OOS LAB

ASSIGNMENT-1

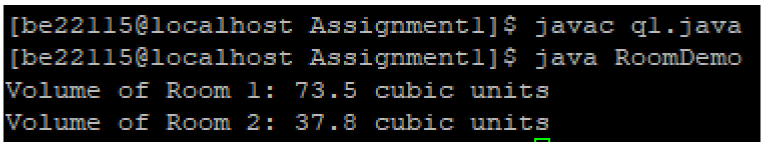
NAME:-SWAPNADEEP MISHRA

ROLL:- 002211001115, SECTION:- A3

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 double height;
    private double width;
    private double breadth;

    public Room(double height, double width, double breadth) {
        this.height = height;
        this.width = width;
        this.breadth = breadth;
    }

    public double volume() {
        return height * width * breadth;
    }
}

public class RoomDemo {
    public static void main(String[] args) {
        Room room1 = new Room(3.5, 4.2, 5.0);
        Room room2 = new Room(2.8, 3.0, 4.5);

        System.out.println("Volume of Room 1: " + room1.volume() + " cubic
units");
```

```
        System.out.println("Volume of Room 2: " + room2.volume() + " cubic
units");
    }
}
```
Output:-

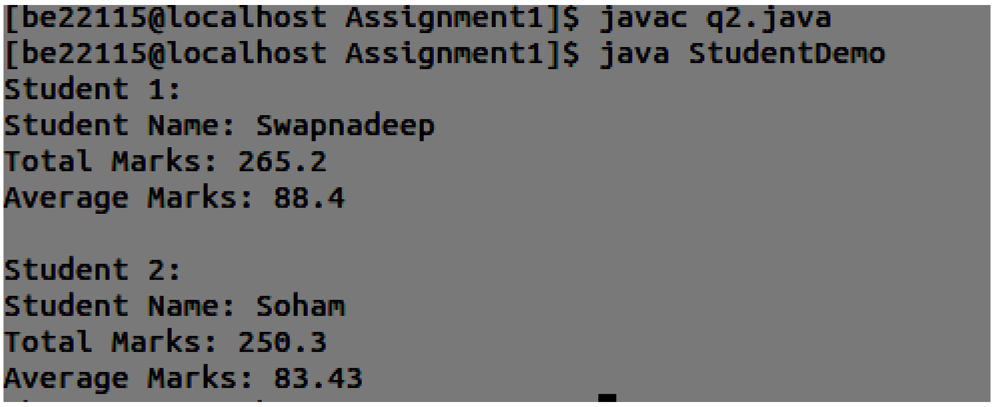


```
[be22115@localhost Assignment1]$ javac q1.java
[be22115@localhost Assignment1]$ java RoomDemo
Volume of Room 1: 73.5 cubic units
Volume of Room 2: 37.8 cubic units
```

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.

```
import java.text.DecimalFormat;
class Student{
private String name;
private double marksSubject1;
private double marksSubject2;
private double marksSubject3;

public void assignValues(String name, double marksSubject1,double
marksSubject2,double marksSubject3){
this.name=name;
this.marksSubject1=marksSubject1;
this.marksSubject2=marksSubject2;
this.marksSubject3=marksSubject3;
}
public double calculateAverage(){
return(marksSubject1+marksSubject2+marksSubject3)/3.0;
}
public void displayInfo(){
DecimalFormat df =  new DecimalFormat("#.##");
System.out.println("Student Name: " + name);
System.out.println("Total Marks: " +
(marksSubject1+marksSubject2+marksSubject3));
System.out.println("Average Marks: " + df.format(calculateAverage()));
}
}
class StudentDemo{
public static void main(String args[]){
Student student1=new Student();
student1.assignValues("Swapnadeep",98.7,86.5,80.0);
Student student2=new Student();
student2.assignValues("Soham",87.7,71.2,91.4);
System.out.println("Student 1:");
student1.displayInfo();
System.out.println();

System.out.println("Student 2:");
student2.displayInfo();
}
}
```

Output:-



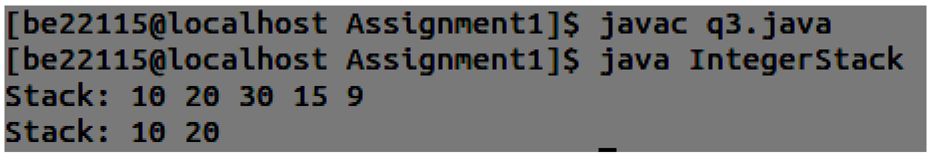
```
[be22115@localhost Assignment1]$ javac q2.java
[be22115@localhost Assignment1]$ java StudentDemo
Student 1:
Student Name: Swapnadeep
Total Marks: 265.2
Average Marks: 88.4

Student 2:
Student Name: Soham
Total Marks: 250.3
Average Marks: 83.43
```

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 IntegerStack{
private int maxSize;
private int[] stackArray;
private int top;
```

```
public IntegerStack(int size){
maxSize=size;
stackArray=new int[maxSize];
top=-1;
}
public void push(int value){
if(top<maxSize-1){
stackArray[++top]=value;
}
else{
System.out.println("Stack is full. Cannot push " +  value);
}
}
public int pop(){
if(top>=0){
return stackArray[top--];
}
else{
System.out.println("Stack is empty. Cannot pop.");
return -1;
}
}
public void printStack(){
if(top>=0){
System.out.print("Stack: ");
for(int i=0; i<=top;i++){
System.out.print(stackArray[i] + " ");
}
System.out.println();
}
else{
System.out.println("Stack is empty.");
}
}
public static void main(String args[]){
IntegerStack stack = new IntegerStack(30);

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

stack.printStack();

stack.pop();
stack.pop();
stack.pop();

stack.printStack();
}
```

Output:-



```
[be22115@localhost Assignment1]$ javac q3.java
[be22115@localhost Assignment1]$ java IntegerStack
Stack: 10 20 30 15 9
Stack: 10 20
```

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.

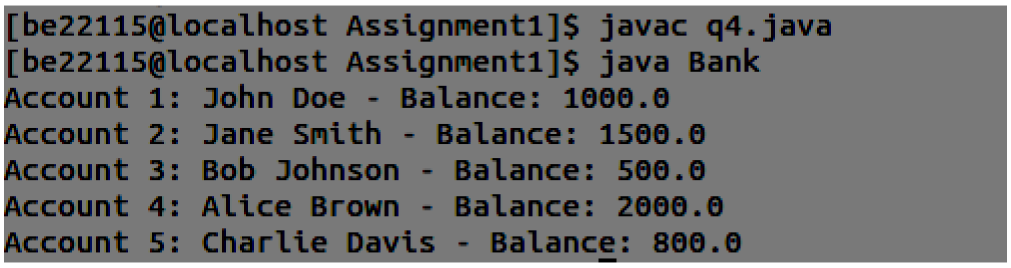
```
class BankAccount {
   private int accountNumber;
   private double balance;
   private String ownerName;

   public BankAccount(int accountNumber, double balance, String
ownerName) {
       this.accountNumber = accountNumber;
       this.balance = balance;
```

```
        this.ownerName = ownerName;
    }

    public int getAccountNumber() {
        return accountNumber;
    }

    public double getBalance() {
        return balance;
    }

    public void add(double amount) {
        balance += amount;
        System.out.println("Amount deposited. New balance: " + balance);
    }

    public void subtract(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Amount withdrawn. New balance: " + balance);
        } else {
            System.out.println("Insufficient funds. Withdrawal failed.");
        }
    }

    public void printDetails() {
        System.out.println("Account " + accountNumber + ": " + ownerName + " -
Balance: " + balance);
    }
}

class AccountManager {
    private BankAccount[] accounts = new BankAccount[5];
    private int numAccounts = 0;

    public void create(int accountNumber, double initialBalance, String
ownerName) {
        if (numAccounts < accounts.length)
            accounts[numAccounts++] = new BankAccount(accountNumber,
initialBalance, ownerName);
        else
```

```
            System.out.println("Cannot create a new account. Maximum capacity
reached.");
        }

    public void delete(int accountNumber) {
        for (int i = 0; i < numAccounts; i++) {
            if (accounts[i].getAccountNumber() == accountNumber) {
                accounts[i] = accounts[--numAccounts];
                System.out.println("Account deleted.");
                return;
            }
        }
        System.out.println("Account not found. Deletion failed.");
    }

    public void deposit(int accountNumber, double amount) {
        for (int i = 0; i < numAccounts; i++) {
            if (accounts[i].getAccountNumber() == accountNumber) {
                accounts[i].add(amount);
                return;
            }
        }
        System.out.println("Account not found. Deposit failed.");
    }

    public void withdraw(int accountNumber, double amount) {
        for (int i = 0; i < numAccounts; i++) {
            if (accounts[i].getAccountNumber() == accountNumber) {
                accounts[i].subtract(amount);
                return;
            }
        }
        System.out.println("Account not found. Withdrawal failed.");
    }

    public void printAllAccountDetails() {
        for (int i = 0; i < numAccounts; i++) {
            accounts[i].printDetails();
        }
    }
}
```

```
public class Bank {
    public static void main(String[] args) {
        AccountManager accountManager = new AccountManager();
        accountManager.create(1, 1000.0, "John Doe");
        accountManager.create(2, 1500.0, "Jane Smith");
        accountManager.create(3, 500.0, "Bob Johnson");
        accountManager.create(4, 2000.0, "Alice Brown");
        accountManager.create(5, 800.0, "Charlie Davis");
        accountManager.printAllAccountDetails();
    }
}
```

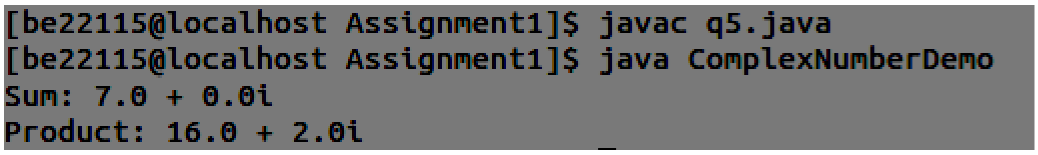
Output:-



```
[be22115@localhost Assignment1]$ javac q4.java
[be22115@localhost Assignment1]$ java Bank
Account 1: John Doe - Balance: 1000.0
Account 2: Jane Smith - Balance: 1500.0
Account 3: Bob Johnson - Balance: 500.0
Account 4: Alice Brown - Balance: 2000.0
Account 5: Charlie Davis - Balance: 800.0
```

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 ComplexNumber {
    private double real;
    private double imaginary;

    public ComplexNumber() {
        this.real = 0;
        this.imaginary = 0;
    }

    public ComplexNumber(double real) {
        this.real = real;
        this.imaginary = 0;
    }

    public ComplexNumber(double real, double imaginary) {
        this.real = real;
        this.imaginary = imaginary;
    }

    public ComplexNumber add(ComplexNumber other) {
        double newReal = this.real + other.real;
        double newImaginary = this.imaginary + other.imaginary;
        return new ComplexNumber(newReal, newImaginary);
    }

    public ComplexNumber multiply(ComplexNumber other) {
        double newReal = this.real * other.real - this.imaginary * other.imaginary;
        double newImaginary = this.real * other.imaginary + this.imaginary *
other.real;
        return new ComplexNumber(newReal, newImaginary);
    }

    public void print() {
        System.out.println(this.real + " + " + this.imaginary + "i");
    }
}

public class ComplexNumberDemo {
    public static void main(String[] args) {
        ComplexNumber complex1 = new ComplexNumber(3, 2);
        ComplexNumber complex2 = new ComplexNumber(4, -2);

        ComplexNumber sum = complex1.add(complex2);
        System.out.print("Sum: ");
        sum.print();

        ComplexNumber product = complex1.multiply(complex2);
        System.out.print("Product: ");
        product.print();
    }
}
```
Output:-

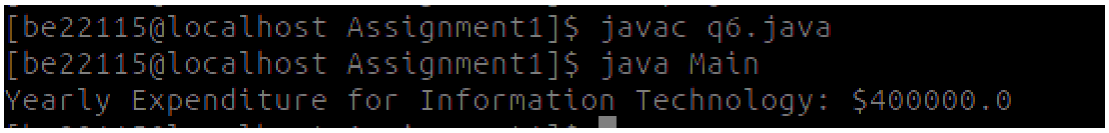

```
[be22115@localhost Assignment1]$ javac q5.java
[be22115@localhost Assignment1]$ java ComplexNumberDemo
Sum: 7.0 + 0.0i
Product: 16.0 + 2.0i
```

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.

```
import java.util.ArrayList;

class Employee {
    private String name;
    private int id;
    private String address;
    private double salary;

    public Employee(String name, int id, String address, double salary) {
        this.name = name;
        this.id = id;
        this.address = address;
        this.salary = salary;
    }

    public String getName() {
        return name;
    }

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

    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }

    public String getAddress() {
        return address;
    }

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

    public double getSalary() {
        return salary;
    }

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

class Dept {
    private String name;
    private String location;
    private ArrayList<Employee> employees;

    public Dept(String name, String location) {
        this.name = name;
        this.location = location;
        this.employees = new ArrayList<>();
    }

    public void addEmployee(Employee employee) {
        employees.add(employee);
    }

    public void removeEmployee(Employee employee) {
        employees.remove(employee);
    }

    public double calculateYearlyExpenditure() {
        double totalExpenditure = 0;
        for (Employee employee : employees) {
            totalExpenditure += employee.getSalary();
        }
        return totalExpenditure;
    }

    public String getName() {
        return name;
    }

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

    public String getLocation() {
        return location;
    }

    public void setLocation(String location) {
        this.location = location;
    }
}

public class Main {
    public static void main(String[] args) {
        Dept itDept = new Dept("Information Technology", "Headquarters");

        itDept.addEmployee(new Employee("John Doe", 1, "123 Main St",
60000));
        itDept.addEmployee(new Employee("Jane Smith", 2, "456 Oak St",
70000));
        itDept.addEmployee(new Employee("Bob Johnson", 3, "789 Pine St",
80000));
        itDept.addEmployee(new Employee("Alice Williams", 4, "101 Maple St",
90000));
        itDept.addEmployee(new Employee("Charlie Brown", 5, "202 Cedar St",
100000));

        double yearlyExpenditure = itDept.calculateYearlyExpenditure();
        System.out.println("Yearly Expenditure for " + itDept.getName() + ": $" +
yearlyExpenditure);
```

```
    }
}
```
Output:-



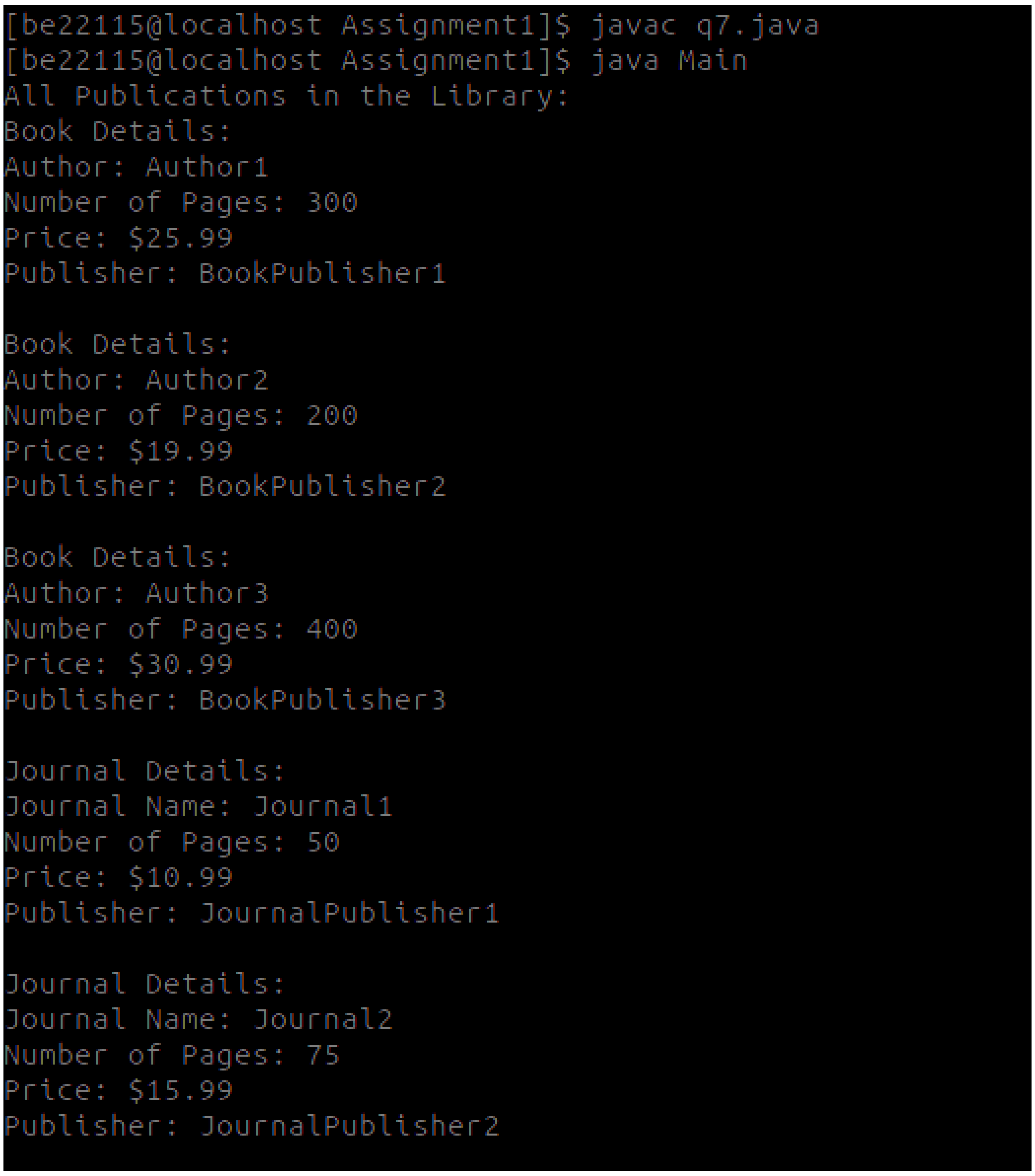
```
[be22115@localhost Assignment1]$ javac q6.java
[be22115@localhost Assignment1]$ java Main
Yearly Expenditure for Information Technology: $400000.0
```

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.

```
import java.util.ArrayList;

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 displayDetails();
}

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;
    }

    @Override
    public void displayDetails() {
        System.out.println("Book Details:");
        System.out.println("Author: " + getAuthor());
        System.out.println("Number of Pages: " + getNoOfPages());
        System.out.println("Price: $" + getPrice());
        System.out.println("Publisher: " + getPublisherName());
        System.out.println();
    }
}

class Journal extends Publication {
    private String journalName;

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

    public String getJournalName() {
        return journalName;
    }

    public void setJournalName(String journalName) {
        this.journalName = journalName;
    }

    @Override
    public void displayDetails() {
        System.out.println("Journal Details:");
        System.out.println("Journal Name: " + getJournalName());
        System.out.println("Number of Pages: " + getNoOfPages());
        System.out.println("Price: $" + getPrice());
        System.out.println("Publisher: " + getPublisherName());
        System.out.println();
    }
}

class Library {
    private ArrayList<Publication> publications;

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

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

    public void printAllPublications() {
        System.out.println("All Publications in the Library:");
        for (Publication publication : publications) {
            publication.displayDetails();
        }
    }
}

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

        Book book1 = new Book(300, 25.99, "BookPublisher1", "Author1");
        Book book2 = new Book(200, 19.99, "BookPublisher2", "Author2");
        Book book3 = new Book(400, 30.99, "BookPublisher3", "Author3");

        Journal journal1 = new Journal(50, 10.99, "JournalPublisher1",
"Journal1");
        Journal journal2 = new Journal(75, 15.99, "JournalPublisher2",
"Journal2");

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

        library.printAllPublications();
    }
}
```
Output:-

8.



```
[be22115@localhost Assignment1]$ javac q7.java
[be22115@localhost Assignment1]$ java Main
All Publications in the Library:
Book Details:
Author: Author1
Number of Pages: 300
Price: $25.99
Publisher: BookPublisher1

Book Details:
Author: Author2
Number of Pages: 200
Price: $19.99
Publisher: BookPublisher2

Book Details:
Author: Author3
Number of Pages: 400
Price: $30.99
Publisher: BookPublisher3

Journal Details:
Journal Name: Journal1
Number of Pages: 50
Price: $10.99
Publisher: JournalPublisher1

Journal Details:
Journal Name: Journal2
Number of Pages: 75
Price: $15.99
Publisher: JournalPublisher2
```

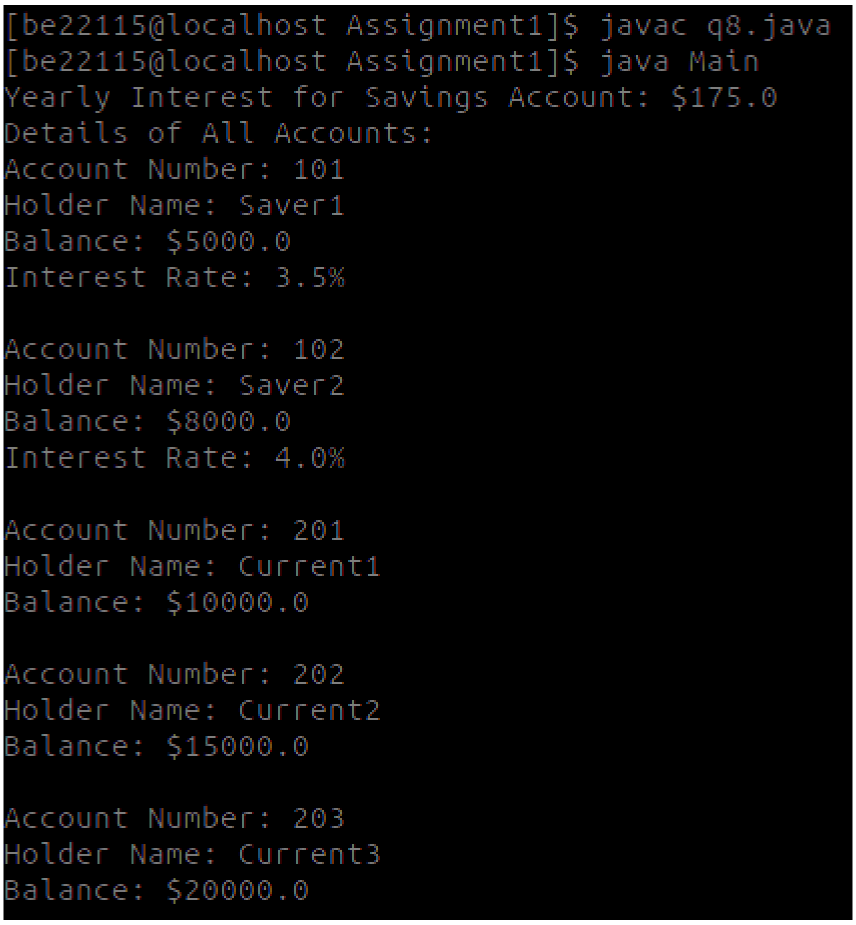
Write a class for “Account” containing data members ‘accountNumber’, ‘holderName’,
‘balance’ and add constructors and necessary accessor/modifier functions for these data
members. Now create two class “SavingsAccount” and “CurrentAccount” extending from this
class. SavingsAccount will have a member variable ‘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.

```
import java.util.ArrayList;

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 displayDetails() {
        System.out.println("Account Number: " + getAccountNumber());
        System.out.println("Holder Name: " + getHolderName());
        System.out.println("Balance: $" + getBalance());
    }
```

```
}

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 void calculateYearlyInterest() {
      double yearlyInterest = getBalance() * interestRate / 100;
      System.out.println("Yearly Interest for Savings Account: $" +
yearlyInterest);
   }

   @Override
   public void displayDetails() {
      super.displayDetails();
      System.out.println("Interest Rate: " + getInterestRate() + "%");
   }
}

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

class Manager {
   private ArrayList<Account> accounts;
```

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

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

    public void printAllAccounts() {
        System.out.println("Details of All Accounts:");
        for (Account account : accounts) {
            account.displayDetails();
            System.out.println();
        }
    }
}

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

        SavingsAccount savings1 = new SavingsAccount(101, "Saver1", 5000,
3.5);
        SavingsAccount savings2 = new SavingsAccount(102, "Saver2", 8000,
4.0);

        CurrentAccount current1 = new CurrentAccount(201, "Current1", 10000);
        CurrentAccount current2 = new CurrentAccount(202, "Current2", 15000);
        CurrentAccount current3 = new CurrentAccount(203, "Current3", 20000);

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

        savings1.calculateYearlyInterest();

        manager.printAllAccounts();
    }
}
```

Output:-

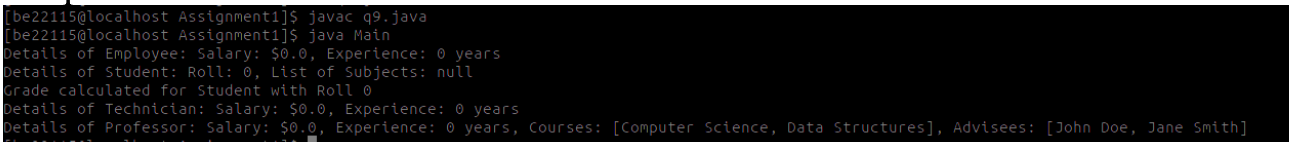


```
[be22115@localhost Assignment1]$ javac q8.java
[be22115@localhost Assignment1]$ java Main
Yearly Interest for Savings Account: $175.0
Details of All Accounts:
Account Number: 101
Holder Name: Saver1
Balance: $5000.0
Interest Rate: 3.5%

Account Number: 102
Holder Name: Saver2
Balance: $8000.0
Interest Rate: 4.0%

Account Number: 201
Holder Name: Current1
Balance: $10000.0

Account Number: 202
Holder Name: Current2
Balance: $15000.0

Account Number: 203
Holder Name: Current3
Balance: $20000.0
```

**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.ArrayList;
import java.util.Date;

class Person {
    private int age;
    private double weight;
    private double height;
    private Date dateOfBirth;
    private String 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 Date getDateOfBirth() {
        return dateOfBirth;
    }

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

    public String getAddress() {
        return address;
    }

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

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

    public double getSalary() {
        return salary;
    }

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

    public Date getDateOfJoining() {
        return dateOfJoining;
    }

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

    public int getExperience() {
        return experience;
    }

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

class Student extends Person {
    private int roll;
    private ArrayList<String> listOfSubjects;
    private ArrayList<Integer> marks;
    public int getRoll() {
        return roll;
    }

    public void setRoll(int roll) {
        this.roll = roll;
    }

    public ArrayList<String> getListOfSubjects() {
        return listOfSubjects;
    }

    public void setListOfSubjects(ArrayList<String> listOfSubjects) {
        this.listOfSubjects = listOfSubjects;
    }

    public ArrayList<Integer> getMarks() {
        return marks;
    }

    public void setMarks(ArrayList<Integer> marks) {
        this.marks = marks;
    }

    public void calculateGrade() {
        System.out.println("Grade calculated for Student with Roll " + getRoll());
    }
}

class Technician extends Employee {}

class Professor extends Employee {
    private ArrayList<String> courses;
    private ArrayList<String> listOfAdvisee;

    public ArrayList<String> getCourses() {
        return courses;
    }

    public void setCourses(ArrayList<String> courses) {
        this.courses = courses;
    }

    public ArrayList<String> getListOfAdvisee() {
        return listOfAdvisee;
    }

    public void setListOfAdvisee(ArrayList<String> listOfAdvisee) {
        this.listOfAdvisee = listOfAdvisee;
    }

    public void addAdvisee(String studentName) {
        listOfAdvisee.add(studentName);
    }

    public void removeAdvisee(String studentName) {
        listOfAdvisee.remove(studentName);
    }
}

public class Main {
    public static void main(String[] args) {
        Employee employee = new Employee();
        Student student = new Student();
        Technician technician = new Technician();
        Professor professor = new Professor();

        professor.setCourses(new ArrayList<>());
        professor.setListOfAdvisee(new ArrayList<>());

        professor.getCourses().add("Computer Science");
        professor.getCourses().add("Data Structures");

        professor.addAdvisee("John Doe");
        professor.addAdvisee("Jane Smith");

        System.out.println("Details of Employee: Salary: $" +
employee.getSalary() +
                ", Experience: " + employee.getExperience() + " years");
        System.out.println("Details of Student: Roll: " + student.getRoll() +
                ", List of Subjects: " + student.getListOfSubjects());
        student.calculateGrade();
        System.out.println("Details of Technician: Salary: $" +
technician.getSalary() +
                ", Experience: " + technician.getExperience() + " years");
        System.out.println("Details of Professor: Salary: $" + professor.getSalary()
+
                ", Experience: " + professor.getExperience() +
                " years, Courses: " + professor.getCourses() +
                ", Advisees: " + professor.getListOfAdvisee());
    }
}
```

Output:-

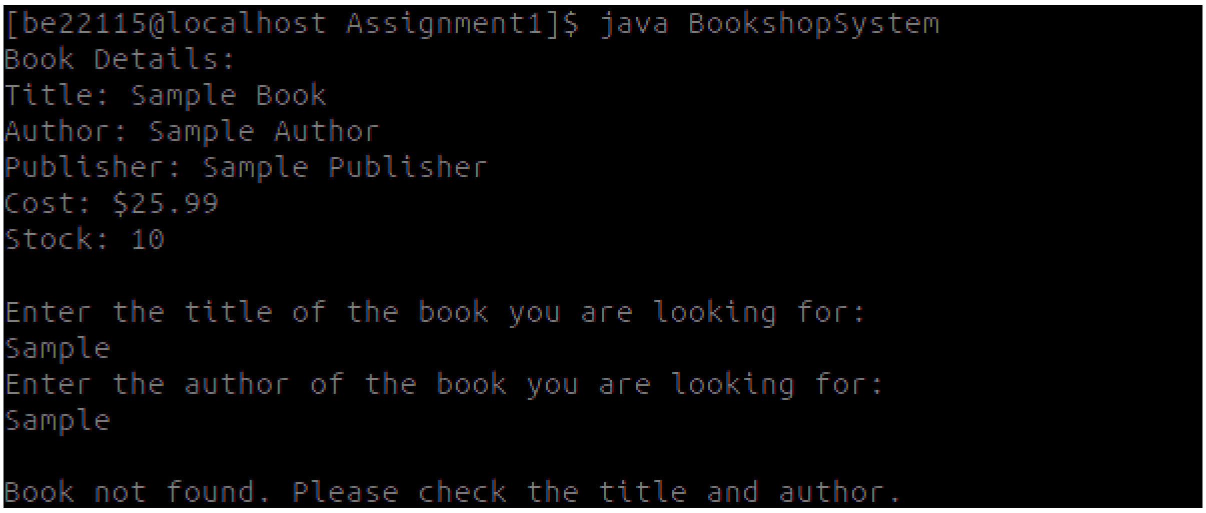


```
[be22115@localhost Assignment1]$ javac q9.java
[be22115@localhost Assignment1]$ java Main
Details of Employee: Salary: $0.0, Experience: 0 years
Details of Student: Roll: 0, List of Subjects: null
Grade calculated for Student with Roll 0
Details of Technician: Salary: $0.0, Experience: 0 years
Details of Professor: Salary: $0.0, Experience: 0 years, Courses: [Computer Science, Data Structures], Advisees: [John Doe, Jane Smith]
```

**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.Scanner;

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

    public void displayDetails() {
        System.out.println("Book Details:");
        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);
    }

    public void processOrder(int requiredCopies) {
        if (stock >= requiredCopies) {
            System.out.println("Book is available. Please proceed with your
order.");
            System.out.println("Enter the number of copies required: " +
requiredCopies);

            double totalCost = cost * requiredCopies;
            System.out.println("Total cost: $" + totalCost);
        } else {
            System.out.println("Required copies not in stock.");
        }
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }
}

public class BookshopSystem {
    public static void main(String[] args) {
        Book sampleBook = new Book("Sample Book", "Sample Author",
"Sample Publisher", 25.99, 10);

        sampleBook.displayDetails();

        Scanner scanner = new Scanner(System.in);

        System.out.println("\nEnter the title of the book you are looking for:");
        String customerTitle = scanner.nextLine();

        System.out.println("Enter the author of the book you are looking for:");
        String customerAuthor = scanner.nextLine();

        if (customerTitle.equalsIgnoreCase(sampleBook.getTitle()) &&
customerAuthor.equalsIgnoreCase(sampleBook.getAuthor())) {
            System.out.println("\nBook found!");
            sampleBook.displayDetails();

            System.out.println("\nEnter the number of copies you want to order:");
            int requiredCopies = scanner.nextInt();

            sampleBook.processOrder(requiredCopies);
        } else {
            System.out.println("\nBook not found. Please check the title and
author.");
        }

        scanner.close();
    }
}
```

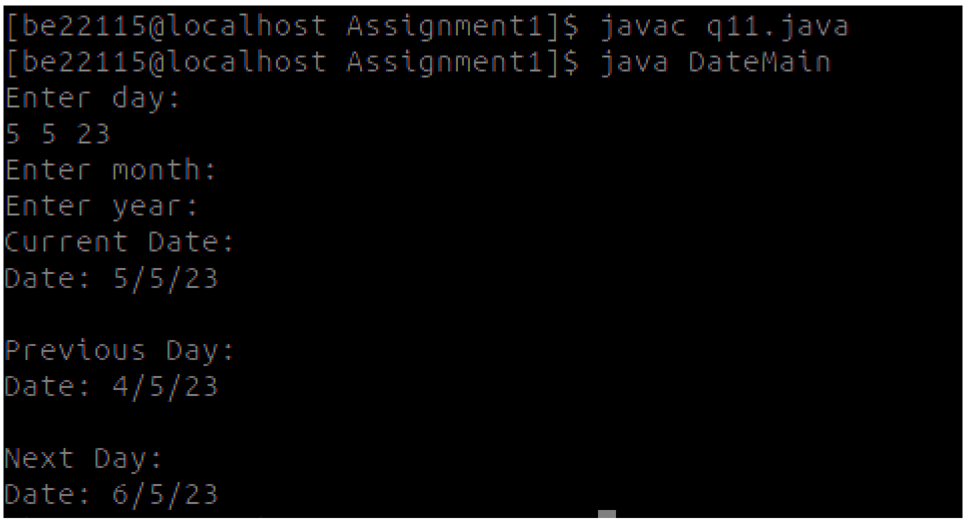
Output:-



```
[be22115@localhost Assignment1]$ java BookshopSystem
Book Details:
Title: Sample Book
Author: Sample Author
Publisher: Sample Publisher
Cost: $25.99
Stock: 10

Enter the title of the book you are looking for:
Sample
Enter the author of the book you are looking for:
Sample

Book not found. Please check the title and author.
```

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.

```
import java.util.Scanner;

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

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

    public Date(int day) {
        this.day = day;
        this.month = 1;
        this.year = 1970;
    }

    public Date(int day, int month) {
        this.day = day;
        this.month = month;
        this.year = 1970;
    }

    public Date(int day, int month, int year) {
        this.day = day;
        this.month = month;
        this.year = year;
    }

    public Date getPreviousDay() {
        if (day > 1) {
            return new Date(day - 1, month, year);
        } else {
            if (month > 1) {
                int previousMonthDays = getDaysInMonth(month - 1, year);
                return new Date(previousMonthDays, month - 1, year);
            } else {
                return new Date(31, 12, year - 1);
            }
        }
    }

    public Date getNextDay() {
        int daysInMonth = getDaysInMonth(month, year);

        if (day < daysInMonth) {
            return new Date(day + 1, month, year);
        } else {
            if (month < 12) {
                return new Date(1, month + 1, year);
            } else {
                return new Date(1, 1, year + 1);
            }
        }
    }
    public void printDay() {
        System.out.println("Date: " + day + "/" + month + "/" + year);
    }

    private int getDaysInMonth(int month, int year) {
        switch (month) {
            case 4:
            case 6:
            case 9:
            case 11:
                return 30;
            case 2:
                return (isLeapYear(year) ? 29 : 28);
            default:
                return 31;
        }
    }

    private boolean isLeapYear(int year) {
        return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
    }
}

public class DateMain {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter day:");
        int day = scanner.nextInt();

        System.out.println("Enter month:");
        int month = scanner.nextInt();

        System.out.println("Enter year:");
        int year = scanner.nextInt();

        Date currentDate = new Date(day, month, year);

        Date previousDay = currentDate.getPreviousDay();
        Date nextDay = currentDate.getNextDay();

        System.out.println("Current Date:");
        currentDate.printDay();

        System.out.println("\nPrevious Day:");
        previousDay.printDay();

        System.out.println("\nNext Day:");
        nextDay.printDay();

        scanner.close();
    }
}
```
Output:-



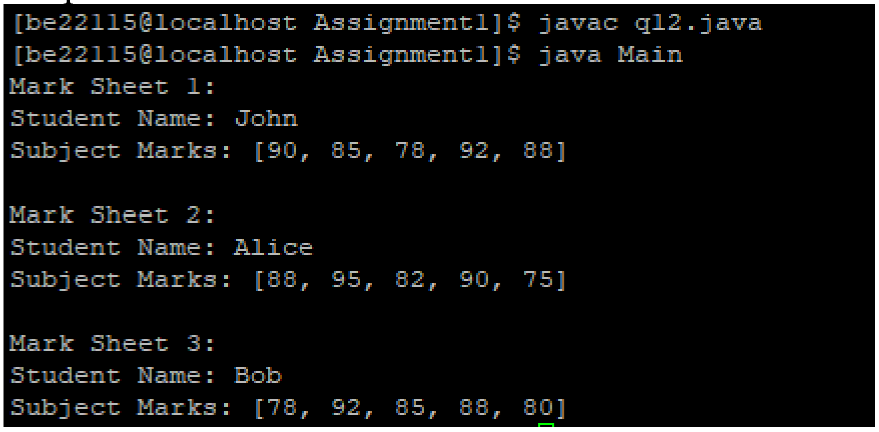
```
[be22115@localhost Assignment1]$ javac q11.java
[be22115@localhost Assignment1]$ java DateMain
Enter day:
5 5 23
Enter month:
Enter year:
Current Date:
Date: 5/5/23

Previous Day:
Date: 4/5/23

Next Day:
Date: 6/5/23
```

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.

```
import java.util.Arrays;

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

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

    public String getName() {
        return name;
    }

    public int getRollNo() {
        return rollNo;
    }

    public String[] getSubjectNames() {
        return subjectNames;
    }

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

    public void setRollNo(int rollNo) {
        this.rollNo = rollNo;
    }

    public void setSubjectNames(String[] subjectNames) {
        this.subjectNames = subjectNames;
    }
}

class TabulationSheet {
    private int[] rollNumbers;
    private int[] marks;
    public TabulationSheet(int[] rollNumbers, int[] marks) {
        this.rollNumbers = rollNumbers;
        this.marks = marks;
    }

    public void addMarks(int rollNo, int marks) {
        for (int i = 0; i < rollNumbers.length; i++) {
            if (rollNumbers[i] == 0) {
                rollNumbers[i] = rollNo;
                this.marks[i] = marks;
                break;
            }
        }
    }
}

class MarkSheet {
    private String studentName;
    private int[] subjectMarks;

    public MarkSheet(String studentName, int[] subjectMarks) {
        this.studentName = studentName;
        this.subjectMarks = subjectMarks;
    }

    public void addMarks(String studentName, int[] subjectMarks) {
        this.studentName = studentName;
        this.subjectMarks = subjectMarks;
    }

    public void printMarkSheet() {
        System.out.println("Student Name: " + studentName);
        System.out.println("Subject Marks: " + Arrays.toString(subjectMarks));
    }
}

public class Main {
    public static void main(String[] args) {
        Student student1 = new Student("John", 101, new String[]{"Math",
"Physics", "Chemistry", "English", "History"});
        Student student2 = new Student("Alice", 102, new String[]{"Math",
"Physics", "Chemistry", "English", "History"});
```

```
        Student student3 = new Student("Bob", 103, new String[]{"Math",
"Physics", "Chemistry", "English", "History"});

        TabulationSheet mathTabulation = new TabulationSheet(new int[3], new
int[3]);
        TabulationSheet physicsTabulation = new TabulationSheet(new int[3], new
int[3]);
        TabulationSheet chemistryTabulation = new TabulationSheet(new int[3],
new int[3]);
        TabulationSheet englishTabulation = new TabulationSheet(new int[3], new
int[3]);
        TabulationSheet historyTabulation = new TabulationSheet(new int[3], new
int[3]);

        MarkSheet markSheet1 = new MarkSheet("", new int[5]);
        MarkSheet markSheet2 = new MarkSheet("", new int[5]);
        MarkSheet markSheet3 = new MarkSheet("", new int[5]);

        mathTabulation.addMarks(student1.getRollNo(), 90);
        physicsTabulation.addMarks(student1.getRollNo(), 85);
        chemistryTabulation.addMarks(student1.getRollNo(), 78);
        englishTabulation.addMarks(student1.getRollNo(), 92);
        historyTabulation.addMarks(student1.getRollNo(), 88);

        mathTabulation.addMarks(student2.getRollNo(), 88);
        physicsTabulation.addMarks(student2.getRollNo(), 95);
        chemistryTabulation.addMarks(student2.getRollNo(), 82);
        englishTabulation.addMarks(student2.getRollNo(), 90);
        historyTabulation.addMarks(student2.getRollNo(), 75);

        mathTabulation.addMarks(student3.getRollNo(), 78);
        physicsTabulation.addMarks(student3.getRollNo(), 92);
        chemistryTabulation.addMarks(student3.getRollNo(), 85);
        englishTabulation.addMarks(student3.getRollNo(), 88);
        historyTabulation.addMarks(student3.getRollNo(), 80);

        markSheet1.addMarks(student1.getName(), new int[]{90, 85, 78, 92, 88});
        markSheet2.addMarks(student2.getName(), new int[]{88, 95, 82, 90, 75});
        markSheet3.addMarks(student3.getName(), new int[]{78, 92, 85, 88, 80});

        System.out.println("Mark Sheet 1:");
        markSheet1.printMarkSheet();
        System.out.println("\nMark Sheet 2:");
        markSheet2.printMarkSheet();

        System.out.println("\nMark Sheet 3:");
        markSheet3.printMarkSheet();
    }
}
```

Output:-



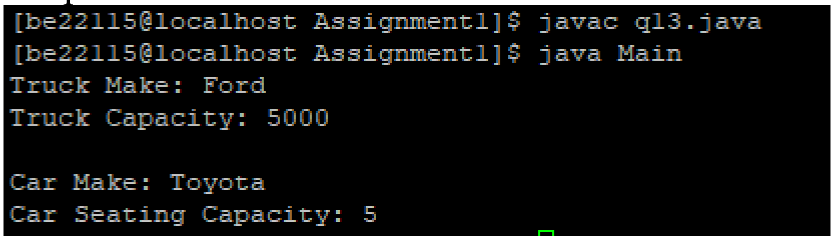
```
[be22115@localhost Assignment1]$ javac q12.java
[be22115@localhost Assignment1]$ java Main
Mark Sheet 1:
Student Name: John
Subject Marks: [90, 85, 78, 92, 88]

Mark Sheet 2:
Student Name: Alice
Subject Marks: [88, 95, 82, 90, 75]

Mark Sheet 3:
Student Name: Bob
Subject Marks: [78, 92, 85, 88, 80]
```

**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 {
    private String make;
    private String type;
    private int maxSpeed;
    private double price;
    private double mileage;
    private String 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;
    }
}

class Truck extends Automobile {
    private int capacity;
    private String hoodType;
    private int 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;
    }
}

class Car extends Automobile {
    private int noOfDoors;
    private int 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 class Main {
    public static void main(String[] args) {
        Truck truck = new Truck();
        truck.setMake("Ford");
        truck.setType("Truck");
        truck.setMaxSpeed(120);
        truck.setPrice(50000.0);
        truck.setMileage(15.5);
        truck.setRegistrationNumber("ABC123");
        truck.setCapacity(5000);
        truck.setHoodType("Open");
        truck.setNoOfWheels(6);

        Car car = new Car();
        car.setMake("Toyota");
        car.setType("Sedan");
        car.setMaxSpeed(150);
        car.setPrice(30000.0);
        car.setMileage(20.5);
        car.setRegistrationNumber("XYZ789");
        car.setNoOfDoors(4);
        car.setSeatingCapacity(5);

        System.out.println("Truck Make: " + truck.getMake());
        System.out.println("Truck Capacity: " + truck.getCapacity());

        System.out.println("\nCar Make: " + car.getMake());
        System.out.println("Car Seating Capacity: " + car.getSeatingCapacity());
    }
}
```

Output:-



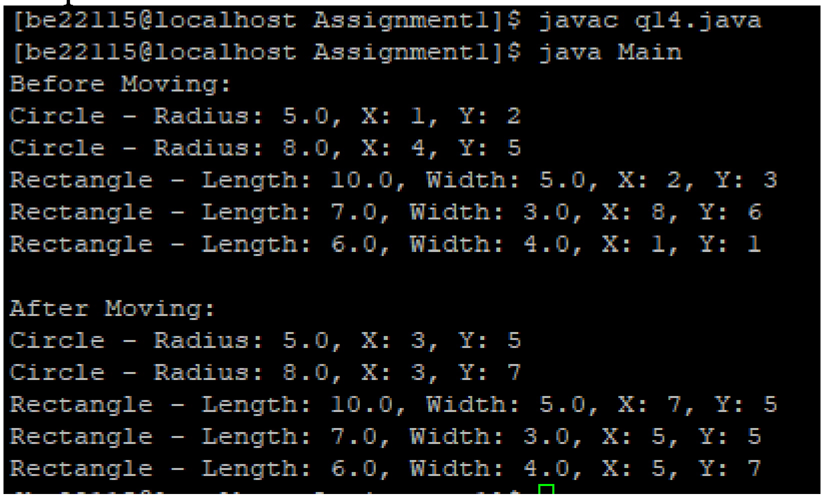
```
[be22115@localhost Assignment1]$ javac q13.java
[be22115@localhost Assignment1]$ java Main
Truck Make: Ford
Truck Capacity: 5000

Car Make: Toyota
Car 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.

```
interface Shape {
    double area();
    void draw();
    void rotate();
    void move(int deltaX, int deltaY);
}

class Circle implements Shape {
    private double radius;
    private int x;
    private int y;

    public Circle(double radius, int x, int y) {
        this.radius = radius;
        this.x = x;
        this.y = y;
    }

    @Override
    public double area() {
```

```java
        return Math.PI * radius * radius;
    }

    @Override
    public void draw() {
        System.out.println("Drawing Circle");
    }

    @Override
    public void rotate() {
        System.out.println("Rotating Circle");
    }

    @Override
    public void move(int deltaX, int deltaY) {
        this.x += deltaX;
        this.y += deltaY;
    }

    public void printDetails() {
        System.out.println("Circle - Radius: " + radius + ", X: " + x + ", Y: " + y);
    }
}

class Rectangle implements Shape {
    private double length;
    private double width;
    private int x;
    private int y;

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

    @Override
    public double area() {
        return length * width;
    }
    @Override
    public void draw() {
        System.out.println("Drawing Rectangle");
    }

    @Override
    public void rotate() {
        System.out.println("Rotating Rectangle");
    }

    @Override
    public void move(int deltaX, int deltaY) {
        this.x += deltaX;
        this.y += deltaY;
    }

    public void printDetails() {
        System.out.println("Rectangle - Length: " + length + ", Width: " + width +
", X: " + x + ", Y: " + y);
    }
}

public class Main {
    public static void main(String[] args) {
        Circle circle1 = new Circle(5, 1, 2);
        Circle circle2 = new Circle(8, 4, 5);

        Rectangle rectangle1 = new Rectangle(10, 5, 2, 3);
        Rectangle rectangle2 = new Rectangle(7, 3, 8, 6);
        Rectangle rectangle3 = new Rectangle(6, 4, 1, 1);

        System.out.println("Before Moving:");
        circle1.printDetails();
        circle2.printDetails();
        rectangle1.printDetails();
        rectangle2.printDetails();
        rectangle3.printDetails();

        circle1.move(2, 3);
        circle2.move(-1, 2);
        rectangle1.move(5, 2);
        rectangle2.move(-3, -1);
        rectangle3.move(4, 6);

        System.out.println("\nAfter Moving:");
        circle1.printDetails();
        circle2.printDetails();
        rectangle1.printDetails();
        rectangle2.printDetails();
        rectangle3.printDetails();
    }
}
```

Output:-

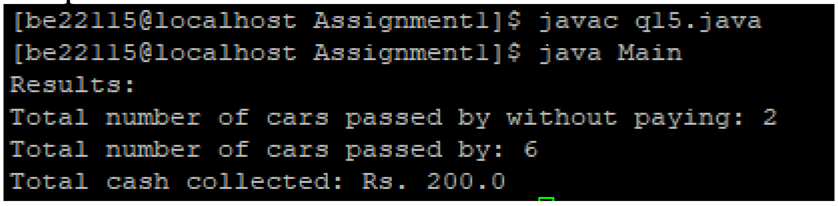


```
[be22115@localhost Assignment1]$ javac q14.java
[be22115@localhost Assignment1]$ java Main
Before Moving:
Circle - Radius: 5.0, X: 1, Y: 2
Circle - Radius: 8.0, X: 4, Y: 5
Rectangle - Length: 10.0, Width: 5.0, X: 2, Y: 3
Rectangle - Length: 7.0, Width: 3.0, X: 8, Y: 6
Rectangle - Length: 6.0, Width: 4.0, X: 1, Y: 1

After Moving:
Circle - Radius: 5.0, X: 3, Y: 5
Circle - Radius: 8.0, X: 3, Y: 7
Rectangle - Length: 10.0, Width: 5.0, X: 7, Y: 5
Rectangle - Length: 7.0, Width: 3.0, X: 5, Y: 5
Rectangle - Length: 6.0, Width: 4.0, X: 5, Y: 7
```

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 printout 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 totalCars;
    private double totalCashCollected;

    public void carPasses() {
        totalCars++;
        totalCashCollected += 50;
    }

    public void carPassesWithoutPayment() {
        carsWithoutPayment++;
        totalCars++;
    }

    public void displayResults() {
        System.out.println("Results:");
        System.out.println("Total number of cars passed by without paying: " +
carsWithoutPayment);
        System.out.println("Total number of cars passed by: " + totalCars);
        System.out.println("Total cash collected: Rs. " + totalCashCollected);
    }
}

public class Main {
    public static void main(String[] args) {
        TollBooth tollBooth = new TollBooth();

        tollBooth.carPasses();
        tollBooth.carPasses();
        tollBooth.carPassesWithoutPayment();
        tollBooth.carPasses();
        tollBooth.carPassesWithoutPayment();
        tollBooth.carPasses();

        tollBooth.displayResults();
    }
}
```

Output:-

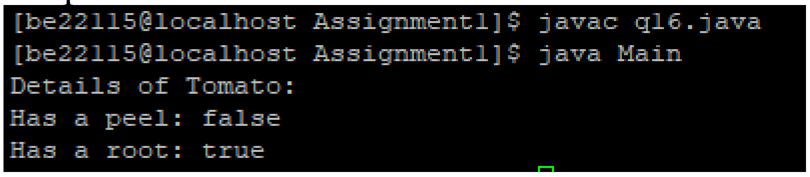


```
[be22115@localhost Assignment1]$ javac q15.java
[be22115@localhost Assignment1]$ java Main
Results:
Total number of cars passed by without paying: 2
Total number of cars passed by: 6
Total cash collected: Rs. 200.0
```

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();
}

interface Vegetable {
   boolean hasARoot();
}

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

   @Override
   public boolean hasARoot() {
      return true;
   }
}

public class Main {
   public static void main(String[] args) {
      Tomato tomato = new Tomato();

      System.out.println("Details of Tomato:");
      System.out.println("Has a peel: " + tomato.hasAPeel());
      System.out.println("Has a root: " + tomato.hasARoot());
   }
}
```

Output:-



```
[be22115@localhost Assignment1]$ javac q16.java
[be22115@localhost Assignment1]$ java Main
Details of Tomato:
Has a peel: false
Has a root: true
```

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.

Same as question no. 10