Page No.
Date / /

Q) Write a program to create a class student with members USN, name, marks (6 subject). Include methods to accept student details and marks also include a method to calculate the percentage and display appropriate details.

Code :-

```
import java.util.Scanner;
class Student
{
private int usn;
private String name;
private int marks[];
Student(int usn, String name)
{
this.usn = usn
this.name = name
this.mark = new int[6];
}
public void accept()
{
Scanner input = new Scanner(System.in);
System.out.println("enter the name");
this.name = input.next();
System.out.println("enter the usn");
this.input = input.nextInt();
System.out.println("enter marks of all subject");
```

```
for(int i=0;i<6;i++)
{
  System.out.println("enter the marks of "+(i+1)+
"subject");
this.marks[i] = input.nextInt();
}
}
public double calpercentage()
{
  int totalmarks=0;
for(int i=0;i<6;i++)
{
  totalmarks = totalmarks + marks[i];
}
double percentage = totalmarks/6;
return percentage;
}
void display()
{
System.out.println("the name is" +this.name)
sop("the.usn is" + this.usn);
}
}
```

```
class main
{
  PSVM(String args[])
{
  SOP("no; of students");
  Scanner input = new Scanner(System.in)
  int num = input.nextInt();
  Student[] s = new Student[num];
  for(int i=0; i<num; i++)
  {
  SOP("enter the details");
  s[i] = new Student(0," ");
  s[i].accept;
  }
  SOP("the details are");
  for(int i=0; i<num; i++)
  {
  s[i].display;
  SOP("n...");
  }
  }
  }
```

Output:

Enter the number of students

2

Enter the details of 1 student

Enter the name

gana

Enter USN

1

marks

30

60

60

60

60

60

Enter the details of 2 student

name

hana

USN

2

marks

70

70

70

To
The details of student is
The name is Gana
The age is 1
the percentage is 55.0%

The name is hans
the age is 2
The percentage is 70.0%

② Create a class Book that 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.

Code:

```
import java.util.Scanner
class Book
{
String name;
String author;
int price;
int num_pages;
```

```
Book(){}
Book(String name, String author, int price, int pages)
{
  this.name = name;
  this.author = author;
  this.price = price
  this.num_pages = num_pages
}
void set()
{
  Scanner input = new Scanner(System.in);
  sop("Enter the name")
  this.name = input.nextLine();
  sop("author");
  this.author = input.nextLine();
  sop("no; of pages");
  this.num_pages = input.nextInt();
  sop("price");
  this.price = input.nextInt();
}
Public String toString()
{
  String name, author, price, num_pages;
  name = "Book name is" + this.name + "\n";
```

author = "author name is " + this.author + "\n";
price = " price is " + this.price + "\n"
num_pages = "no of pages is " + this.num_pages +
"\n";
return name + author + price + num_pages;
}
}
class main
{
public static void main(String args[])
{
Scanner input = new Scanner(System.in);
System.out.println("no. of books");
int num = input.nextInt();
Book[] b = new Book[num];
for (int i=0; i<num; i++)
{
b[i] = new Book();
b[i] = b[i].read();
}
for (int i=0; i<num; i++)
{System.out.println("book " + b[i]);
}
}
}

Output

Enter the number of books

2

Enter the name of the book

Book

author

Gana

number of page

200

price

500

Name

the book allowed

author

hana

page

200

price

30000

book1 : Book name is the book

author name is gana

price is 500

number of page is 200

book 2 : ii the book class
author name is hari,
price is 3000,
number of pages is 900.

Quadratic equation:

```
class Quadratic
{
    double a, b, c;
    d=4*a*b*c;
    if d>=0;
    Quadratic (a, b, c)
    this.a=a;
    this.b=b;
    this.c=c;
    }
    d = b*b - 4ac;
void calc()
    {
    double d= b*b-4*a*c;
    if (d>0.0)
    {
    double ll= (-b + Math.pow(d, 0.5)
    }
else if (d==0.0)
```

```
{
double r1 = b1/(2*a);
SOP("The root is r1" + r1);
}
else
{
SOP("The roots are not real");
}
}
else main
{
psvm (String args[])
{
Quadratic result = new Quadratic(2,3,1);
result.calc();
}
}
```

output

The roots are not real

4) Shape:

```
import java.util.Scanner;
abstract class Shape{
    protected int dimension1;
    protected int dimension2;
    public Shape(int dimension1, int dimension2)
    {
        this.dimension1 = dimension1;
        this.dimension2 = dimension2;
    }
    public abstract void printArea();
}

class Rectangle extends Shape{
    public Rectangle(int length, int width)
    {
        super(length, width);
    }
    public void printArea(){
        int area = dimension1 * dimension2;
        SOP(area);
    }
}
```

class Triangle extends Shape{
printArea()
SOP(area);
}
class circle extends Shape{
printArea()
{SOP()}
}

class Main{
psvm(String[] args)
{
Rectangle rectangle = new Rectangle(Ex);
rectangle.printArea()
Triangle triangle = new Triangle(5,6);
triangle.printArea();
Circle circle = new Circle(7);
circle.printArea();
}
}

Output
Area of Rectangle = 20
Area of Triangle = 9
Area of circle = 153.94.

5) Bank Java.

```
import java.util.Scanner;

public class Bank {
    psvm(S a)
    Scanner sc = new Scanner(System.in)
    SOP(Welcome to Bank);
    SOP(Name);
    SOP(Enter the account number);
    String savingsAccount = sc.nextLine();
SavingsAccount savingsAccount = new
    SavingsAccount(savingsAccountName, number);
    SOP(Enter the number of current account)
Current Account = new CurrentAccount(Name,
number).
    SOP('specify on savings account)
    SOP(Enter the amount to deposit);
    double savingsDeposit(sc.nextDouble());
savings account.displayBalance
```

```
double currentDeposit (currentDepositA
-mount);
current Account. display balance();
current Account. display balance();
JOP( Amount to withdraw);
  double withdraw = nextDouble();
  current. display balance());
  scanner. close();
}
class Account {
  String customerName;
  String accountnumber
  double balance
public void deposit (double amount){
    balance += amount.
  }
public void displayBalance()
    JOP( balance)
}
class SavingsAccount extends Account {
    (if balance > amount)
      balance = amount)
  }
```

class CurrentAccount extends Account {
private double minimumBalance = 1000;
void withdraw(double amount) {
if (balance - amount >= minimumBalance)
balance -= amount;
super.withdraw();
else
System.out.println("Insufficient");
imposeServiceCharge();
}
}
private void imposeServiceCharge() {
double serviceCharge = 200;
balance -= serviceCharge;
System.out.println(serviceCharge)
}

Output:-
Welcome to the Bank!
Enter name for Account (Savings / current): Gaurav
Enter account number: 1001
current account no.: 1001

performing operation on 'savings Account':
Enter amount to deposit into savings Account: 5987.0
successful
Enter the amount to withdraw: 342
successful
Balance: 5645.
performing operation on Current, deposit into current account: 56990
Successful
withdraw: 5679.0
successful
Balance: 51312.0

SSD
2nd/11/24.

Package

Student Create a package CIE which has two class - Student and Internals. The class Student has members like usn, name, sem. The class internals derived from student has an array that stores internal mark scored in five courses of the current semester of the student. Create another package SEE which has class Student.

Student

```
package cie;
public class Student {
    public String name;
    public String usn;
    public int sem;
}
```

Internals

```
package cie;
public class Internals extends Student {
    public int[] marks = new int[5];
}
```

External

```
package sec;
import cie.sheet;
public class External extends Student {
public int[] semarks = new int[5];
}
```

Main

```
import cie.Internals;
import sec.External;
import java.util.Scanner;

public class Main{
  psvm(String[] args)
  SOP("Enter the number of students");
  Scanner input = new Scanner(System.in);
  int n = input.nextInt();
  Internals[] s1 = new Internals[n];
  External[] s2 = new External[n];
  int[] finalcie = new int[n];
  int[] finalsee = new int[n];
  for(int i=0; i<n; i++){
  s1[i] = new Internals();
  System.SOP(name)
```

```
JOP( USN )
JOP(JPM)
JOP(Marks 15 subject)
for (i=0 i<5 i++)
  {
    s[i]_marks = input.nextInt();
    final cie[i] += s1[i] marks[j];
  }
}
for(int i=0; i<n i++)
  JL[i] = new Label();
JOP(name)
for (USN)
for(sem)
for( marks)
for(int j=0; j<9 ; j++)
  {
    s[i].sc.(marks[j]) = input.nextInt()
    final ar.(i) = s(i) .sem marks[j];
  }
}
JOP
for(int i=0; i<n; i++)
{
  JOP( name + s[i].name + USN + s[i].usn
```

```
,em + s[i].rem);
SOP("Internal marks" + finalcie[i]);
SOP("External marks" + finalsee[i]);
SOP("Total marks" + (finalcie+finalsee));
}
}
}
```

Output

Enter the number of students

1

Enter the name

aa

Enter the usn

23

Enter the sem

3

Enter the marks of 5 subjects

10

10

10

10

10

Enter the name

aa

Enter the usn

<3
Enter the no
3
mark
20
20
20
20
Rc2
20
Name : aa USN : 23 Sem : 3
Internal mark : 50
External mark : 202
Total mark : 352

9) Write a program which creates two threads one thread displaying BMS college of Engineering once every 10 second and CSE every 10 second

Code:

```
class BMSCE extends Thread {
    public void run() {
        while (true) {
            SOP ("BMS college of Engineering");
            try {
                Thread.sleep(10000); }
```

```
catch (InterruptedException e){
    SOP(e);
  }
 }
}}
class CSE extends Thread{
  public void run(){
   while(true){
     SOP("CSE");
   try{
     Thread.sleep(2000);
   }catch(InterruptedException e){
     SOP(e);
  }}}}
class run{
  PSVM(String[] args)
  BMSCE b = new BMS();
  CSE c = new CSE();
  b.start();
  c.start();
 }}
```

Output

```
BMS college of Engineering
CSE
CSE
CSE
```

CSE
BMS College of Engineering

Q) Exception handling

class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}

class Father {
    private int age;
    public Father(int age) throws WrongAgeException {
        validateAge(age);
        this.age = age;
    }
    private void validateAge(int age) throws
    WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Ent
");
        }
    }
}

```
class Son extends Father {
    int sonAge;
    public Son(int fatherAge, int sonAge) throws
WrongAgeException {
        super(fatherAge);
        validateSonAge(sonAge);
        this.sonAge = sonAge;
    }
    void validateSonAge(int sonAge) throws
WrongAgeException {
        if(sonAge >= getAge()) {
            throw new WrongAgeException();
        }
    }
    public int getSonAge() {
        return sonAge;
    }
}
class Test {
    public static void main(String[] args) {
        try {
            Father father = new Father(45);
        } catch(Exception e) {
            System.out.println(e);
        }
    }
}
```

```
try {
    Son son = new Son(30, 35);
}
catch (WrongAgeException e) {
    System.out.println();
}
}
}
```

Output:

Age cannot be negative
Child Age cannot be greater than or equal to father's age.

Date 9/2/24