(1) Develop a java program that prints all real solutions to the quadratic equation $ax^2+bx+c=0$. Read in a, b, c and use the quadratic formula. If the discriminant b^2-4ac is negative, display a message stating that there are no real solutions.

```
import java.util.Scanner;
class Quadratic
{
  int a,b,c;
  double r1, r2, d;
  void getd()
  {
  Scanner s = new Scanner (System.in);
  System.out.println("Enter the coefficient of a,b,c");
  a = s.nextInt();
  b = s.nextInt();
  c = s.nextInt();
  }
  void compute()
  {
    while (a==0)
  {
    System.out.println("Not a quadratic Equation");
  System.out.println("Enter a non zero value for a:");
  Scanner s = new Scanner (System.in);
    a = s.nextInt();
  }
  d = b*b - 4*a*c;
  if (d==0)
  {
    r1 = (-b)/(2*a);
  System.out.println("Roots are real and Equal");
  System.out.println("Root1 = Root2 = " + r1);
  }
  else if (d>0)
  {
```

```
r1 = ((-b) + (math.sqrt(d))) / (double)(2*a);
r2 = ((-b) - (math.sqrt(d))) / (double)(2*a);
System.out.println("Roots are real and distinct");
System.out.println("Root1 = " +r1+ " Root2 = " +r2);
}
Else if (d>0)
{
  System.out.println("Roots are imaginary");
r1 = (-b) / (2*a);
r2 = math.sqrt(-d) / (2*a);
System.out.println("Root1 = " +r1+ "+i" +r2);
System.out.println("Root1 = " +r1+ "-i" +r2);
  }
}
}
class Quadratic main
{
  Public Static void main (String args[])
{
Quadratic q = new Quadratic();
q.getd();
 q.compute();
}
}
```

Out Put:

Enter the coefficient of a, b, c

10

15

20

roots are imaginary

root 1 = 0.0 + i1.1989578808281798

root 1 = 0.0 + i1.1989578808281798

Enter the coefficient of a, b, c

3

9

5

Roots are real and distinct

Root1 = -0.7362373738417402 67

Root2 = -2.2637626158259 73

(2) Develop a java program to create a class Student with member usn, name, an array credits and an array marks, include method to accept and display details and a method to calculate SGPA of a Student.

-> import java.util.Scanner;

class Student {

Private String usn;

Private String name;

Private int[] credits;

Private int[] marks;

Public Student (String usn, String name, int numSubjects)

{ this.usn = usn;

this.name = name;

this.credits = new int [numSubjects];

this.marks = new int [numSubjects];

}

Public void acceptDetails ()

{

Scanner scanner = new Scanner(System.in);

System.out.Print ("Enter USN: ");

usn = scanner.nextLine();

System.out.Print ("Enter Name:");

name = Scanner.nextLine();

```
System.out.println("Enter details for each subject:");
for (int i=0; i<credits.length; i++) {
System.out.print("Enter credits for subject "+(i+1)+":");
credits[i] = scanner.nextInt();
System.out.print("Enter marks for subject "+
(i+1)+":");
marks[i] = scanner.nextInt();
}
}
Public void displayDetails() {
System.out.println("Student Details:");
System.out.println("USN:"+usn);
System.out.println("Name:"+name);
System.out.println("Subject-wise details:");
for (int i=0; i<credits.length; i++) {
System.out.println("Subject "+(i+1)+": credits-"
+credits[i]+", marks-"+marks[i]);
}
}
Public double calculateSGPA(){
double totalCredits=0;
double totalGradePoints=0;
for (int i=0; i<credits.length; i++) {
totalCredits+=credits[i];
totalGradePoints+=CalculateGradePoints(marks[i])*credits[i];
}
return totalGradePoints/totalCredits;
}
Private double calculateGradePoints(int marks){
if (marks>=90) {
return 10.0;
} else if (marks>=80) {
```

```
        return 9.0;
    } else if (marks >= 70) {
        return 8.0;
    } else if (marks >= 60) {
        return 7.0;
    } else if (marks >= 50) {
        return 6.0;
    } else if (marks >= 40) {
        return 5.0;
    } else {
        return 0.0;
    }
  }
}

Public class StudentMain{
Public Static void main (String[] args) {
 Scanner scanner = new Scanner (System.in);
System.out.Print ("Enter the number of Subjects :");
int numSubjects = Scanner.nextInt();
Student student = new Student ("123", "John Doe",
        numSubjects);
Student.acceptDetails();
Student.displayDetails();
double SGpa = student.calculateSGPA();
System.out.println ("SGPA:" + SGpa);
  Scanner.close();
}
}
```

(3) Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString() method that could display the complete details of the book. Develop a java program to create n book objects.

→
```
import java.util.Scanner;
class Book {
    private String name;
    private String author;
    private double price;
    private int numpages;
public Book (String name, String author,
double price, int numpages) {
    this.name = name;
    this.author = author;
    this.price = price;
    this.numpages = numpages;
}
Public void setName (String name) {
    this.name = name;
}
Public void setAuthor (String author)
{ this.author = author;
}
Public void setPrice (double price) {
    this.Price = Price;
}
Public void setNumPages (int numPages) {
    this.numpages = numPages;
}
```

```
Public String getName(){
    return name;
}
Public String getPrice(){
    return Price;
}
Public int getNumPages(){
    return numpages; }
@Override
Public String toString(){
return "BOOK Details : \n Name: " + name + "
\n Author : " + author + "\n price : $" + price +
"\n number of Pages : " + numpages;
 }
}
Public class BookMain {
 Public Static void main (String[] args){
Scanner Scanner = new Scanner (System.in);
System.out.print ("Enter the number of
  books:");
int n = Scanner.nextInt();
Book[] books = new Book[n];
for (int i=0; i<n; i++){
System.out.println ("\n Enter details for
  Book" + (i+1) + ":");
Scanner.nextLine();
System.out.print ("Name:");
String name = Scanner.nextLine();
System.out.Print ("
String name = Scanner.nextLine();
System.out.print ("Author:");
String author = Scanner.nextLine();
```

```
System.out.Print("Price : $");
double price = scanner.nextDouble();
System.out.Print("Number of pages : ");
int numpages = scanner.nextInt();
books[i] = new Book(name, author, Price, numpages);
}
System.out.Println("\nDetails of the Books :");
for (Book book : books) {
System.out.Println(book.toString());
System.out.println();
}
scanner.close();
}
}
```

→2 Output:

```
Enter USN : 124
Enter name : teju
Enter credits for Subject 1 : 4
Enter marks for subject 1 : 80
Enter credits for Subject 2 : 3
Enter marks for Subject 2 : 90
Enter Credits for Subject 3 : 2
Enter marks for subject 3 : 70
Enter Credits for subject 4 : 1
Enter marks for Subject 4 : 100
Enter credits for Subject 5 : 85
Enter Credits for Subject 5 : 75
USN : 124
Name : Teju
Credits : [4, 3, 2, 1, 85]
```

Marks : [80, 90, 70, 100, 75]

SGPA : 8.126315789473685

8.1.124

4) Develop a java program to create an abstract class named shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that Each one of the classes extends the class Shape. Each one of the classes contains only the method printArea() that prints the area of the given shape.

→
```
abstract class shape {
    int length;
    int breadth;
    abstract void printArea();
}
class Rectangle Extends shape {
  Rectangle (int length, int breadth) {
      this.length = length;
      this.breadth = breadth;
  }
  void printArea () {
  System.out.Print ("Area of rectangle is" +
                  (length * breadth));
  }
}
class Triangle Extends shape {
  Triangle (int length, int breadth) {
    this.length = length;
    this.breadth = breadth;
  }
  void printArea () {
  System.out.println ("Area of triangle is"
        + (0.5 * length * breadth));
  }
}
```

```
class Circle extends Shape {
    Circle (int length) {
        this.length = length;
    }
    void printArea () {
        System.out.println ("Area of Circle is " +
        (3.14 * length * length));
    }
}
public class main {
    public static void main (String [] args) {
    Rectangle rectangle = new Rectangle (10, 20);
    Rectangle.printArea ();
Triangle triangle = new Triangle (10, 20);
    triangle.printArea ();
Circle circle = new circle (10);
    circle.printArea ();
    }
}
```

Output:

Area of rectangle is 200

Area of triangle is 100.0

Area of Circle is 314.0

5) Develop a java program to create a class Bank that maintains two kinds of account for the customers, one called Savings account and the other current account. The Savings account provide compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

→ Public class Bank {
Public Static void main (String[] args) {
SavingsAccount savingsAccount = new SavingsAccount();
SavingsAccount.deposit(1000);
SavingsAccount.withdraw(500);
System.out.println("Savings account balance:"
+ savingsAccount.getBalance());
CurrentAccount.currentAccount = new CurrentAccount();
currentAccount.deposit(1000);
currentAccount.withdraw(500);
System.out.println("current account balance:"
+ currentAccount.getBalance());
}
}

Class SavingsAccount {
private double balance = 0;
private double interestRate = 0.05;
public void deposit(double amount) {
balance += amount;
}

```
    public void withdraw(double amount) {
        balance -= amount;
    }
    public double getBalance() {
        return balance * (1 + interestRate);
    }
}

class CurrentAccount {
    private double balance = 0;
    private double minimumBalance = 500;
    private double serviceCharge = 50;
    public void deposit(double amount) {
        balance += amount;
    }
    public void withdraw(double amount) {
    if (balance - amount < minimumBalance) {
        balance -= serviceCharge;
    }
    balance -= amount;
}
    public double getBalance() {
        return balance;
    }
}
```

Output:

```
Savings account balance : 525.0
Current account balance : 500.0
```

6) Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:
a) accept deposit from customer and update that balance.
b) Display the balance
c) Compute and deposit interest
d) permit withdrawal and update the balance.

→ import java.util.Scanner;
class Account {
    String CustomerName;
    int accountNumber;
    String accountType;
    double balance;
public Account (String CustomerName,
    int accountNumber, String accountType,
    double balance) {
    this.CustomerName = CustomerName;
    this.accountNumber = accountNumber;
    this.accountType = accountType;
    this.balance = balance;
}
public void deposit (double amount) {
    balance += amount;
System.out.println ("Deposit of " + amount +
    "successful.");
}

```
Public void displayBalance () {
  System.out.println ("Current balance : " + balance);
}

Public void computeInterest (double rate) {
  double interest = balance * (rate / 100);
    balance += interest;
System.out.println ("Interest computed
    and deposited : " + interest);
}

  Public Void withdraw (double amount) {
    if (balance < amount) {
    System.out.Println ("Insufficient balance.");
  } else {
      balance -= amount;
    System.out.println ("withdrawal of " +
        + amount + " Successful.");
    }
  }
}

class CurrentAccount extends Account {
  public currentAccount (String CustomerName,
  int accountNumber, String accountType,
  double balance) {
    super (customerName, accountNumber,
    accountType, balance);
  }
  @override
  public void computeInterest (double rate) {
    System.out.println ("Interest cannot be
      computed for current accounts.");
  }
}
```

```
class SavingsAccount extends Account {
public SavingsAccount (String customerName,
int accountNumber, String accountType,
double balance){
super (customerName, AccountNumber,
accountType, balance);
}
@Override
public void withdraw (double amount){
if (balance < amount){
System.out.println ("Insufficient balance.");
} else if (balance - amount < 1000){
System.out.println("withdrawal amount
~~} else~~ too low.");
} else {
balance -= amount;
System.out.println("withdrawal of "
+ amount + "successful");
}
}
}
Public class Main{
public static void main(String[] args){
Scanner scanner = new Scanner(System.in);
System.out.print("Enter customer name:");
String customerName = scanner.nextLine();
System.out.print("Enter account number:");
int accountNumber = scanner.nextInt();
System.out.print("Enter account type
(current/Savings):");
String accountType = scanner.nextLine();
System.out.print("Enter initial balance:");
double balance = scanner.nextDouble();
```

Account account;
if (accountType.Equals("current")){
account = new CurrentAccount (customerName, accountNumber, accountType, balance);
} else {
account = new SavingsAccount (customerName, accountNumber, accountType, balance);
}
System.out.println("\n Account created Successfully");
System.out.println("customer name:" + account.customerName);
System.out.println("Account number:" + account.accountType);
System.out.println("Initial balance:" + account.balance);
while (true){
System.out.println("\n Select an option:");
System.out.println("1. Deposit");
System.out.println("2. Display balance");
System.out.println("3. Compute and deposit interest");
System.out.println("4. withdraw");
System.out.println("5. Exit");
int choice = Scanner.nextInt();
switch (choice) {
case 1:
System.out.print("Enter amount to deposit:");
double depositAmount = Scanner.nextDouble();
account.deposit (depositAmount);
break;
case 2:

```
        account.displayBalance();
        break;
    case 3:
        System.out.print("Enter interest rate: ");
        double interestRate = scanner.nextDouble();
        account.computeInterest(interestRate);
        break;
    case 4:
        System.out.print("Enter amount to
                          withdraw: ");
        double withdrawAmount = scanner.
                                nextDouble();
        account.withdraw(withdrawAmount);
        break;
    case 5:
        System.exit(0);
    default:
        System.out.println("Invalid choice.");
    }
  }
}
```

05/02/24

19/2/24

Date
Page

(5) Develop a Java Program to create a class Bank that maintains two kinds of account for it customers, one called Savings account and the other current account. The Savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed
create a class account that stores customer name, account number and type of account. From the derive the classes current account and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:
(a) Accept deposit from customer and update the balance.
(b) Display the balance
(c) Compute and deposit interest
(d) permit withdrawal and update the balance

→
```
import java.util.Scanner;
class account
{

String name;
int accno;
string type;
double balance;
account (string name, int accno,
string type, double balance)
```

```
{
    this.name = name;
    this.accno = accno;
    this.type = type;
    this.balance = balance;
}
void deposit (double amount)
{
    balance += amount;
}
void withdraw (double amount)
{
    if ((balance - amount) >= 0)
    {
        balance -= amount;
    }
    else
    {
        System.out.println("insufficient balance;
                Cant withdraw");
    }
}
void display()
{
System.out.println("name : "+name+"accno:"+accno+
    "type:"+type+"balance:"+balance);
}
}
class SavAct Extends account
{
private static double rate=5;
SavAct (String name, int accno, double balance)
```

```
{
super (name, accno, "savings", balance);
}
void interest ()
{
balance += balance * (rate)/100;
System.out.println ("balance:" + balance);
}
}
class current extends account
{
private double minBal = 500;
private double ServiceCharges = 50;
current (String name, int accno, double balance)
{
super (name, accno, "current", balance);
}
void checkmin ()
{
if (balance < minBal)
{
System.out.println ("balance is less than min
balance, Service charges imposed :" + ServiceCharges
balance -= ServiceCharges;
System.out.println ("balance is :" + balance);
}
}
}
class accountMain
{
public static void main (String a[])
{
```

```
Scanner S = new Scanner (System.in);
System.out.println ("Enter the name:");
String name = S.next();
System.out.println ("Enter the type (Current/
                    Saving):");
String type = S.next();
System.out.println ("Enter the account numbe:");
int accno = S.nextInt();
System.out.println ("Enter the initial
                    balance:");
double balance = S.nextDouble();
int ch;
double amount1, amount2;
account acc = new account (name, accno, type, balance);
SavAcct sa = new SavAcct (name, accno, balance);
CurAcct ca = new CurAcct (name, accno,
                    balance);
while (true)
{
  if (acc.type.Equals ("savings"))
{
System.out.println ("1.menu\n1. deposit
2. withdraw 3. compute interest 4. dsplg");
System.out.println ("Enter the choice:");
  ch = S.nextInt();
  switch (ch)
{
case 1: System.out.println ("Enter the amount:");
  amount1 = S.nextInt();
  sa.deposit (amount1);
  break;
```

```
case 2: System.out.println("Enter the amount:");
    amount2 = s.nextInt();
    sa.withdraw(amount2);
    break;
case 3: sa.interest();
    break;
case 4: sa.display();
    break;
case 5: System.exit(0);
default: System.out.println("invalid input");
    break;
}
}
else
{
System.out.println("\n menu \n1: deposit 2.
    withdraw 3. display");
System.out.println("Enter the choice:");
    ch = s.nextInt();
    switch(ch)
    {
case 1: System.out.println("Enter the amount:");
    amount1 = s.nextInt();
    ca.deposit(amount1);
    break;
case 2: System.out.println("Enter the amount:");
    amount2 = s.nextInt();
    ca.withdraw(amount2);
    ca.checkmin();
    break;
case 3: ca.display();
    break;
```

```
case 4 : System.exit(0);
default : System.out.println("Invalid
          input");
break;
}
}
}
```

→ output:

```
Enter the the name :
shenba
Enter the type (current (savings)):
current
Enter the Account number :
167
Enter the initial balance :
45000

menu
1. deposit 2. withdraw 3. deplay 4. Exit
Enter the choice:
2
Enter the amount:
10000

menu
1. Deposit 2. withdraw 3. Deplay 4. Exit
enter the choice:
4. Exit
```

12/02/24

(6) Create a package CIE which has two classes Student and Internals. The class Student has members like usn, name, sem. The class Internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

```
package cie;
import java.util.Scanner;
public class Student {
String usn;
String name;
int Semester;
public void Studdetails(){
Scanner S1 = new Scanner(System.in);
System.out.println("Enter the USN of student");
usn = st.next();
System.out.println("Enter the name of Student");
Semester = S1.nextInt();
}

public void printdetails(){
System.out.println("the USN of student is"+usn);
System.out.println("the name of student is"+name);
System.out.println("the Semester of Student is"+Semesters);
}
}
```

```
Package cie;
import java.util.Scanner;
Public class internals extends student{
Public float studmarks[] = new float[5];
Public void Studmarks(){
Scanner sI = new Scanner (System.in);
System.out.println("Enter the internal marks
    of student");
for (int i=0; i<5; i++){
studmarks[i] = sI.nextFloat();
}
}
 Public void printmarks(){
System.out.println("the marks of student is");
  for (int i=0; i<5; i++){
studmarks[i] = sI.nextFloat();
}
}
public void printmarks(){
System.out.println("the marks of Student is");
for (int i=0; i<5; i++){
System.out.println(studmarks[i]);
}
}
}

import java.util.Scanner;
import cie.student;
import cie.internals;
import see.External;
class main{
public static void main (String xx[]){
int n;
```

```
Scanner s3 = new Scanner (system.in);
System.out.println ("Enter the number of studens");
 n = s3. nextInt ();
internal internalmarks [] = new internals[n];
External Externalmarks[] = new External[n];
 for (int i=0; i<n; i++) {
internalmarks [i] = new internals ();
External marks [i] = new external();
internalmarks [i] . Studdetails ();
internalmarks [i] . studmarks ();
 Externalmarks [i] . seemarks ();
 }
 float finalmarks [][] = new float [n][5];
 for (int i=0; i<n; i++){
 for (int j=0; j<5; j++){
finalmarks[i][j] = internalmarks [i]. studmarks[i]+
 (( External marks [i]. seemarks [i]/2);
}
}
 for (int i=0; i<n; i++) {
internal marks [i] . printdetails ();
internal marks [i]. printmarks ();
External marks [i]. printseemarks ();
}
 for (int i=0; i<n; i++) {
internalmarks [i]. printdetails();
internalmarks [i]. printmarks ();
Externalmarks[i]. printseemarks ();
 }
}
```

Output

Enter the number of Students.
2
Enter the usn of Students.
1
Enter the name of Students.
a
Enter the Semester of Student
3
Enter the Internal mark of Student.
40
40
40
40
40
Enter the See marks of Student.
80
80
80
80
80
Enter the usn of Student
2
Enter the name of Students.
b
Enter the Semester of Studen
3

19/02/24

(7) write a program that demonstrate handling of exception in Inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge() when the input age > father's age.

→
```
class wrongage extends Exception {
public wrongage (String message){
 super (message);
}
}

class father {
private int age;
public father (int age) throws wrongage {
if (age < 0) {
throw new wrongage ("age cannot be negative");
}
 this.age = age;
 }

public int getage (){
 return age;
}
}

Class Son Extends father {
private int Sonage;
public Son (int fatherage, int Sonage) throws
wrongage {
super (fatherage);
if (Sonage >= fatherage){
throw new wrongage ("Son's age cannot be
```

greater than or Equal to father's age");
}
this.Sonage = Sonage;
}
public int getSonage() {
return Sonage;
}
}
public class exceptioninheritance {
public static void main(String xx[]) {
try {
Son son1 = new Son(40, 20);
System.out.println("father's age:" + son1.getage());
System.out.println("Son's age:" + son1.getSonage());
Son invalidSon = new Son(30, 35);
System.out.println("the line will not be reached");
}
catch (wrongage e) {
System.out.println("Exception:" + e.getMessage());
}
}
}

→ Output

father's age : 40
Son's age : 20
Exception : Son's age cannot be greater than or Equal to father's age.

19/02/24