

OOS LAB ASSIGNMENT - 1



Name - Anuska Nath
**Department - Information
Technology**
Roll No - 002311001003
Section - A1

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.

Source code :

```
class Room
{
    int height, width, breadth;
    Room(int h, int w, int b){
        height=h;
        width=w;
        breadth=b;
    }

    int volume(){
        return (height*breadth*width);
    }
}

class RoomDemo
{
    public static void main(String args[])
    {
        Room R1=new Room(1,2,3);
        Room R2=new Room(3,4,5);
        System.out.println("Volume of room 1 is: " + R1.volume());
        System.out.println("Volume of room 2 is: " + R2.volume());
    }
}
```

Output:

```
[be2303@localhost a1]$ javac q1.java
[be2303@localhost a1]$ java RoomDemo
Volume of room 1 is: 6
Volume of room 2 is: 60
```

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.

Source Code:

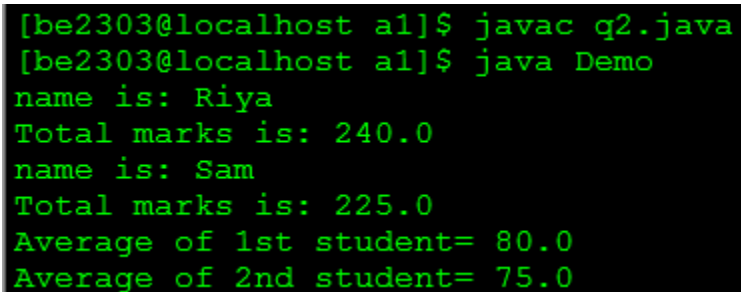
```
class student
{
    String name;
    double s1m;
    double s2m;
    double s3m;

    void set(String s, double m1, double m2, double m3){
        name=s;
        s1m=m1;
        s2m=m2;
        s3m=m3;
    }

    double avg(){
        return ((s1m+s2m+s3m)/3);
    }

    void disp(){
        System.out.println("name is: " + name);
        System.out.println("Total marks is: " + (s1m+s2m+s3m));
    }
}

class Demo
{
    public static void main(String args[])
    {
        student ob1=new student();
        ob1.set("Riya",70,80,90);
        student ob2=new student();
        ob2.set("Sam",65,85,75);
        ob1.disp();
        ob2.disp();
        System.out.println("Average of 1st student= " + ob1.avg());
        System.out.println("Average of 2nd student= " + ob2.avg());
    }
}
```

Output :

```
[be2303@localhost a1]$ javac q2.java
[be2303@localhost a1]$ java Demo
name is: Riya
Total marks is: 240.0
name is: Sam
Total marks is: 225.0
Average of 1st student= 80.0
Average of 2nd student= 75.0
```

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.

Source code :

```
class Stack
{
    int size;
    int arr[];
    int top;

    Stack(int s){
        size=s;
        arr = new int[size];
        top=-1;
    }

    void push(int n){
        if (top==size-1){
            System.out.println("Stack Overflow");
            return;
        }
        top++;
        arr[top]=n;
    }

    int pop(){
        if (top==-1){
            System.out.println("Stack Underflow");
            return -1;
        }
        int v=arr[top];
        top--;
        return v;
    }

    void print(){
        if(top==-1){
            System.out.println("Stack empty");
            return;
        }
        System.out.println("The elements of the stack are : ");
        for (int i=top; i>-1; i--){
```

```

        System.out.println(arr[i]);
    }
}

class Demo{
    public static void main(String args[])
    {
        Stack s1=new Stack(30);

        s1.push(10);
        s1.push(20);
        s1.push(30);
        s1.push(15);
        s1.push(9);

        s1.print();

        System.out.println(s1.pop() + " popped");
        System.out.println(s1.pop() + " popped");
        System.out.println(s1.pop() + " popped");

        s1.print();
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q3.java
[be2303@localhost a1]$ java Demo
The elements of the stack are :
9
15
30
20
10
9 popped
15 popped
30 popped
The elements of the stack are :
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.

Source code :

```
class BankAccount
{
    int AccountNumber;
    double balance;
    String name;

    BankAccount(){
        AccountNumber=0;
        balance=0;
        name=" ";
    }

    BankAccount(int n, String s, double b){
        AccountNumber=n;
        balance=b;
        name=s;
    }

    void balance(){
        System.out.println("balance is: " + balance);
    }

    void add(double n){
        balance+=n;
    }

    void subtract(double n){
        balance-=n;
    }

    void disp(){
        System.out.println("Account Number: " + AccountNumber);
        System.out.println("Name of Account holder: " + name);
        System.out.println("Balance is: " + balance);
    }
}
```

```

class AccountManager
{
    int n;
    BankAccount Accts[];

    AccountManager(int x){
        n=x;
        Accts = new BankAccount[x];
    }

    int i=0;

    void create(int N, String s, double b){
        if (i>=n){
            System.out.println("Full");
            return;
        }
        Accts[i++] = new BankAccount(N,s,b);
    }

    void delete(int N){
        for (int j=0;j<n;j++){
            if (Accts[j].AccountNumber==N){
                Accts[j]=null;
                return;
            }
        }
        System.out.println("No account with this account number");
    }

    void deposit(int N, int b){
        for (int j=0;j<n;j++){
            if (Accts[j].AccountNumber==N){
                Accts[j].add(b);
                System.out.println("Deposited");
                return;
            }
        }
        System.out.println("No account with this account number");
    }

    void withdraw(int N, int b){
        for (int j=0;j<n;j++){
            if (Accts[j].AccountNumber==N){
                Accts[j].subtract(b);
            }
        }
    }
}

```

```

        System.out.println("Withdrawn");
        return;
    }
}
System.out.println("No account with this account number");
}

void show(){
    for (int j=0;j<n;j++){
        Accts[j].disp();
    }
}
}

class Bank
{
    public static void main(String args[])
    {
        AccountManager am = new AccountManager(5);

        am.create(1001,"Riya",6537);
        am.create(1002,"Sam",5462);
        am.create(1003,"Mani",6378);
        am.create(1004,"Raj",4342);
        am.create(1005,"Abhi",56234);

        am.show();
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q4.java
[be2303@localhost a1]$ java Bank
Account Number: 1001
Name of Account holder: Riya
Balance is: 6537.0
Account Number: 1002
Name of Account holder: Sam
Balance is: 5462.0
Account Number: 1003
Name of Account holder: Mani
Balance is: 6378.0
Account Number: 1004
Name of Account holder: Raj
Balance is: 4342.0
Account Number: 1005
Name of Account holder: Abhi
Balance is: 56234.0

```


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.

Source Code :

```
class Complex
{
    int real;
    int imag;

    Complex(){
        real=0;
        imag=0;
    }

    Complex(int r){
        real=r;
    }

    Complex(int r, int i){
        real=r;
        imag=i;
    }

    void add(Complex ob){
        int r=real+ob.real;
        int i=imag+ob.imag;
        System.out.println("The sum is: " + r+" + "+i+"i");
    }

    void product(Complex ob){
        int r=real*ob.real;
        int a=r-imag*ob.imag;
        int i=(real*ob.imag)+(imag*ob.real);
        System.out.println("The product is: " + a+" + "+i+"i");
    }
}
```

```

class Demo{
    public static void main(String args[]){
        Complex c1= new Complex(3,2);
        Complex c2= new Complex(4,-2);
        c1.add(c2);
        c1.product(c2);
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q5.java
[be2303@localhost a1]$ java Demo
The sum is: 7 + 0i
The product is: 16 + 2i

```

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.

Source code :

```

lass Employee
{
    String name;
    int ID;
    String Address;
    int salary;

    Employee(){
        name=" ";
        ID=0;
        Address=" ";
        salary=0;
    }

    Employee(String n, int i, String a, int s){
        name=n;
        ID=i;
        Address=a;
        salary=s;
    }
}

```

```

        void read(){
            System.out.println("Name is: " + name);
            System.out.println("ID is: " + ID);
            System.out.println("Address is: " + Address);
            System.out.println("Salary is: " + salary);
        }
    }

class Dept
{
    String dname;
    Employee emps[];
    int n;

    Dept(String s, int x){
        dname=s;
        n=x;
        emps = new Employee[n];
    }

    int i=0;

    void add(String n, int d, String a, int s){
        emps[i++] = new Employee(n,d,a,s);
    }

    void remove(int d){
        for (int j=0; j<n; j++){
            if (emps[j].ID==d){
                emps[j]=null;
                return;
            }
        }
        System.out.println("No employee with such ID");
    }

    int sum(){
        int su=0;
        for (int j=0;j<n;j++){
            emps[j].read();
            su+=emps[j].salary;
        }
        return su;
    }
}

```

```

}

class Demo
{
    public static void main(String args[]){
        Dept ob = new Dept("Information technology",5);

        ob.add("Riya",101,"Kolkata",1000);
        ob.add("Keya",102,"Birbhum",3500);
        ob.add("Raj",103,"Bangalore",7000);
        ob.add("Tina",104,"Delhi",5000);
        ob.add("Karan",105,"Mumbai",4000);

        int s = ob.sum();

        System.out.println("The yearly expenditure of the IT department is: " + (s*13));
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q6.java
[be2303@localhost a1]$ java Demo
Name is: Riya
ID is: 101
Address is: Kolkata
Salary is: 1000
Name is: Keya
ID is: 102
Address is: Birbhum
Salary is: 3500
Name is: Raj
ID is: 103
Address is: Bangalore
Salary is: 7000
Name is: Tina
ID is: 104
Address is: Delhi
Salary is: 5000
Name is: Karan
ID is: 105
Address is: Mumbai
Salary is: 4000
The yearly expenditure of the IT department is: 266500

```

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.

Source code :

```
import java.util.ArrayList;
import java.util.List;

abstract class Publication {
    private int noOfPages;
    private double price;
    private String publisherName;

    public Publication(int noOfPages, double price, String publisherName) {
        this.noOfPages = noOfPages;
        this.price = price;
        this.publisherName = publisherName;
    }

    public int getNoOfPages() {
        return noOfPages;
    }

    public void setNoOfPages(int noOfPages) {
        this.noOfPages = noOfPages;
    }

    public double getPrice() {
        return price;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    public String getPublisherName() {
        return publisherName;
    }

    public void setPublisherName(String publisherName) {
        this.publisherName = publisherName;
    }
}
```

```
    public abstract void printDetails();  
}
```

```
class Book extends Publication {  
    private String author;  
  
    public Book(int noOfPages, double price, String publisherName, String author) {  
        super(noOfPages, price, publisherName);  
        this.author = author;  
    }  
  
    public String getAuthor() {  
        return author;  
    }  
  
    public void setAuthor(String author) {  
        this.author = author;  
    }  
  
    public void printDetails() {  
        System.out.println("Book:");  
        System.out.println(" Pages: " + getNoOfPages());  
        System.out.println(" Price: $" + getPrice());  
        System.out.println(" Publisher: " + getPublisherName());  
        System.out.println(" Author: " + author);  
    }  
}
```

```
class Journal extends Publication {  
    private String issue;  
  
    public Journal(int noOfPages, double price, String publisherName, String issue) {  
        super(noOfPages, price, publisherName);  
        this.issue = issue;  
    }  
  
    public String getIssue() {  
        return issue;  
    }  
  
    public void setIssue(String issue) {  
        this.issue = issue;  
    }  
  
    public void printDetails() {
```

```

        System.out.println("Journal:");
        System.out.println(" Pages: " + getNoOfPages());
        System.out.println(" Price: $" + getPrice());
        System.out.println(" Publisher: " + getPublisherName());
        System.out.println(" Issue: " + issue);
    }
}

```

```

class Library {
    private List<Publication> publications;

    public Library() {
        publications = new ArrayList<>();
    }

    public void addPublication(Publication publication) {
        publications.add(publication);
    }

    public void printAllPublications() {
        for (Publication publication : publications) {
            publication.printDetails();
            System.out.println();
        }
    }
}

```

```

class Demo {
    public static void main(String[] args) {
        Library library = new Library();

        Book book1 = new Book(300, 19.99, "Penguin Books", "Author A");
        Book book2 = new Book(450, 25.50, "HarperCollins", "Author B");
        Book book3 = new Book(150, 10.75, "Random House", "Author C");

        Journal journal1 = new Journal(100, 5.99, "Science Direct", "Issue 1");
        Journal journal2 = new Journal(200, 7.99, "Nature", "Issue 2");

        library.addPublication(book1);
        library.addPublication(book2);
        library.addPublication(book3);
        library.addPublication(journal1);
        library.addPublication(journal2);

        library.printAllPublications();
    }
}

```

```
}  
}
```

Output :

```
[be2303@localhost a1]$ javac q7.java  
[be2303@localhost a1]$ java Demo  
Book:  
  Pages: 300  
  Price: $19.99  
  Publisher: Penguin Books  
  Author: Author A  
  
Book:  
  Pages: 450  
  Price: $25.5  
  Publisher: HarperCollins  
  Author: Author B  
  
Book:  
  Pages: 150  
  Price: $10.75  
  Publisher: Random House  
  Author: Author C  
  
Journal:  
  Pages: 100  
  Price: $5.99  
  Publisher: Science Direct  
  Issue: Issue 1  
  
Journal:  
  Pages: 200  
  Price: $7.99  
  Publisher: Nature  
  Issue: Issue 2
```

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.

Source Code :


```
import java.util.ArrayList;
import java.util.List;

class Account {
    private int accountNumber;
    private String holderName;
    private double balance;

    public Account(int accountNumber, String holderName, double balance) {
        this.accountNumber = accountNumber;
        this.holderName = holderName;
        this.balance = balance;
    }

    public int getAccountNumber() {
        return accountNumber;
    }

    public void setAccountNumber(int accountNumber) {
        this.accountNumber = accountNumber;
    }

    public String getHolderName() {
        return holderName;
    }

    public void setHolderName(String holderName) {
        this.holderName = holderName;
    }

    public double getBalance() {
        return balance;
    }

    public void setBalance(double balance) {
        this.balance = balance;
    }

    public void printDetails() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Holder Name: " + holderName);
        System.out.println("Balance: $" + balance);
    }
}
```

```

class SavingsAccount extends Account {
    private double interestRate;
    public SavingsAccount(int accountNumber, String holderName, double balance,
double interestRate) {
        super(accountNumber, holderName, balance);
        this.interestRate = interestRate;
    }

    public double getInterestRate() {
        return interestRate;
    }

    public void setInterestRate(double interestRate) {
        this.interestRate = interestRate;
    }

    public double calculateYearlyInterest() {
        return getBalance() * (interestRate / 100);
    }

    public void printDetails() {
        super.printDetails();
        System.out.println("Account Type: Savings Account");
        System.out.println("Interest Rate: " + interestRate + "%");
        System.out.println("Yearly Interest: $" + calculateYearlyInterest());
    }
}

class CurrentAccount extends Account {
    public CurrentAccount(int accountNumber, String holderName, double balance) {
        super(accountNumber, holderName, balance);
    }

    public void printDetails() {
        super.printDetails();
        System.out.println("Account Type: Current Account");
    }
}

class Manager {
    private List<Account> accounts;

    public Manager() {
        accounts = new ArrayList<>();
    }
}

```

```

    public void addAccount(Account account) {
        accounts.add(account);
    }

    public void printAllAccounts() {
        for (Account account : accounts) {
            account.printDetails();
            System.out.println();
        }
    }
}

class Demo {
    public static void main(String[] args) {
        Manager manager = new Manager();

        SavingsAccount savings1 = new SavingsAccount(1001, "John Doe", 5000.0, 2.5);
        SavingsAccount savings2 = new SavingsAccount(1002, "Jane Smith", 8000.0, 3.0);

        CurrentAccount current1 = new CurrentAccount(2001, "Alice Johnson", 10000.0);
        CurrentAccount current2 = new CurrentAccount(2002, "Bob Brown", 12000.0);
        CurrentAccount current3 = new CurrentAccount(2003, "Eve Lee", 15000.0);

        manager.addAccount(savings1);
        manager.addAccount(savings2);
        manager.addAccount(current1);
        manager.addAccount(current2);
        manager.addAccount(current3);

        System.out.println("Details of All Accounts:");
        System.out.println("-----");
        manager.printAllAccounts();
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q8.java
[be2303@localhost a1]$ java Demo
Details of All Accounts:
-----
Account Number: 1001
Holder Name: John Doe
Balance: $5000.0
Account Type: Savings Account
Interest Rate: 2.5%
Yearly Interest: $125.0

Account Number: 1002
Holder Name: Jane Smith
Balance: $8000.0
Account Type: Savings Account

```

```
Account Type: Savings Account
Interest Rate: 3.0%
Yearly Interest: $240.0

Account Number: 2001
Holder Name: Alice Johnson
Balance: $10000.0
Account Type: Current Account

Account Number: 2002
Holder Name: Bob Brown
Balance: $12000.0
Account Type: Current Account

Account Number: 2003
Holder Name: Eve Lee
Balance: $15000.0
Account Type: Current Account
```

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.

Source code :

```
import java.util.ArrayList;
import java.util.List;

class Person {
    private int age;
    private double weight;
    private double height;
    private String dateOfBirth;
    private String address;

    public Person(int age, double weight, double height, String dateOfBirth, String
address) {
        this.age = age;
        this.weight = weight;
        this.height = height;
        this.dateOfBirth = dateOfBirth;
        this.address = address;
    }
}
```

```
}

public int getAge() {
    return age;
}

public void setAge(int age) {
    this.age = age;
}

public double getWeight() {
    return weight;
}

public void setWeight(double weight) {
    this.weight = weight;
}

public double getHeight() {
    return height;
}

public void setHeight(double height) {
    this.height = height;
}

public String getDateOfBirth() {
    return dateOfBirth;
}

public void setDateOfBirth(String dateOfBirth) {
    this.dateOfBirth = dateOfBirth;
}

public String getAddress() {
    return address;
}

public void setAddress(String address) {
    this.address = address;
}

public void printDetails() {
    System.out.println("Age: " + age);
    System.out.println("Weight: " + weight);
}
```

```
        System.out.println("Height: " + height);
        System.out.println("Date of Birth: " + dateOfBirth);
        System.out.println("Address: " + address);
    }
}
```

```
class Employee extends Person {
    private double salary;
    private String dateOfJoining;
    private int experience;

    public Employee(int age, double weight, double height, String dateOfBirth, String
address,
                    double salary, String dateOfJoining, int experience) {
        super(age, weight, height, dateOfBirth, address);
        this.salary = salary;
        this.dateOfJoining = dateOfJoining;
        this.experience = experience;
    }

    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }

    public String getDateOfJoining() {
        return dateOfJoining;
    }

    public void setDateOfJoining(String dateOfJoining) {
        this.dateOfJoining = dateOfJoining;
    }

    public int getExperience() {
        return experience;
    }

    public void setExperience(int experience) {
        this.experience = experience;
    }

    public void printDetails() {
```

```

        super.printDetails();
        System.out.println("Salary: " + salary);
        System.out.println("Date of Joining: " + dateOfJoining);
        System.out.println("Experience: " + experience + " years");
    }
}

```

```

class Student extends Person {
    private int roll;
    private List<String> listOfSubjects;
    private List<Double> marks;

    public Student(int age, double weight, double height, String dateOfBirth, String
address, int roll,
                    List<String> listOfSubjects, List<Double> marks) {
        super(age, weight, height, dateOfBirth, address);
        this.roll = roll;
        this.listOfSubjects = listOfSubjects;
        this.marks = marks;
    }

```

```

    public double calculateGrade() {
        double total = 0;
        for (double mark : marks) {
            total += mark;
        }
        return total / marks.size();
    }

```

```

    public void printDetails() {
        super.printDetails();
        System.out.println("Roll: " + roll);
        System.out.println("Subjects: " + listOfSubjects);
        System.out.println("Marks: " + marks);
        System.out.println("Grade: " + calculateGrade());
    }
}

```

```

class Technician extends Employee {
    public Technician(int age, double weight, double height, String dateOfBirth, String
address,
                    double salary, String dateOfJoining, int experience) {
        super(age, weight, height, dateOfBirth, address, salary, dateOfJoining, experience);
    }

```

```

        public void printDetails() {
            super.printDetails();
            System.out.println("Role: Technician");
        }
    }

    class Professor extends Employee {
        private List<String> courses;
        private List<String> listOfAdvisees;

        public Professor(int age, double weight, double height, String dateOfBirth, String
address,
                        double salary, String dateOfJoining, int experience, List<String> courses,
                        List<String> listOfAdvisees) {
            super(age, weight, height, dateOfBirth, address, salary, dateOfJoining, experience);
            this.courses = courses;
            this.listOfAdvisees = listOfAdvisees;
        }

        public void addAdvisee(String advisee) {
            listOfAdvisees.add(advisee);
        }

        public void removeAdvisee(String advisee) {
            listOfAdvisees.remove(advisee);
        }

        public void printDetails() {
            super.printDetails();
            System.out.println("Courses: " + courses);
            System.out.println("Advisees: " + listOfAdvisees);
        }
    }

    class Demo {
        public static void main(String[] args) {
            Person person = new Person(30, 70, 175, "1993-05-15", "123 Main St");

            Employee employee = new Employee(40, 80, 180, "1983-09-20", "456 Elm St",
50000, "2010-06-01", 13);

            List<String> subjects = new ArrayList<>();
            subjects.add("Math");
            subjects.add("Physics");
            List<Double> marks = new ArrayList<>();

```



```

marks.add(85.0);
marks.add(90.0);
Student student = new Student(20, 60, 165, "2003-11-15", "789 Maple St", 101,
subjects, marks);

Technician technician = new Technician(35, 75, 170, "1988-02-14", "321 Oak St",
40000, "2015-03-12", 8);

List<String> courses = new ArrayList<>();
courses.add("Computer Science");
courses.add("AI");
List<String> advisees = new ArrayList<>();
advisees.add("John");
advisees.add("Jane");
Professor professor = new Professor(50, 85, 180, "1973-12-25", "654 Pine St",
70000, "2000-08-15", 23, courses, advisees);

System.out.println("Person Details:");
person.printDetails();
System.out.println();

System.out.println("Employee Details:");
employee.printDetails();
System.out.println();

System.out.println("Student Details:");
student.printDetails();
System.out.println();

System.out.println("Technician Details:");
technician.printDetails();
System.out.println();

System.out.println("Professor Details:");
professor.printDetails();
}
}

```

Output :

```

[be2303@localhost a1]$ javac q9.java
[be2303@localhost a1]$ java Demo
Person Details:
Age: 30
Weight: 70.0
Height: 175.0
Date of Birth: 1993-05-15
Address: 123 Main St

```

Employee Details:

Age: 40
Weight: 80.0
Height: 180.0
Date of Birth: 1983-09-20
Address: 456 Elm St
Salary: 50000.0
Date of Joining: 2010-06-01
Experience: 13 years

Student Details:

Age: 20
Weight: 60.0
Height: 165.0
Date of Birth: 2003-11-15
Address: 789 Maple St
Roll: 101
Subjects: [Math, Physics]
Marks: [85.0, 90.0]
Grade: 87.5

Technician Details:

Age: 35
Weight: 75.0
Height: 170.0
Date of Birth: 1988-02-14
Address: 321 Oak St
Salary: 40000.0
Date of Joining: 2015-03-12
Experience: 8 years
Role: Technician

Professor Details:

Age: 50
Weight: 85.0
Height: 180.0
Date of Birth: 1973-12-25
Address: 654 Pine St
Salary: 70000.0
Date of Joining: 2000-08-15
Experience: 23 years
Courses: [Computer Science, AI]
Advisees: [John, Jane]

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.

Source code :

```
import java.util.*;
```

```
class Book {  
    String author;  
    String title;  
    String publisher;  
    double cost;  
    int stock;
```

```
    Book(String title, String author, String publisher, double cost, int stock) {  
        this.title = title;  
        this.author = author;  
        this.publisher = publisher;  
        this.cost = cost;  
        this.stock = stock;  
    }
```

```
    boolean matches(String title, String author) {  
        return (this.title.equalsIgnoreCase(title) && (this.author).equalsIgnoreCase(author));  
    }
```

```
    void displayDetails() {  
        System.out.println("Title: " + title);  
        System.out.println("Author: " + author);  
        System.out.println("Publisher: " + publisher);  
        System.out.println("Cost: " + cost);  
        System.out.println("Stock: " + stock);  
    }
```

```
    void processPurchase(int requestedCopies) {
```

```

        if (requestedCopies <= stock) {
            System.out.println("Total Cost: " + (cost * requestedCopies));
            stock -= requestedCopies;
        } else {
            System.out.println("Required copies not in stock");
        }
    }
}

class Demo {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Book[] inventory = {
            new Book("The Mountain is You", "David Goggins", "Bloomsbury", 500, 5),
            new Book("The Alchemist", "Paulo Coelho", "HarperCollins", 300, 10),
            new Book("Zero to One", "J.R.R. Tolkien", "Allen & Unwin", 400, 8),
            new Book("Atomic Habits", "James Clear", "Secker & Warburg", 350, 6)
        };
        boolean found = false;

        String title, author;
        System.out.print("Enter the title: ");
        title = sc.nextLine();
        System.out.print("Enter the author: ");
        author = sc.nextLine();

        for (Book book : inventory) {
            if (book.matches(title, author)) {
                found = true;
                System.out.println("Book is available. Details:");
                book.displayDetails();

                System.out.print("Enter the number of copies required: ");
                int requestedCopies = sc.nextInt();

                book.processPurchase(requestedCopies);
                break;
            }
        }
    }
}

```

```

        if (!found) {
            System.out.println("Book not available.");
        }
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q10.java
[be2303@localhost a1]$ java Demo
Enter the title: Book 3
Enter the author: author 3
Book is available. Details:
Title: Book 3
Author: Author 3
Publisher: Publisher 3
Cost: 400.0
Stock: 8
Enter the number of copies required: 3
Total Cost: 1200.0

```

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.

Source code :

```

class Date
{
    int day;
    int month;
    int year;

    Date(){
        day=1;
        month=1;
        year=1970;
    }

    Date(int d){
        day=d;
        month=1;
    }
}

```

```

        year=1970;
    }

    Date(int d, int m){
        day=d;
        month=m;
        year=1970;
    }

    Date(int d, int m, int y){
        day=d;
        month=m;
        year=y;
    }

    int leap[]={31,29,31,30,31,30,31,31,30,31,30,31};
    int nonleap[]={31,28,31,30,31,30,31,31,30,31,30,31};

    void tod(){
        System.out.println("Today date: " + day + "/" + month + "/" + year);
    }

    void prev(){
        if ((year%4==0 && year%100!=0)|| (year%400==0)){
            if (day==1 && month!=1){
                int d=leap[month-1];
                system.out.println("The previous date is: " + d + "/" + (month-1) + "/"
+ year);
            }
            else if (day==1 && month==1){
                int d=31;
                system.out.println("The previous date is: " + d + "/" + 12 + "/" +
(year-1));
            }
            else{
                int d=day-1;
                system.out.println("The previous date is: " + d + "/" + month + "/" +
year);
            }
        }
        else{
            if (day==1 && month!=1){
                int d=nonleap[month-1];

```

```

        system.out.println("The previous date is: " + d + "/" + (month-1) + "/"
+ year);
    }
    }
    else if (day==1 && month==1){
        int d=31;
        system.out.println("The previous date is: " + d + "/" + 12 + "/" +
(year-1));
    }
    else{
        int d=day-1;
        system.out.println("The previous date is: " + d + "/" + month + "/" +
year);
    }
}

void next(){
    if ((year%4==0 && year%100!=0)|| (year%400==0)){
        if (day==leap[month-1] && month!=12){
            system.out.println("The next date is: " + 1 + "/" + (month+1) + "/" +
year);
        }
        else if (day==nonleap[month-1] && month==12){
            system.out.println("The next date is: " + "1/1" + "/" + (year+1));
        }
        else{
            int d=day+1;
            system.out.println("The next date is: " + d + "/" + month + "/" + year);
        }
    }

    else{
        if (day==nonleap[month-1] && month!=12){
            system.out.println("The next date is: " + 1 + "/" + (month+1) + "/" +
year);
        }
        else if (day==nonleap[month-1] && month==12){
            system.out.println("The next date is: " + "1/1" + "/" + (year+1));
        }
        else{
            int d=day+1;
            system.out.println("The next date is: " + d + "/" + month + "/" + year);
        }
    }
}

```

```

        System.out.println(" ");
    }
}

class Demo{
    public static void main(String args[]){
        Date ob1=new Date();
        Date ob2=new Date(13,9,2004);
        Date ob3=new Date(28,2,2020);
        Date ob4=new Date(31,12,2023);

        ob1.tod();
        ob1.prev();
        ob1.next();
        ob2.tod();
        ob2.prev();
        ob2.next();
        ob3.tod();
        ob3.prev();
        ob3.next();
        ob4.tod();
        ob4.prev();
        ob4.next();
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q11.java
[be2303@localhost a1]$ java Demo
Todays date: 1/1/1970
The previous date is: 31/12/1969
The next date is: 2/1/1970

Todays date: 13/9/2004
The previous date is: 12/9/2004
The next date is: 14/9/2004

Todays date: 28/2/2020
The previous date is: 27/2/2020
The next date is: 29/2/2020

Todays date: 31/12/2023
The previous date is: 30/12/2023
The next date is: 1/1/2024

```


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.

Source code :

```
import java.util.Scanner;

class Student {
    private String name;
    private int rollNo;
    private String[] subjects;

    public Student(String name, int rollNo, String[] subjects) {
        this.name = name;
        this.rollNo = rollNo;
        this.subjects = subjects;
    }

    public String getName() {
        return name;
    }

    public int getRollNo() {
        return rollNo;
    }

    public String[] getSubjects() {
        return subjects;
    }
}

class TabulationSheet {
    private String subject;
    private int[] rollNos;
    private int[] marks;
    private int count;

    public TabulationSheet(String subject, int capacity) {
        this.subject = subject;
        rollNos = new int[capacity];
    }
}
```

```

        marks = new int[capacity];
        count = 0;
    }

    public void addMarks(int rollNo, int mark) {
        if (count < rollNos.length) {
            rollNos[count] = rollNo;
            marks[count] = mark;
            count++;
        }
    }

    public int getMark(int rollNo) {
        for (int i = 0; i < count; i++) {
            if (rollNos[i] == rollNo) {
                return marks[i];
            }
        }
        return -1;
    }

    public String getSubject() {
        return subject;
    }
}

class MarkSheet {
    private String studentName;
    private String[] subjects;
    private int[] marks;
    private int count;

    public MarkSheet(String studentName, int subjectCount) {
        this.studentName = studentName;
        subjects = new String[subjectCount];
        marks = new int[subjectCount];
        count = 0;
    }

    public void addMarks(String subject, int mark) {
        if (count < subjects.length) {
            subjects[count] = subject;
            marks[count] = mark;
            count++;
        }
    }
}

```

```

    }

    public void printMarkSheet() {
        System.out.println("Marksheet for: " + studentName);
        for (int i = 0; i < count; i++) {
            System.out.println(subjects[i] + ": " + marks[i]);
        }
        System.out.println();
    }
}

class Demo {
    public static void main(String[] args) {
        String[] subjects = {"Math", "Physics", "Chemistry", "Biology", "English"};

        Student[] students = {
            new Student("Alice", 101, subjects),
            new Student("Bob", 102, subjects),
            new Student("Charlie", 103, subjects)
        };

        TabulationSheet[] tabSheets = new TabulationSheet[subjects.length];
        for (int i = 0; i < subjects.length; i++) {
            tabSheets[i] = new TabulationSheet(subjects[i], students.length);
        }

        int[][] marksData = {
            {85, 90, 75},
            {80, 88, 70},
            {78, 84, 65},
            {92, 85, 80},
            {88, 79, 82}
        };

        for (int i = 0; i < subjects.length; i++) {
            for (int j = 0; j < students.length; j++) {
                tabSheets[i].addMarks(students[j].getRollNo(), marksData[i][j]);
            }
        }

        MarkSheet[] markSheets = new MarkSheet[students.length];
        for (int i = 0; i < students.length; i++) {
            markSheets[i] = new MarkSheet(students[i].getName(), subjects.length);
            for (TabulationSheet tabSheet : tabSheets) {

```

```

        markSheets[i].addMarks(tabSheet.getSubject(),
tabSheet.getMark(students[i].getRollNo()));
    }
}

for (MarkSheet markSheet : markSheets) {
    markSheet.printMarkSheet();
}
}
}

```

Output :

```

[be2303@localhost a1]$ javac q12.java
[be2303@localhost a1]$ java Demo
Marksheet for: Alice
Math: 85
Physics: 80
Chemistry: 78
Biology: 92
English: 88

Marksheet for: Bob
Math: 90
Physics: 88
Chemistry: 84
Biology: 85
English: 79

Marksheet for: Charlie
Math: 75
Physics: 70
Chemistry: 65
Biology: 80
English: 82

```

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.

Source code :

```

class Automobile {

```

```
private String make;
private String type;
private int maxSpeed;
private double price;
private double mileage;
private String registrationNumber;

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

public String getMake() {
    return make;
}

public void setMake(String make) {
    this.make = make;
}

public String getType() {
    return type;
}

public void setType(String type) {
    this.type = type;
}

public int getMaxSpeed() {
    return maxSpeed;
}

public void setMaxSpeed(int maxSpeed) {
    this.maxSpeed = maxSpeed;
}

public double getPrice() {
    return price;
}
```

```

    public void setPrice(double price) {
        this.price = price;
    }

    public double getMileage() {
        return mileage;
    }

    public void setMileage(double mileage) {
        this.mileage = mileage;
    }

    public String getRegistrationNumber() {
        return registrationNumber;
    }

    public void setRegistrationNumber(String registrationNumber) {
        this.registrationNumber = registrationNumber;
    }

    public void displayInfo() {
        System.out.println("Make: " + make);
        System.out.println("Type: " + type);
        System.out.println("Max Speed: " + maxSpeed);
        System.out.println("Price: " + price);
        System.out.println("Mileage: " + mileage);
        System.out.println("Registration Number: " + registrationNumber);
    }
}

class Track extends Automobile {
    private int capacity;
    private String hoodType;
    private int noOfWheels;

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

    public int getCapacity() {
        return capacity;
    }
}

```

```

    }

    public void setCapacity(int capacity) {
        this.capacity = capacity;
    }

    public String getHoodType() {
        return hoodType;
    }

    public void setHoodType(String hoodType) {
        this.hoodType = hoodType;
    }

    public int getNoOfWheels() {
        return noOfWheels;
    }

    public void setNoOfWheels(int noOfWheels) {
        this.noOfWheels = noOfWheels;
    }

    public void displayInfo() {
        super.displayInfo();
        System.out.println("Capacity: " + capacity);
        System.out.println("Hood Type: " + hoodType);
        System.out.println("Number of Wheels: " + noOfWheels);
    }
}

class Car extends Automobile {
    private int noOfDoors;
    private int seatingCapacity;

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

    public int getNoOfDoors() {
        return noOfDoors;
    }
}

```

```

    public void setNoOfDoors(int noOfDoors) {
        this.noOfDoors = noOfDoors;
    }

    public int getSeatingCapacity() {
        return seatingCapacity;
    }

    public void setSeatingCapacity(int seatingCapacity) {
        this.seatingCapacity = seatingCapacity;
    }

    public void displayInfo() {
        super.displayInfo();
        System.out.println("Number of Doors: " + noOfDoors);
        System.out.println("Seating Capacity: " + seatingCapacity);
    }
}

class Demo {
    public static void main(String args[]) {
        Track t = new Track("Volvo", "Heavy Duty", 120, 50000, 8, "TRK1234", 10000,
"Closed", 12);
        Car c = new Car("Toyota", "Sedan", 200, 30000, 15, "CAR5678", 4, 5);

        System.out.println("Track Details:");
        t.displayInfo();

        System.out.println("\nCar Details:");
        c.displayInfo();
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q13.java
[be2303@localhost a1]$ java Demo
Track Details:
Make: Volvo
Type: Heavy Duty
Max Speed: 120
Price: 50000.0
Mileage: 8.0
Registration Number: TRK1234
Capacity: 10000
Hood Type: Closed
Number of Wheels: 12

Car Details:
Make: Toyota
Type: Sedan

```



```
Max Speed: 200
Price: 30000.0
Mileage: 15.0
Registration Number: CAR5678
Number of Doors: 4
Seating Capacity: 5
```

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.

Source code :

```
interface Shape
{
    void area();
    void draw();
    void rotate();
    void move();
}

class Circle implements Shape
{
    Circle(int number)
    {
        System.out.println("-----");
        System.out.println("Printing details of circle "+number+" : ");
    }

    public void area()
    {
        System.out.println("Area of circle is PI * Radius * Radius");
    }

    public void draw()
    {
        System.out.println("Drawing a circle");
    }

    public void rotate()
    {
        System.out.println("Rotating the circle");
    }
}
```

```

    }

    public void move()
    {
        System.out.println("Moving the circle");
    }

    void getDetails()
    {
        this.area();
        this.draw();
        this.rotate();
        this.move();
    }
}

class Rectangle implements Shape
{
    Rectangle(int number)
    {
        System.out.println("-----");
        System.out.println("Printing details of rectangle "+number+" : ");
    }

    public void area()
    {
        System.out.println("Area of rectangle is Length * Breadth");
    }

    public void draw()
    {
        System.out.println("Drawing a rectangle");
    }

    public void rotate()
    {
        System.out.println("Rotating the rectangle");
    }

    public void move()
    {
        System.out.println("Moving the rectangle");
    }

    void getDetails()

```

```

        {
            this.area();
            this.draw();
            this.rotate();
            this.move();
        }
    }

    class Demo
    {
        public static void main(String args[])
        {
            Circle c1 = new Circle(1);
            c1.getDetails();

            Circle c2 = new Circle(2);
            c2.getDetails();

            Rectangle r1 = new Rectangle(1);
            r1.getDetails();

            Rectangle r2 = new Rectangle(2);
            r2.getDetails();

            Rectangle r3 = new Rectangle(3);
            r3.getDetails();
        }
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q14.java
[be2303@localhost a1]$ java Demo
-----
Printing details of circle 1 :
Area of circle is PI * Radius * Radius
Drawing a circle
Rotating the circle
Moving the circle
-----
Printing details of circle 2 :
Area of circle is PI * Radius * Radius
Drawing a circle
Rotating the circle
Moving the circle
-----
Printing details of rectangle 1 :
Area of rectangle is Length * Breadth
Drawing a rectangle
Rotating the rectangle
Moving the rectangle
-----

```

```

Printing details of rectangle 2 :
Area of rectangle is Length * Breadth
Drawing a rectangle
Rotating the rectangle
Moving the rectangle
-----
Printing details of rectangle 3 :
Area of rectangle is Length * Breadth
Drawing a rectangle
Rotating the rectangle
Moving the rectangle

```

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.

Source code :

```

import java.util.*;
import java.lang.*;
class Tollbooth
{
    int totalCarsPassed;
    int carsPassedWithoutPaying;
    int tollCollected;

    Tollbooth()
    {
        totalCarsPassed = 0;
        carsPassedWithoutPaying = 0;
        tollCollected = 0;
    }

    void getData()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the total number of cars passed : ");
        totalCarsPassed = sc.nextInt();
        System.out.println("Enter the number of cars that did not pay the toll : ");
        carsPassedWithoutPaying = sc.nextInt();
        if(carsPassedWithoutPaying > totalCarsPassed)

```

```

        {
            System.out.println("Cars passed without paying toll must be less
than the total number of cars passed!");
            System.exit(0);
        }
    }

    void calculateToll()
    {
        tollCollected = 50*(totalCarsPassed-carsPassedWithoutPaying);
    }

    void printDetails()
    {
        System.out.println("\t\t\t TOLL DETAILS \t\t\t");
        System.out.println("-----");
        System.out.println("Total number of cars passed without paying :
"+carsPassedWithoutPaying);
        System.out.println("Total number of cars passed : "+totalCarsPassed);
        System.out.println("Total cash collected : "+tollCollected);
    }
}

class Demo
{
    public static void main(String args[])
    {
        Tollbooth obj = new Tollbooth();
        obj.getData();
        obj.calculateToll();
        obj.printDetails();
    }
}

```

Output :

```

[be2303@localhost a1]$ javac q15.java
[be2303@localhost a1]$ java Demo
Enter the total number of cars passed :
10
Enter the number of cars that did not pay the toll :
3
                                TOLL DETAILS
-----
Total number of cars passed without paying : 3
Total number of cars passed : 10
Total cash collected : 350

```

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.

Source code :

```
interface Fruit
{
    void hasAPeel();
}

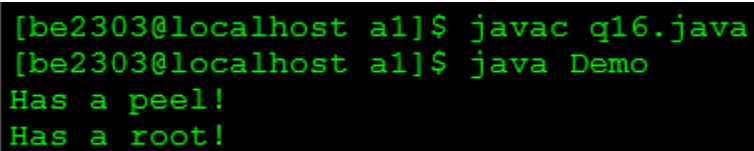
interface Vegetable
{
    void hasARoot();
}

class Tomato implements Fruit, Vegetable
{
    public void hasAPeel()
    {
        System.out.println("Has a peel!");
    }

    public void hasARoot()
    {
        System.out.println("Has a root!");
    }
}

class Demo
{
    public static void main(String args[])
    {
        Tomato obj = new Tomato();
        obj.hasAPeel();
        obj.hasARoot();
    }
}
```

Output :



```
[be2303@localhost a1]$ javac q16.java
[be2303@localhost a1]$ java Demo
Has a peel!
Has a root!
```

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.

Source code :

```
import java.util.*;

class Book {
    String author;
    String title;
    String publisher;
    double cost;
    int stock;

    Book(String title, String author, String publisher, double cost, int stock) {
        this.title = title;
        this.author = author;
        this.publisher = publisher;
        this.cost = cost;
        this.stock = stock;
    }

    boolean matches(String title, String author) {
        return (this.title.equalsIgnoreCase(title) && (this.author.equalsIgnoreCase(author));
    }

    void displayDetails() {
        System.out.println("Title: " + title);
        System.out.println("Author: " + author);
        System.out.println("Publisher: " + publisher);
        System.out.println("Cost: " + cost);
        System.out.println("Stock: " + stock);
    }

    void processPurchase(int requestedCopies) {
        if (requestedCopies <= stock) {
```

```

        System.out.println("Total Cost: " + (cost * requestedCopies));
        stock -= requestedCopies;
    } else {
        System.out.println("Required copies not in stock");
    }
}
}
}
class Demo {

```

```

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Book[] inventory = {
            new Book("Book 1", "author 1", "publisher 1", 500, 5),
            new Book("Book 2", "author 2", "publisher 2", 300, 10),
            new Book("Book 3", "author 3", "publisher 3", 400, 8),
            new Book("Book 4", "author 4", "publisher 4", 350, 6)
        };
        boolean found = false;

```

```

        String title, author;
        System.out.print("Enter the title: ");
        title = sc.nextLine();
        System.out.print("Enter the author: ");
        author = sc.nextLine();

```

```

        for (Book book : inventory) {
            if (book.matches(title, author)) {
                found = true;
                System.out.println("Book is available. Details:");
                book.displayDetails();

```

```

                System.out.print("Enter the number of copies required: ");
                int requestedCopies = sc.nextInt();

```

```

                book.processPurchase(requestedCopies);
                break;
            }
        }
    }
}

```



```
        if (!found) {  
            System.out.println("Book not available.");  
        }  
    }  
}
```

Output :

```
[be2303@localhost a1]$ javac q17.java  
[be2303@localhost a1]$ java Demo  
Enter the title: Book 1  
Enter the author: author 1  
Book is available. Details:  
Title: Book 1  
Author: author 1  
Publisher: publisher 1  
Cost: 500.0  
Stock: 5  
Enter the number of copies required: 2  
Total Cost: 1000.0  
$
```