int year) {
      this.date = date;
      this.month = month;
      this.year = year;
      this.leap = isLeap();
   }
   Date(int date) {
      this.date = date;
      this.leap = isLeap();
   }
   Date(int date, int month) {
      this.date = date;
      this.month = month;
      this.leap = isLeap();
   }
   Date() {
      this.leap = isLeap();
   }
   String previousDate() {
      if (date > 1) {
         return (date - 1) + "/" + month + "/" + year;
      } else if (month > 1) {
         int prevMonth = month - 1;
         int prevDate = (prevMonth == 2) ? (leap ? 29 : 28) : (contains(month31, prevMonth) ? 31 : 30);
         return prevDate + "/" + prevMonth + "/" + year;
      } else {
         int prevYear = year - 1;
         return "31/12/" + prevYear;
      }
   }
```

```
    String nextDate() {
        int maxDays = getMaxDaysInMonth(month, leap);
        if (date < maxDays) {
            return (date + 1) + "/" + month + "/" + year;
        } else if (month < 12) {
            return "1/" + (month + 1) + "/" + year;
        } else {
            int nextYear = year + 1;
            return "1/1/" + nextYear;
        }
    }
    String today() {
        return date + "/" + month + "/" + year;
    }
    private boolean isLeap() {
        return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
    }
    private boolean contains(int[] array, int value) {
        for (int num : array) {
            if (num == value) {
                return true;
            }
        }
        return false;
    }
    private int getMaxDaysInMonth(int month, boolean leap) {
        if (contains(month30, month)) {
            return 30;
        } else if (contains(month31, month)) {
            return 31;
        } else {
            return leap ? 29 : 28;
        }
    }
}
class Main {
    public static void main(String[] args) {
        Date todayDate = new Date(22, 1, 2024);
        System.out.println("Today: " + todayDate.today());
        System.out.println("Previous Date: " + todayDate.previousDate());
        System.out.println("Next Date: " + todayDate.nextDate());
    }
}
```



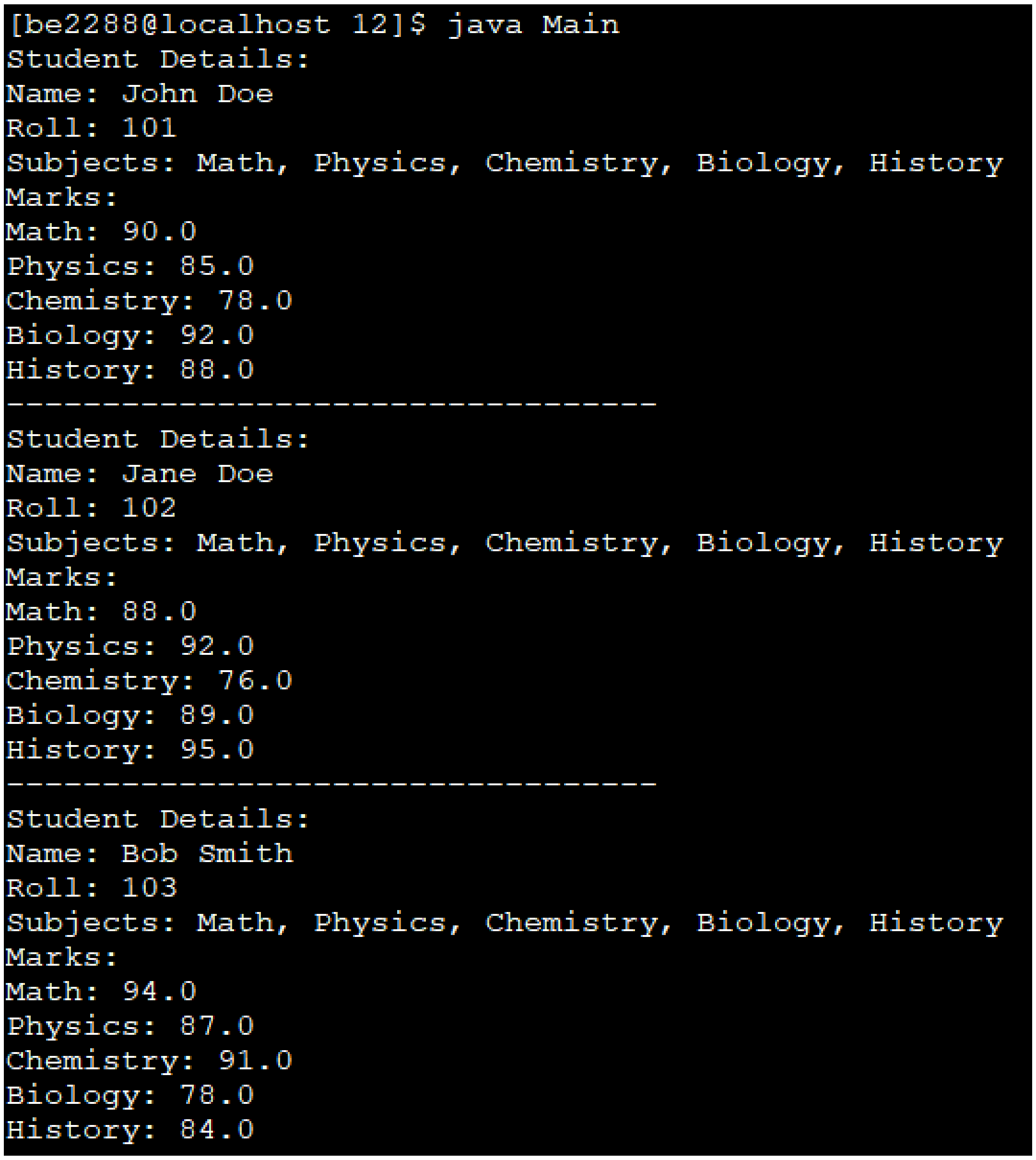
```
[be2288@localhost 11]$ java Main
Today: 22/1/2024
Previous Date: 21/1/2024
Next Date: 23/1/2024
```

12. Implement a class for a “Student”. Information about a student includes name, roll no and an array of five subject names. The class should have suitable constructor and get/set methods. Implement a class “TabulationSheet”. A tabulation sheet contains roll numbers and marks of each student for a particular subject. This class should have a method for adding the marks and roll no of a student. Implement a class “MarkSheet”. A mark sheet contains marks of all subjects for a particular student. This class should have a method to add name of a student and marks in each subject. Write a main() function to create three “Student” objects, Five “Tabulationsheet” objects for Five subjects and three “Marksheet” object for three students. Print the mark sheets.

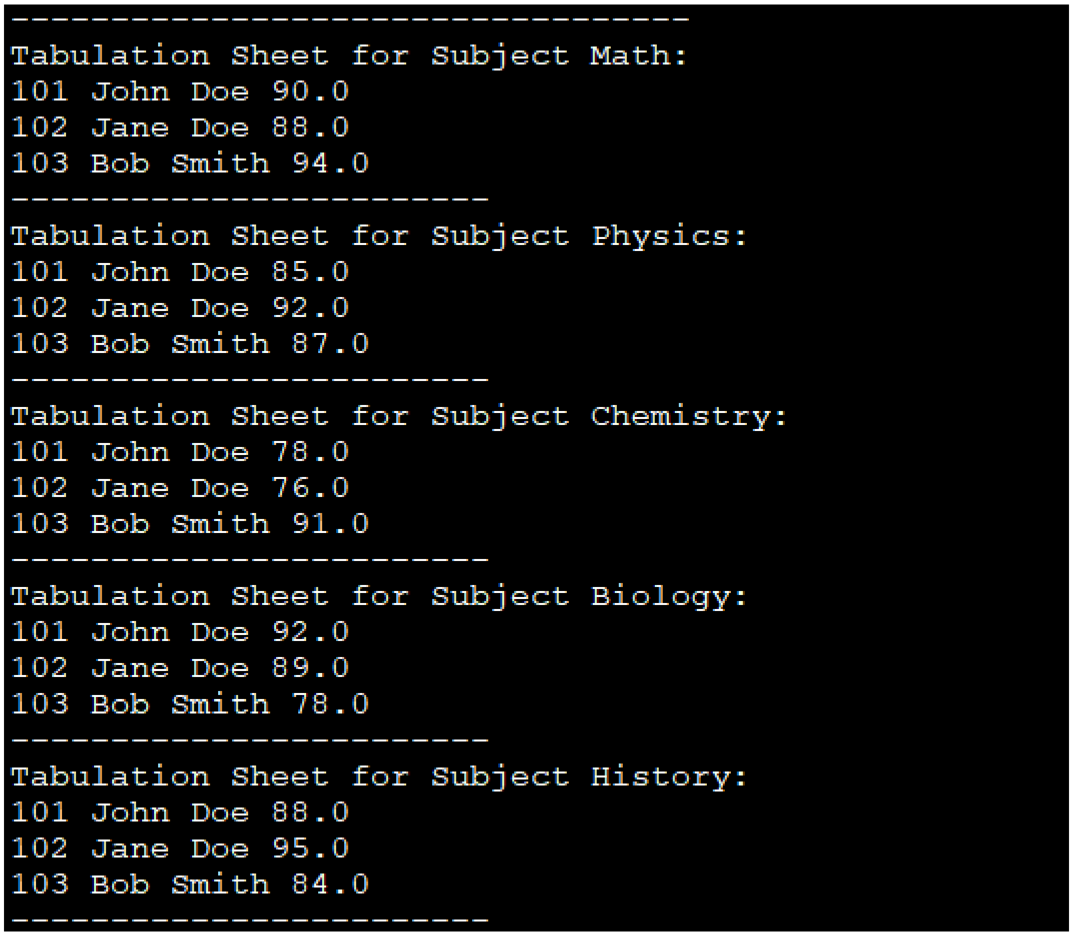
```
class Student {
   String name;
   int roll;
   String[] subjects;
   Student(String name, int roll, String subject1, String subject2, String subject3, String subject4,
String subject5) {
      this.name = name;
      this.roll = roll;
      subjects = new String[5];
      subjects[0] = subject1;
      subjects[1] = subject2;
      subjects[2] = subject3;
      subjects[3] = subject4;
      subjects[4] = subject5;
   }
   void display() {
      System.out.println("Student Details:");
      System.out.println("Name: " + name);
      System.out.println("Roll: " + roll);
      System.out.println("Subjects: " + String.join(", ", subjects));
   }
}
class Marksheet {
   Student studentdetails;
   float[] marks;
   Marksheet(Student student, float marks1, float marks2, float marks3, float marks4, float marks5) {
      studentdetails = student;
      marks = new float[5];
      marks[0] = marks1;
      marks[1] = marks2;
      marks[2] = marks3;
      marks[3] = marks4;
      marks[4] = marks5;
   }
```

```
  void display() {
    studentdetails.display();
    System.out.println("Marks:");
    for (int i = 0; i < marks.length; i++) {
      System.out.println(studentdetails.subjects[i] + ": " + marks[i]);
    }
    System.out.println("-----------------------------------");
  }
}
class TabulationSheet {
  Marksheet[] marksheetreference;
  int nextemptyindex = 0;
  int maxelements;
  int subjectindex;
  float[] marks;
  TabulationSheet(int numberofstudents, int subjectindex) {
    marksheetreference = new Marksheet[numberofstudents];
    marks = new float[numberofstudents];
    maxelements = numberofstudents;
    this.subjectindex = subjectindex;
  }
  void append(Marksheet studentmarks) {
    if (nextemptyindex < maxelements) {
      marksheetreference[nextemptyindex] = studentmarks;
      marks[nextemptyindex] = studentmarks.marks[subjectindex];
      nextemptyindex += 1;
    } else {
      System.out.println("Tabulation sheet is full!");
    }
  }
  void display() {
    System.out.println("Tabulation Sheet for Subject " +
marksheetreference[0].studentdetails.subjects[subjectindex] + ":");
    for (int i = 0; i < nextemptyindex; i++) {
      Student student = marksheetreference[i].studentdetails;
      System.out.println(  student.roll + " " + student.name + " " + marks[i]);
    }
    System.out.println("------------------------");
  }
}
```

```
class Main {
   public static void main(String[] args) {
      Student student1 = new Student("John Doe", 101, "Math", "Physics", "Chemistry", "Biology",
"History");
      Student student2 = new Student("Jane Doe", 102, "Math", "Physics", "Chemistry", "Biology",
"History");
      Student student3 = new Student("Bob Smith", 103, "Math", "Physics", "Chemistry", "Biology",
"History");
      Marksheet marksheet1 = new Marksheet(student1, 90, 85, 78, 92, 88);
      Marksheet marksheet2 = new Marksheet(student2, 88, 92, 76, 89, 95);
      Marksheet marksheet3 = new Marksheet(student3, 94, 87, 91, 78, 84);
      TabulationSheet tabulationSheetMath = new TabulationSheet(3, 0);
      TabulationSheet tabulationSheetPhysics = new TabulationSheet(3, 1);
      TabulationSheet tabulationSheetChemistry = new TabulationSheet(3, 2);
      TabulationSheet tabulationSheetBiology = new TabulationSheet(3, 3);
      TabulationSheet tabulationSheetHistory = new TabulationSheet(3, 4);
      tabulationSheetMath.append(marksheet1);
      tabulationSheetMath.append(marksheet2);
      tabulationSheetMath.append(marksheet3);
      tabulationSheetPhysics.append(marksheet1);
      tabulationSheetPhysics.append(marksheet2);
      tabulationSheetPhysics.append(marksheet3);
      tabulationSheetChemistry.append(marksheet1);
      tabulationSheetChemistry.append(marksheet2);
      tabulationSheetChemistry.append(marksheet3);
      tabulationSheetBiology.append(marksheet1);
      tabulationSheetBiology.append(marksheet2);
      tabulationSheetBiology.append(marksheet3);
      tabulationSheetHistory.append(marksheet1);
      tabulationSheetHistory.append(marksheet2);
      tabulationSheetHistory.append(marksheet3);
      marksheet1.display();
      marksheet2.display();
      marksheet3.display();
      tabulationSheetMath.display();
      tabulationSheetPhysics.display();
      tabulationSheetChemistry.display();
      tabulationSheetBiology.display();
      tabulationSheetHistory.display();
   }
}
```



```
[be2288@localhost 12]$ java Main
Student Details:
Name: John Doe
Roll: 101
Subjects: Math, Physics, Chemistry, Biology, History
Marks:
Math: 90.0
Physics: 85.0
Chemistry: 78.0
Biology: 92.0
History: 88.0
----------------------------------
Student Details:
Name: Jane Doe
Roll: 102
Subjects: Math, Physics, Chemistry, Biology, History
Marks:
Math: 88.0
Physics: 92.0
Chemistry: 76.0
Biology: 89.0
History: 95.0
----------------------------------
Student Details:
Name: Bob Smith
Roll: 103
Subjects: Math, Physics, Chemistry, Biology, History
Marks:
Math: 94.0
Physics: 87.0
Chemistry: 91.0
Biology: 78.0
History: 84.0
```



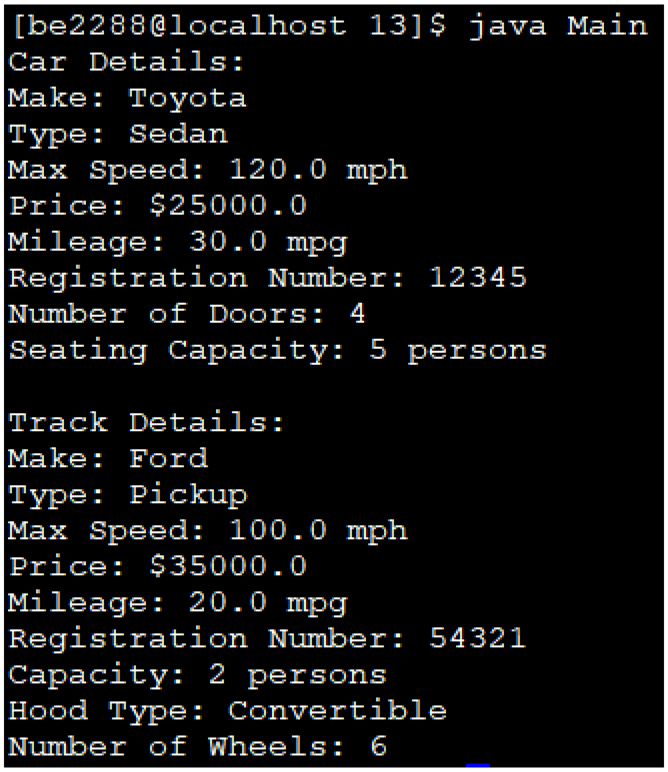
```
-----------------------------------
Tabulation Sheet for Subject Math:
101 John Doe 90.0
102 Jane Doe 88.0
103 Bob Smith 94.0
-------------------------
Tabulation Sheet for Subject Physics:
101 John Doe 85.0
102 Jane Doe 92.0
103 Bob Smith 87.0
-------------------------
Tabulation Sheet for Subject Chemistry:
101 John Doe 78.0
102 Jane Doe 76.0
103 Bob Smith 91.0
-------------------------
Tabulation Sheet for Subject Biology:
101 John Doe 92.0
102 Jane Doe 89.0
103 Bob Smith 78.0
-------------------------
Tabulation Sheet for Subject History:
101 John Doe 88.0
102 Jane Doe 95.0
103 Bob Smith 84.0
-------------------------
```

13. Create a base class “Automobile”. An Automobile contains data members ‘make’, ‘type’, ‘maxSpeed’, ‘price’, ‘mileage’, ‘registrationNumber’ etc. with their reader/writer methods. Now create two sub-classes “Track” and “Car”. Track has data members ‘capacity’, ‘hoodType’, ‘noOfWheels’ etc. Car has data members ‘noOfDoors’, ‘seatingCapacity’ and their reader/writer methods. Create a main() function to demonstrate this.

```
class Automobile {
    String make;
    String type;
    float maxSpeed;
    float price;
    float mileage;
    int registrationNumber;

    public Automobile(String make, String type, float maxSpeed, float price, float mileage, int
registrationNumber) {
        this.make = make;
        this.type = type;
        this.maxSpeed = maxSpeed;
        this.price = price;
        this.mileage = mileage;
        this.registrationNumber = registrationNumber;
    }

    void display() {
        System.out.println("Make: " + make);
        System.out.println("Type: " + type);
        System.out.println("Max Speed: " + maxSpeed + " mph");
        System.out.println("Price: $" + price);
        System.out.println("Mileage: " + mileage + " mpg");
        System.out.println("Registration Number: " + registrationNumber);
    }
}
```

```
class Track extends Automobile {
   int capacity;
   String hoodType;
   int numberOfWheels;


   public Track(String make, String type, float maxSpeed, float price, float mileage, int
registrationNumber,
                int capacity, String hoodType, int numberOfWheels) {
      super(make, type, maxSpeed, price, mileage, registrationNumber);
      this.capacity = capacity;
      this.hoodType = hoodType;
      this.numberOfWheels = numberOfWheels;
   }


   @Override
   void display() {
      super.display();
      System.out.println("Capacity: " + capacity + " persons");
      System.out.println("Hood Type: " + hoodType);
      System.out.println("Number of Wheels: " + numberOfWheels);
   }
}

class Car extends Automobile {
   int noOfDoors;
   int seatingCapacity;


   public Car(String make, String type, float maxSpeed, float price, float mileage, int
registrationNumber,
               int noOfDoors, int seatingCapacity) {
      super(make, type, maxSpeed, price, mileage, registrationNumber);
      this.noOfDoors = noOfDoors;
      this.seatingCapacity = seatingCapacity;
   }


   @Override
   void display() {
      super.display();
      System.out.println("Number of Doors: " + noOfDoors);
      System.out.println("Seating Capacity: " + seatingCapacity + " persons");
   }
}
```

```
class Main {
   public static void main(String[] args) {
      // Create car and track objects and display()
      Car myCar = new Car("Toyota", "Sedan", 120.0f, 25000.0f, 30.0f, 12345, 4, 5);
      Track myTrack = new Track("Ford", "Pickup", 100.0f, 35000.0f, 20.0f, 54321, 2, "Convertible", 6);

      System.out.println("Car Details:");
      myCar.display();

      System.out.println("\nTrack Details:");
      myTrack.display();
   }
}
```



```
[be2288@localhost 13]$ java Main
Car Details:
Make: Toyota
Type: Sedan
Max Speed: 120.0 mph
Price: $25000.0
Mileage: 30.0 mpg
Registration Number: 12345
Number of Doors: 4
Seating Capacity: 5 persons

Track Details:
Make: Ford
Type: Pickup
Max Speed: 100.0 mph
Price: $35000.0
Mileage: 20.0 mpg
Registration Number: 54321
Capacity: 2 persons
Hood Type: Convertible
Number of Wheels: 6
```

14. Implement the classes for the following inheritance hierarchies. Create an interface “Shape” that contains methods ‘area’, ‘draw’, ‘rotate’, ‘move’ etc. Now create two classes “Circle” and “Rectangle” that implement this ‘Shape’ interface and implement the methods ‘area’, ‘move’, etc. Write a main() function to create two “Circle” and three “Rectangle”, then move them. Print the details of circles and rectangles before after moving them.

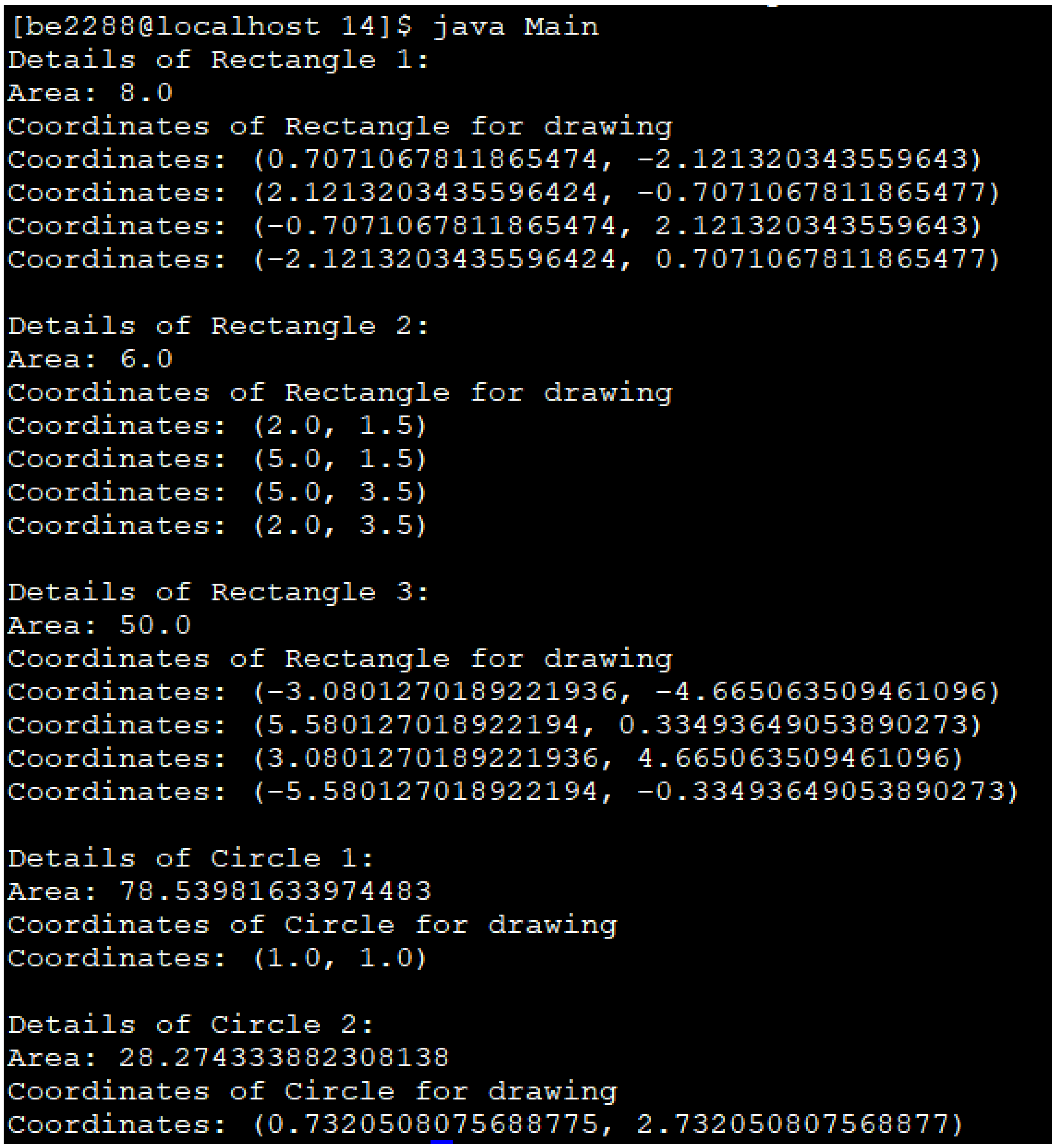
```
import java.util.*;
class Coordinate {
  public double x, y;
  Coordinate(double x, double y) {
    this.x = x;
    this.y = y;
  }
  Coordinate() {
    this(0.0, 0.0);
  }
  public void rotate(double angleDegree) {
    double angleRadian = Math.toRadians(angleDegree);
    double[][] matx = {{Math.cos(angleRadian), -Math.sin(angleRadian)}, {Math.sin(angleRadian),
Math.cos(angleRadian)}};
    double[] newCoords = new double[2];
    double[] oldCoords = {x, y};
    double temp = 0;
    for (int row = 0; row < 2; row++) {
      temp = 0;
      for (int col = 0; col < 2; col++) {
        temp = temp + matx[row][col] * oldCoords[col];
      }
      newCoords[row] = temp;
    }
    x = newCoords[0];
    y = newCoords[1];
  }
  public void move(double shiftXBy, double shiftYBy) {
    x = x + shiftXBy;
    y = y + shiftYBy;
  }
  public void display() {
    System.out.println("Coordinates: (" + x + ", " + y + ")");
  }
  public double distanceFrom(Coordinate other) {
    return Math.sqrt(Math.pow(this.x - other.x, 2) + Math.pow(this.y - other.y, 2));
  }
}
```

```
interface Shape {
   double area();
   void draw();
   void rotate(double angleDegree);
   void move(double x, double y);
}

class Circle implements Shape {
   double radius;
   Coordinate center;
   Circle(double xCenter, double yCenter, double radius) {
      center = new Coordinate(xCenter, yCenter);
      if (radius > 0) {
         this.radius = radius;
      } else {
         this.radius = -radius;
      }
   }
   Circle(double radius) {
      this(0.0, 0.0, radius);
   }
   public double area() {
      return Math.PI * radius * radius;
   }
   public void draw() {
      System.out.println("Coordinates of Circle for drawing");
      center.display();
   }
   public void rotate(double angleDegree) {
      center.rotate(angleDegree);
   }
   public void move(double shiftXBy, double shiftYBy) {
      center.move(shiftXBy, shiftYBy);
   }
}
```

```
class Rectangle implements Shape {
  Coordinate[] points = new Coordinate[4];

  Rectangle(double xCenter, double yCenter, double length, double breadth) {
    if (length > 0 && breadth > 0) {
      double halfLength = length / 2;
      double halfBreadth = breadth / 2;

      points[0] = new Coordinate(xCenter - halfLength, yCenter - halfBreadth);
      points[1] = new Coordinate(xCenter + halfLength, yCenter - halfBreadth);
      points[2] = new Coordinate(xCenter + halfLength, yCenter + halfBreadth);
      points[3] = new Coordinate(xCenter - halfLength, yCenter + halfBreadth);
    } else {
      for (int i = 0; i < 4; i++) {
        points[i] = new Coordinate();
      }
    }
  }

  Rectangle(double length, double breadth) {
    this(0.0, 0.0, length, breadth);
  }
  public double area() {
    double length = points[0].distanceFrom(points[1]);
    double width = points[1].distanceFrom(points[2]);

    return length * width;
  }

  public void draw() {
    System.out.println("Coordinates of Rectangle for drawing");
    for (Coordinate point : points) {
      point.display();
    }
  }
```

```
    public void rotate(double angleDegree) {
      double centerx = 0;
      double centery = 0;

      for (Coordinate point : points) {
        centerx = centerx + point.x;
        centery = centery + point.y;
      }
      centerx = centerx /4;
      centery = centery/4;

      for (Coordinate point : points) {
        point.move(-centerx,-centery);
      }
      for (Coordinate point : points) {
         point.rotate(angleDegree);
      }
      for (Coordinate point : points) {
         point.move(centerx,centery);
      }
    }

    public void move(double shiftXBy, double shiftYBy) {
        for (Coordinate point : points) {
            point.move(shiftXBy, shiftYBy);
        }
    }
}
```

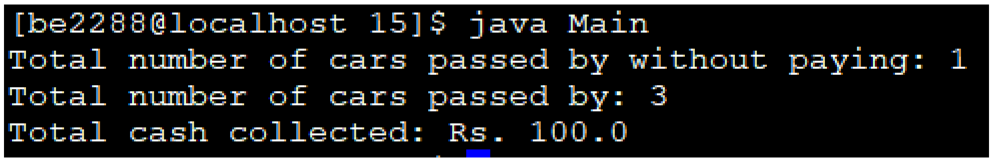
```java
class Main {
    public static void main(String[] args) {
        Rectangle rectangle1 = new Rectangle(2.0, 4.0);
        Rectangle rectangle2 = new Rectangle(1.5, 1.5, 3.0, 2.0);
        Rectangle rectangle3 = new Rectangle(1.0,1.0,10.0, 5.0);
        Circle circle1 = new Circle(5.0);
        Circle circle2 = new Circle(2.0, 2.0, 3.0);

        rectangle1.rotate(45.0);
        rectangle2.move(2.0, 1.0);
        rectangle3.rotate(30.0);
        rectangle3.move(-1.0,-1.0);

        circle2.rotate(30.0);
        circle1.move(1.0, 1.0);

        System.out.println("Details of Rectangle 1:");
        System.out.println("Area: " + rectangle1.area());
        rectangle1.draw();
        System.out.println("\nDetails of Rectangle 2:");
        System.out.println("Area: " + rectangle2.area());
        rectangle2.draw();
        System.out.println("\nDetails of Rectangle 3:");
        System.out.println("Area: " + rectangle3.area());
        rectangle3.draw();


        System.out.println("\nDetails of Circle 1:");
        System.out.println("Area: " + circle1.area());
        circle1.draw();
        System.out.println("\nDetails of Circle 2:");
        System.out.println("Area: " + circle2.area());
        circle2.draw();
    }
}
```



```
[be2288@localhost 14]$ java Main
Details of Rectangle 1:
Area: 8.0
Coordinates of Rectangle for drawing
Coordinates: (0.7071067811865474, -2.121320343559643)
Coordinates: (2.1213203435596424, -0.7071067811865477)
Coordinates: (-0.7071067811865474, 2.121320343559643)
Coordinates: (-2.1213203435596424, 0.7071067811865477)

Details of Rectangle 2:
Area: 6.0
Coordinates of Rectangle for drawing
Coordinates: (2.0, 1.5)
Coordinates: (5.0, 1.5)
Coordinates: (5.0, 3.5)
Coordinates: (2.0, 3.5)

Details of Rectangle 3:
Area: 50.0
Coordinates of Rectangle for drawing
Coordinates: (-3.0801270189221936, -4.665063509461096)
Coordinates: (5.580127018922194, 0.33493649053890273)
Coordinates: (3.0801270189221936, 4.665063509461096)
Coordinates: (-5.580127018922194, -0.33493649053890273)

Details of Circle 1:
Area: 78.53981633974483
Coordinates of Circle for drawing
Coordinates: (1.0, 1.0)

Details of Circle 2:
Area: 28.274333882308138
Coordinates of Circle for drawing
Coordinates: (0.7320508075688775, 2.732050807568877)
```

15. Imagine a toll booth and a bridge. Cars passing by the booth are expected to pay an amount of Rs.50/- as toll tax. Mostly they do but sometimes a car goes by without paying. The toll booth keeps track of the number of the cars that have passed without paying, total number of cars passed by, and the total amount of money collected. Execute this with a class called “Tollbooth” and print out the result as follows: The total number of cars passed by without paying. Total number of cars passed by. Total cash collected

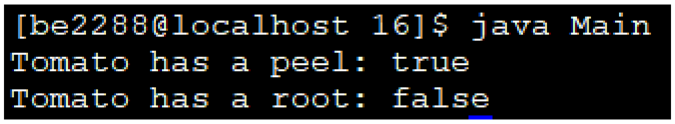
```
class Tollbooth {
    private int carsWithoutPayment;
    private int totalCarsPassed;
    private double totalCashCollected;

    Tollbooth() {
        carsWithoutPayment = 0;
        totalCarsPassed = 0;
        totalCashCollected = 0;
    }
    public void carPasses(boolean paid) {
        totalCarsPassed++;

        if (!paid) {
            carsWithoutPayment++;
        } else {
            totalCashCollected =totalCashCollected+ 50.0;
        }
    }
    public void displayResults() {
        System.out.println("Total number of cars passed by without paying: " + carsWithoutPayment);
        System.out.println("Total number of cars passed by: " + totalCarsPassed);
        System.out.println("Total cash collected: Rs. " + totalCashCollected);
    }
}

class Main{
    public static void main(String[] args) {
        Tollbooth tollbooth = new Tollbooth();
        tollbooth.carPasses(true);
        tollbooth.carPasses(false);
        tollbooth.carPasses(true);
        tollbooth.displayResults();
    }
}
```



```
[be2288@localhost 15]$ java Main
Total number of cars passed by without paying: 1
Total number of cars passed by: 3
Total cash collected: Rs. 100.0
```

16. Write two interfaces “Fruit” and “Vegetable” containing methods ‘hasAPeel’ and ‘hasARoot’. Now write a class “Tomato” implementing Fruit and Vegetable. Instantiate an object of Tomato Print the details of this object.

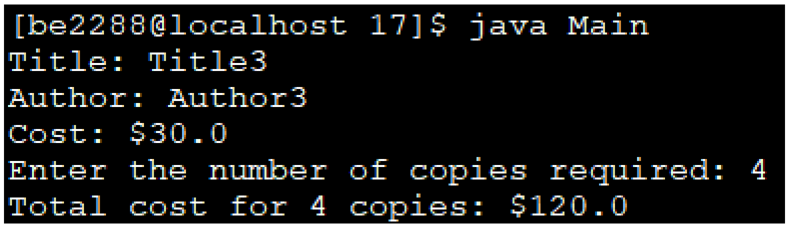
```
interface Fruit{
  boolean hasAPeel();
  boolean hasARoot();
}

interface Vegetable{
  boolean hasAPeel();
  boolean hasARoot();
}

class Tomato implements Fruit, Vegetable{
  public boolean hasARoot(){
    return false;
  }
  public boolean hasAPeel(){
    return true;
  }
}

class Main {
  public static void main(String[] args) {
    Tomato tomato = new Tomato();
    System.out.println("Tomato has a peel: " + tomato.hasAPeel());
    System.out.println("Tomato has a root: " + tomato.hasARoot());
    }
}
```



```
[be2288@localhost 16]$ java Main
Tomato has a peel: true
Tomato has a root: false
```

17. A bookshop maintains the inventory of books that are being sold at the shop. The list includes details such as author, title, publisher, cost and stock position. Whenever a customer wants a book, the sales person inputs the title and author and the system searches the list and displays whether it is available or not. If it is not, an appropriate message is displayed. If it is, then the system displays the book details and details and requests for the number of copies required. If the required copies are available, the total cost of the requested copies is displayed, otherwise the message “requires copies not in stock” is displayed. Design a system using a class called “Book” with suitable member methods and constructors.

```
import java.util.HashMap;
import java.util.Scanner;
class Book {
    String author;
    String title;
    float cost;
    public Book(String author, String title, float cost) {
        this.author = author;
        this.title = title;
        this.cost = cost;
    }
    void display() {
        System.out.println("Title: " + title);
        System.out.println("Author: " + author);
        System.out.println("Cost: $" + cost);
    }
}
class BookStore {
    HashMap<String, Integer> inventory;
    HashMap<String, Book> bookDetails;
    BookStore() {
        inventory = new HashMap<>();
        bookDetails = new HashMap<>();
    }
    void updateInventory(Book book, int newInventoryCount) {
        String key = book.title + " BY : " + book.author;
        inventory.put(key, newInventoryCount);
        bookDetails.put(key, book);
    }
    void searchInventory(String title, String author) {
        String key = title + " BY : " + author;
        if (bookDetails.containsKey(key)) {
            Book book = bookDetails.get(key);
            book.display();
            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter the number of copies required: ");
            int requiredCopies = scanner.nextInt();
            if (requiredCopies <= inventory.get(key)) {
                float totalCost = requiredCopies * book.cost;
                System.out.println("Total cost for " + requiredCopies + " copies: $" + totalCost);
            } else {
                System.out.println("Required amount not available.");
            }
        } else {
            System.out.println("Book unavailable.");
        }
    }
}
class Main {
    public static void main(String[] args) {
        Book book1 = new Book("Author1", "Title1", 20.0f);
        Book book2 = new Book("Author2", "Title2", 25.0f);
        Book book3 = new Book("Author3", "Title3", 30.0f);
        Book book4 = new Book("Author4", "Title4", 15.0f);
        Book book5 = new Book("Author5", "Title5", 18.0f);
        BookStore bookstore = new BookStore();
        bookstore.updateInventory(book1, 10);
        bookstore.updateInventory(book2, 15);
        bookstore.updateInventory(book3, 8);
        bookstore.updateInventory(book4, 20);
        bookstore.updateInventory(book5, 12);
        bookstore.searchInventory("Title3", "Author3");
    }
}
```



```
[be2288@localhost 17]$ java Main
Title: Title3
Author: Author3
Cost: $30.0
Enter the number of copies required: 4
Total cost for 4 copies: $120.0
```