B.M.S COLLEGE OF ENGINEERING BENGALURU

Autonomous Institute, Affiliated to VTU

LAB REPORT

23CS3PCOOJ

Submitted in partial fulfilment of the requirements for Lab

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:

PIYUSH BELLUBBI (1BM22CS192)

Department of Computer Science and Engineering, B.M.S

College of Engineering,

Bull Temple Road, Basavanagudi, Bangalore, 560 019 2023-2024.

INDEX

Sl.No.	Title	Date
1	Complete scanned Observation Book	12/12/2023 - 20/02/2024
2	Lab 1	12/12/2023
3	Lab 2	19/12/2023
4	Lab 3	26/12/2023
5	Lab 4	02/01/2024
6	Lab 5	09/01/2024
7	Lab 6	16/01/2024
8	Lab 7	23/01/2024
9	Lab 8	30/01/2024
10	Lab 9	06/02/2024
11	Lab 10	20/02/2024

12/12/23 OOJ LAB OBSERVATION

•) Parse Int method

→ Write a program in Java to find the area of a rectangle and verify the same with various inputs (length, breadth).

code:
```
class RectangleArea {
public static void main (String args[] ) {
 int length , breadth ;
length = Integer. parseInt (args [0]);
breadth = Integer. parseInt (args [1]);
int area = length * breadth ;
System . out . println (" length of rectangle = " + length );
System . out . println (" breadth of rectangle = " + breadth);
System . out . println (" Name = PUSHBELLVBBI  USN = 1BM22CS192 ");
 }}
```

•) Scanner

→
```
import java . util . Scanner ;
class HelloWorld {
public static void main( String args [])
{
  int a; float b; String s ;
  Scanner in = new ~~String~~ Scanner (System . in);
  System . out . println ("Enter a string ");
  s = in . nextline ();
  System . out . println ("You entered a string " + s);
  System . out . println ("You entered an integer " + a);
  ~~a = in . nextint();~~  a = in . nextint();
  System . out . println (" Enter a float value ");
```

```
system.out.println("Your entered string is " + s);
System.out.println("Enter an integer");
a = in.nextInt();
System.out.println("You entered an integer" + a);
System.out.println("Enter a float value");
b = in.nextFloat();
System.out.println("Your float value is " + b);
```

- Array

```
class AutoArray {
public static void main(String Args[]) {
int month_days[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
System.out.println("April has" + month_days[3] + " days.");}
}
```

- Factorial

```
class factorial {
    public static void main (String args[])
    {
        int fac = 1;
        System.out.println("Enter a number: ");
        int n = sc.nextInt();
        for (int i = 1; i<= n; i++) {
            fac = fac * i;
        }
        System.out.println("The factorial : " + fac);
    }
}
```

⇒ Palindrome → WAP to find the given 5 digits int is a palindrome or not.

→
```
class palindrome {
public static void main (String args []){
int n , t, rem, rev = 0;
Scanner sc = new Scanner (System.in);
System.out.println ("Enter a 5 digit number : ");
n = sc.nextInt ();
 t = n ;
while (t > 0){
        rem = t % 10;
        rev = rev * 10 + rem;
        t = t/10;
      }
      if (rev == n) {
      System.out.println (" Palindrome ");
      }
      else {
      System.out.println (" not palindrome ");
      }
  }
}
```

→ Sum of digits: WAP to find the sum of digits in a 5 digit number.

```
class sumofdigits {
    public static void main (String args[]) {
        long number, sum;
        Scanner sc = new Scanner (System.in);
        System.out.println("Enter a 5 digit number: ");
        number = sc.nextlong();
        for (sum=0; number!=0 ; number = number/10) {
            sum = sum + number % 10;
        }
        System.out.println("Sum of digits: "+sum);
    }
}
```

→ Conversion

```
class Conversion {
public static void main (String args[]) {
 byte b=1; short s1 = 1000, s2;
 int i1 = 100000, i2;
 long l1 = 1000000, l2;
 char c='+';
 float f1= 25.69f, f2;
 double d1 = 536987.125, d2;
System.out.println(b+" "+s1+" "+i1+" "+l1+" "+c+" "+f1+"
s2= b;
 i2 = s1;
 l2 = i1;
System.out.println(s2+" "+i2+" "+l2);
  }
}
```

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 discriminate b^2-4ac is negative, display a message stating that there are no real solutions.

code:

```
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 coefficients 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 & equal");
                System.out.println("Root 1 = Root 2 = " + 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 & 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("Root2 = " +r2+ " -i " +r2);
            }
    }
}

class Quadratic Main
{
    public static void main(String args[])
    {
        Quadratic q = new Quadratic();
        q.getd();
        q.compute();
    }
}
```

o) Output:

i) Enter the coefficients of a, b, c

3

4

5

Roots are imaginary

Root1 = 0.0+ i1. 1055415967

Root2 = 0.0 - i1. 1055415967

ii) Enter the coefficients of a, b, c

2

4

2

Roots are real and equal

Root1 = Root2 = -1.0

iii) Enter the coefficients of a, b, c

2

6

2

Roots are real and distinct

Root1 = -0.381966

Root2 = -2.61803

12/12

iv) Enter the coefficients of a, b, c

0

1

1

Not a quadratic equation

LAB:2 19/12

```
import java.util.Scanner;
class subjects {
int subject marks, credit, grade; }
class student {
    string name;
    String USN;
    double SGPA;
    Scanner S;
    Subject subjects [];

student()
{
    int;
    subjects = new subject[9];
    for (i=0; i<8 ; i++)
    subjects [i] = new subject();
    s = new Scanner(system.in);
}
public void getstudentDetails () {
  system.out.println("Enter Student name}
  name = S.nextLine();

System.out.println("Enter student USN);
USN = S.nextLine() ; }

public void getMarks() {
int i;
for (i=0; i<8 ; i++) {
System.out.println("Enter Marks of subjects "+ (i+1) +"
Subjects [i].subject Marks = s.nextInt();
```

2

```
if Subjects[i].Subject Marks>= 40 && Subject Marks<=
100){
Subject[i].grade = CalculateGrade
(Subject[i].Subject Marks); }
else {
System.out.println(" Invalid Marks. Marks should be
between 40 & 100");
}
System.out.println("Enter Credits ");
Subjects[i].Credits = S.nextInt();
}
}

public int calculateGrade(int marks){
if (marks >=90)
return 10;
else if (marks >=70 && marks<=80)
return 9;
else if (marks >=60 && marks <=70)
return 8;
else if (marks >= 50 && marks <=60)
return 7;
else
return 6;
}

public void computeSGPA(){
int totalScore =0;
int totalCred = 0;
for (int i=0; i<8; i++){
    totalscore += subjects[i].grade * subjects[i].
                Credits;
```

```
        total cred + = subjects [i]. Credits ;
    }
    SGPA = (double) total score / (double) total cred;
    }
}
public class std {
    public static void main (String args [])
    {
        Student SL = new Student ();
        SL. getStudentDetails ();
        SL. getMarks ();
        SL. ComputeSGPA ();
        System. out. println ("student name: " +SL.name);
        System. out. println ("student USN " +SL USN);
        System. out. println ("student SGPA-, " +SL. SGPA);
    }
```

=> Output:

```
Enter student name :
Manya
Enter student USN:
109283
Enter Marks of subject:
88
Enter Credits:
8
Enter marks of subject 2;
78
Enter credit
7
Enter Marks of subject 3;
89
```

Date
Page

Enter Credits:
9
Enter Marks of subject 4:
78
Enter Credits:
7
Enter marks of subject 5:
88.
Enter credits:
8
Enter marks of subject 6:
67
Enter Credits:
7
Enter Marks of subject 7:
78
Enter Credits:
7
Enter marks of subject 8:
78
Enter Credits:
7
Student name: ~~Manya~~ Manya
Student USN: 109283
Student SGPA 7.62

```
Enter student name:
manya
Enter Student USN:
109283
Enter marks of subject1:
88
enter credits:
8
Enter marks of subject2:
78
enter credits:
7
Enter marks of subject3:
89
enter credits:
9
Enter marks of subject4:
78
enter credits:
7
Enter marks of subject5:
88
enter credits:
8
Enter marks of subject6:
67
enter credits:
6
Enter marks of subject7:
78
enter credits:
7
Enter marks of subject8:
78
enter credits:
7
Student name:manya
Student usn:109283
Student sgpa:7.627118644067797
```

LAB 3: 26/12

```
import java.util.Scanner;

class Book{
    String name;
    String author;
    int numPages;
    int price;

  Book (String name, String author, int price, int
        numPages){
    this.name = name;
    this.author = author;
    this.price = price;
    this.numPages = numPages;
}

  public String toString(){
    String bookdetails = "Book name:" + this.name + "\n"+
                         "Author name:" + this.author + "\n"
                         "Price:" + this.price + "\n"+
                         "Number of Pages:" + this.numPages
                         "\n";
    return bookDetails;
  }
}

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

    System.out.print("Enter the number of books: ");
```

```
int n = s.nextInt();

Book [] books = new Book[n];

for (int i=0; i<n; i++){
    System.out.println("Enty details for Book" +
    (i+1));
    System.out.println("Name: ");
String ~~System.out~~ name = s.next();
    System.out.print("Author: ");
    String author= s.next();
    System.out.print("Price: ");
    int price = s.nextInt();
    System.out.print("Number of pages: ");
    int numPages = s.nextInt();

    books[i] = new Book(name, author, price, numPages);
}
System.out.println("\nDetails of the books:");
for (int i=0; i<n; i++)
    {
        System.out.println("Book" + (i+1) + ":\n" +
        books[i].toString());
    }
}
}
```

-> Output:

Enter the number of books: 2
Enter details for Books 1
Name: Coding
Author: pp
Price: 798
Number of pages: 260
Enter details for Book 2
Name: Cooking
Author: KP
Price: 798
Number of Pages: 300

Details of the books 1:

Book 1:
Book name: Coding
Author name: pp
Price: 798
Number of pages: 260

Book 2:
Book name: Cooking
Author name: KP
Price: 798
Number of pages: 300

26/12/23

```
Enter the number of books: 2
Enter details for Book 1
Name:
kumar
Author:
hsgdy
Price:
124
Number of pages:
500
Enter details for Book 2
Name:
345
Author:
lkis
Price:
1234
Number of pages:
7000

Details of the books:
Book 1:
Book name: kumar
Author name: hsgdy
Price: 124
Number of pages: 500

Book 2:
Book name: 345
Author name: lkis
Price: 1234
Number of pages: 7000
```

LAB 4:

Date 02/01/24
Page

```
import java.util.Scanner;
class InputScanner
 {
     Scanner s = new Scanner (System.in);
     double getInput (String prompt){
     System.out.println (prompt);
      return s.nextDouble (); }
 }
abstract class Shape extends InputScanner{
 double side1, side2;
abstract void area();
}

 class Rectangle extends Shape {
 Rectangle ()
       {
         side1 = getInput ("Enter length of rectangle: ");
         side2 = getInput ("Enter breadth of rectangle: ");
       }
 void area()
       {
         double area = side1 * side2;
         System.out.println ("Area of rectangle = " +area);
       }
}
 class Triangle extends shape {
 Triangle ()
       {
         side1 = getInput ("Enter base of the triangle: ");
         side2 = getInput ("Enter height of the triangle: ");
       }
```

```
    void area ()
        {
            double, area = side1 * side2 / 2;
            System.out.println ("Area of the Triangle = "+area
        }
}
  class circle extends shape {
  circle ()
        {
            side1 = getInput ("Enter the radius of the
        }                          circle : ");
  void area()
        {
            double. area = Math. PI * side1 * side1;
            System .out. println ("Area of the circle = "+area
        }
}

  class Main1{
{       public static void main (String Args[])
        {
            Rectangle rectangle = new Rectangle ();
            triangle Triangle = new triangle ();
            circle Circle = new circle ();

            rectangle. area();
            Triangle. area();
            Circle. area ();
        }
}
```

Date
Page

→ Output:

Enter the length of rectangle : 20
Enter the breadth of rectangle : 40
Enter the height of triangle : 8
Enter the base of triangle : 6
Enter the radius of circle : 4

Area of the Rectangle = 800.0
Area of the Triangle = 24.0
Area of the Circle = 50.2654

02/01/24

```
Enter length of rectangle:
20
Enter breadth of rectangle:
40
Enter base of the triangle:
6
Enter height of the triangle:
8
Enter the radius of the circle:
4
Area of the Rectangle =800.0
Area of the triangle=24.0
Area of the circle=50.26548245743669
```

LAB-5 09.01.2024

```
import java.util.Scanner;

class Account{
    private String customerName;
    private String accountNumber;
    private String accountType;
    protected double balance;

    public Account(String customerName, String 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("Amount deposited successfully
        Current balance: " + balance);
    }

    public void displayBalance(){
        System.out.println("Account Type: " + accountType)
        System.out.println("Customer Name: " + customerNa
        System.out.println("Account Number: " + accountNum
        System.out.println("Current Balance: " + balance);
    }
}
```

Date
Page

```
class SavingsAccount extends Account {
    private double interestRate;

    public SavingsAccount(String customerName, String
    accountNumber, double balance, double interestRate){
    super(customerName, accountNumber, "Savings", balance);
    this.interestRate = interestRate;
}

public void computeAndDepositInterest() {
    double interest = balance * interestRate /100;
    deposit(interest);
    System.out.println("Interest computed and deposited.
                        Current balance: " + balance);
}

public void withdraw(double amount){
    if (balance >= amount){
        balance -= amount;
    System.out.println("~~Insufficient funds. Withdrawal~~
      ~~failed.~~"); ("Amount withdrawn successfully.
    ( current balance:" + balance); }
    else {
    System.out.println("Insufficient funds. Withdrawal
                        failed.");
    }
}
}

class CurrentAccount extends Account {
    private static final double SERVICE_CHARGE = 250;
    private double minimumBalance;
```

```
public CurrentAccount( ~~extends Account~~ String
        customerName, String accountNumber, double
        balance, double minimumBalance ) {
        ~~this.minimum~~
        super (customerName, accountNumber, "Current",
        balance);
        this.minimumBalance = minimumBalance; }
public void withdraw (double amount) {
        if (balance - amount >= minimumBalance) {
            balance -= amount;
            System.out.println("Amount withdrawn
            successfully. Current balance: " + balance);
        } else {
            System.out.println("Withdrawal failed.
                Below minimum balance.");
            imposeServiceCharge();
        }
    }
    private void imposeServiceCharge() {
        if (balance < minimum balance) {
        System.out.println("Service charge of" +
                : SERVICE_CHARGE + " imposed.");
        balance -= SERVICE CHARGE;
        System.out.println("Service charge not imposed.
                Balance is still above the minimum.");
        }
    }
}

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

Date
Page

```
System.out.println("Enter customer name: ");
String customerName = scanner.nextLine();

System.out.println("Enter account number: ");
String accountNumber = scanner.nextLine();

System.out.println("Enter initial balance: ");
double initialBalance = scanner.nextDouble();

SavingsAccount savingsAccount = new SavingsAccount(customerName, accountNumber, initialBalance,
7.5);

CurrentAccount currentAccount = new CurrentAccount(customerName, accountNumber, initialBalance,
500);

int choice;
do {
System.out.println("\n Select an option: ");
System.out.println("1. Deposit to savings account");
System.out.println("2. Compute and Deposit Interest
                    for Savings account");
System.out.println("3. Withdraw from savings
                    account");
System.out.println("4. Deposit to current Account");
System.out.println("5. Withdraw from Current
                    Account");
System.out.println("6. Display Balance");
System.out.println("0. Exit");
System.out.println("Enter your choice: ");
choice = scanner.nextInt();
```

```
switch (choice) {
        case 1:
System.out.println("Enter the amount to
  deposit to Savings Account : ");
double savingsDepositAmount = scanner.nextDouble();

savingsAccount.deposit(savingsDepositAmount);
break;
        case 2:
savingsAccount.computeAndDepositInterest();
        break;
        case 3:
System.out.println("Enter the amount to
  withdraw from Savings Account: ");
        double savingsWithdrawAmount =
        Scanner.nextDouble();
savingsAccount.withdraw(savingsWithdrawAmount);
        break;
        case 4:
System.out.println("Enter the amount to deposit
  to current Account : ");
double currentDepositAmount = scanner.nextDouble();

currentAccount.deposit(currentDepositAmount);
        break;
        case 5:
System.out.println("Enter the amount to withdraw
  from Current Account: ");
currentWithdrawAmount = scanner.nextDouble();

currentAccount.withdraw(currentWithdrawAmount);
        break;
```

```
            case 6:
        System.out.println("\n Savings Account Details: ");

        savingsAccount.displayBalance();

        System.out.println("\n Current Account Details: ");

        currentAccount.displayBalance();
            break;
            case 0:
        System.out.println("Exiting the program. Goodbye!");
            break;
            default:
        System.out.println("Invalid choice. Please enter a
            valid choice.");
        }
    } while (choice != 0);
  }
}
```

*) Output:-

```
Enter customer name:
Piyush
Enter account number:
1
Enter initial balance:
45000

Select an option:
1. Deposit to Savings Account
2. Compute and Deposit Interest for Savings Account
```

3. Withdraw from Savings Account
4. Deposit to Current Account
5. Withdraw from Current Account
6. Display Balance
0. Exit
Enter your choice : 2
Amount deposited successfully. Current Balance : 48375.0
Interest computed and deposited. Current Balance : 48375.0

Select an option:
1. Deposit to Savings Account
2. Compute and deposit Interest for Savings Account
3. Withdraw from Savings Account.
4. Deposit to Current Account
5. Withdraw from Current Account
6. Display Balance
0. Exit
Enter your choice : 3
Enter the amount to withdraw from Savings Account :
45000
Amount withdrawn successfully. Current Balance : 3375.0

Select an option:
1. Deposit to Savings Account
2. Compute and deposit Interest for Savings Account
3. Withdraw from Savings Account
4. Deposit to Current Account.
5. Withdraw from Current Account
6. Display Balance.
0. Exit
Enter your choice : 6

Date
Page

Savings Account Details:
Account Type: Savings
Customer Name: Piyush
Account Number: 1
Current Balance: 30375.0

Current Account Details:
Account Type: current
customer Name: Piyush
Account Number: 1
Current Balance: 45000.0

WEEK - 6 Strings 16/10/24

4. Extracted Substring : Bmsce

2. String 1 : Hello World!
Length of String 1 : 13

String 2 : Java

String 3 : Hello
String 4 : Java!
Concated String : Hello Java!

3. Original String : Hello, toString!
String Representation using toString() : Hello, toString!

5. Expected Output :

Original String : Hello! GetBytes!
Byte Array : 72, 101 108 108 111 44 32 71 101
116 66 121 116 101 115 33

Original String : Hello, To CharArray!
Character Array : H e l lo , To Char Array!

6. Bmsce equals Bmsce → True
Bmsce equals BMSCE → False
Bmsce equals College → False
Bmsce equalsIgnoreCase BMSCE → True

7. Substring is matched!

Date
Page

8. Does the string start with 'Hello'? true
Does the string start with 'Java'? false

9. Does the string end with 'World'? true
Does the string end with 'Java'? false

10- using equals(): true, false
Using ==: true, false

11. Sorted Words:

apple
ball
cat
dog
ant
free
gun
hen
ice
jug
kite
lift
man
net
orange
parrot
queen
ring
star
tree
umbrella
van
watch
xmas
yatch
zee

12. Sorted Number (10 to 1):
10 9 8 7 6 5 4 3 2 1

13. Modified String: ~~It is a beautiful day. It is sunny and it is warm.~~
This is a test. This is, too.

14. Concatenated string : helloworld

15. Original string : Welcome to college.
Modified string : Welcome to commerce

16. Original string : 'Hello friend'
Trimmed string : 'Hello Friends'

17. Enter details for student 1:
Registration Number : 101
Enter name: Piyush
Semester : 3
CGPA : 8.9

Enter details for student 2:
Registration Number : 102
Name : Ram
Semester : 3
CGPA : 7.8

Enter details for student 3:
Registration Number: 103
Enter name: Jay
Semester : 3
CGPA : 9.3

Date
Page

18. After setLength (5) : Hello
charAt (1) : After setchar (1, 'a')
Halloget chars (0, 5, charArray, 0 : Hallo after append :
Hello World : After Insert (6, "Java");
Hello, Java World !
After reverse : ! dlroW avaJ , olleH
After delete : (3, 11) : ! dlro , olleH
After deleteCharAt() : ! lro , olleH
After replace (7, 12, " World") :
Hello, World Substring (7, 12) : World

19. Eagle : Eagle flies high in the skay, Eagle screaches loudly.

Hawk :
Hawk soars gracefully through the air,
Hawk emits a piercing waff.

20. Circle :
Area = 78 . 539816339 7448
Perimeter . 31. 4159265358 9783

Triangle :
Area : 6.0
Perimeter : 12.0

16/01/2024

LAB-7 Programs 28/01/24

Student.Java

```
package cie;
public class students {
        public string name, USN;
        public int sem;
}
```

Internal.java

```
package cie;
import.java.util.scanner;
public class internal extends student {
        public int marks[] = new int [5];
        public void Input marks () {
        {
            scanner sc = new Scanner (System.in);
            for (int i=0; i<5 ; i++) {
                System.out.println("Enter
                  subject" + (i+1) + "marks");
                marks[i] = sc.nextInt();
            }
        }
        public void display marks() {
            for (int i=0; i<5 ; i++) {
                System.out.println("Subject" + (i+1) + "marks"+
                    marks[i]);
            }
        }
}
```

Externals
~~Extends~~.java

```
package see;
import cie.Student;
import java.util.scanner;

public class Externals Extends Students {
        public int marks ()= new int [5];
        public void input marks(){
                Scanner sc = new Scanner (System.in);
                for(int i = 0; i < 5 ; i++ ){
                  System.out.println ("Enter Subject " + (i+1)+
                                "marks");
                marks [i] = sc.next Int();
                }
    }

    public void display marks () {
            for(int i=0; i<5; i++) {
              System .out. println ("Subject " + (i+1) + "marks"
                          + marks [i]);
            }
    }
}
```

Main.java

```
import cie.Student;
import cie.internals;
import cie.externals;
import java.util.Scanner;

class Main {
        public static void main (String args []){
            int no=2;
            Externals final marks [] = new. Externals [100];
            Internals int marks [] = new Internals [100];
            for (int i=0; i<100; i++) {
                final marks [i]= new Externals();
                int marks [i] = new Internals();
                final marks [i]* input marks[];
                int marks [i]. input marks [];
            }
            for (int i=0; i<100; i++){
                System.out. println ("CIE:" );
                int marks [i]. display marks();
                System. Out. println ("SEE : ");
                final marks[i]. display marks();
            }
        }
}
```

Date
Page

.) Output:-

Enter subject 1 Marks: 30
Enter subject 2 Marks: 50
Enter subject 3 Marks: 40
Enter subject 4 Marks: 20
Enter subject 5 Marks: 10

Enter subject 1 Marks: 30
Enter subject 2 Marks: 70
Enter subject 3 Marks: 60
Enter subject 4 Marks: 80
Enter subject 5 Marks: 90

CIE

Subject 1 Marks : 30
Subject 2 Marks : 50
Subject 3 Marks : 40
Subject 4 Marks : 20
Subject 5 Marks : 10

SEE

Subject 1 Marks : 30
Subject 2 Marks : 70
Subject 3 Marks : 60
Subject 4 Marks : 80
Subject 5 Marks : 90

LAB -8 Programs 30/01/24

```
import java.util.Scanner;

class WrongAge extends RuntimeException {
        public WrongAge() {
                super ("Age cannot be negative");
        }

        public WrongAge (String message) {
                super (message);
        }
}

class InputScanner {
        protected Scanner scanner;

        public InputScanner () {
                scanner = new Scanner (System.in);
        }

        public int nextInt () {
                return scanner.nextInt();
        }
}

class Father extends InputScanner {
        protected int FatherAge;

        public Father () {
                System.out.println ("Enter father's age : ");
                fatherAge = super.nextInt();
```

```
        if (fatherAge < 0) {
            throw new WrongAge ("Age cannot be nega
        }
    }

class Son extends Father {
    private int sonAge;

    public Son () {
        super ();
        System.out.println ("Enter Son's age : ");
        sonAge = super.nextInt ();

        if (sonAge > fatherAge) {
            throw new WrongAge ("Son's age cannot be
                greater than father's age");
        }
        else if (sonAge < 0) {
            throw new WrongAge ("Age cannot be negat
        }
}

public void display () {
    super.display ();
    System.out.println ("Son's Age : " + sonAge);
    }
}
```

```
public class InheritanceException {
    public static void main (String [] args) {
        try {
            Son son = new Son();
            son.display();
        }
        catch (WrongAge e) {
            System.out.println ("Exception: " + e.getMessage())
        }
    }
}
```

→ Output:

Enter Fathers Age : 50
Enter Son's Age : 45

Father's Age : 50
Son's Age : 45

Enter Father's Age : -45

Exception : Age cannot be negative

Enter Fathers Age : 50
Enter Sons Age : 55

Exception : Son's Age cannot be greater than Fathers Age.

06/02/24 Week - 8

Creating two threads

```
class DisplayThread extends Threads {
        private String message;
        private int interval;
        private boolean running = true;
        public DisplayThread (String message, int interval)
            this. message = message;
            this. interval = interval;
        }
        public void run () {
            while (running) {
                System.out. println(message) ;
                try {
                    Thread. sleep (interval);
                }
                catch (InterruptedException) {
                    e. printstackTrace ();
                }
            }
        }
        public void stopThread () {
            running. false;
        }
}
public class Thread 2 {
        public static void main (String [] args) {

        DisplayThread bmsThread = new Display
                Thread ("BMS College of Engineering",
                1000);
```

```
        DisplayThread cseThread = new DisplayThread("CSE", 2000);
        bmsThread.start();
        cseThread.start();
        System.out.println("Press Enter to stop thread..");
        try {
            System.in.read();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
        bms.Thread.stopThread();
        cseThread.stopThread();
    }
}
```

Outputs:-

```
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
```

20/02/24 Lab - 9: Create a user interface to perform integer div.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event;

class SwingDemo {
    SwingDemo () {
        JFrame jfrm = new JFrame ("Divider App");
        jfrm.setSize (275, 150);
        jfrm.setLayout (new FlowLayout ());
        jfrm.setDefaultCloseOperation (JFrame.Exit_On_Close

        JLabel jLab = new JLabel("Enter the divider &
                                    dividend");

        JTextFeild ajtf = new JTextfield (8);
        JTextField bjtf = new JTextfield (8);

        JButton button = new JButton ("Calculate");

        JLabel err = new JLabel();
        JLabel alab = new JLabel();
        JLabel blab = new JLabel();
        JLabel anslab = new JLabel();

        jfrm.add(err);
        jfrm.add(jlab);
        jfrm.add(ajtf);
        jfrm.add(bjtf);
        jfrm.add(button);
        jfrm.add(alab);
        jfrm.add(blab);
        jfrm.add(anslab);
```

```
Action listner l = new Action listner(){
    public void actionPerformed (Action Event evt){
        System.out.println("Action event from a text
                                field");
    }
};
aftf.addActionListner(l);
bjtf.addAction Listner(l);

button.addActionListenr (new Action listenr(){
    public void actionPerformed(ActionEvent evt){
        try{
            int a = Integer.parseInt (ajtf.getText());
            int b = Integer.parseInt (bjtf.getText());
            int ans = a/b;

            alab.setText("\n A=" +a);
            blab.setText("\n B=" +b);
            anslab.setText("\n Ans = " +ans);
        }
        catch( NumberFormatException e){
            alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("Enter only Integers");
        }
        catch (ArithmeticException e){
            alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("B should be NON ZERO!");
        }
    }
});
```

```
        jframe.setVisible(True);
    }

public static void main (String args[]){
    SwingUtilities.invokeLater(new Runnable(){
        public void run(){
            new SwingDemo();
        }
    }};
    }
}
```

Output:-

Enter the dividend & divider:

[200] [2]

[Calculate] A=200 B=2 Ans = 100

20.08.24

06/02/24 Week - 10 IPC

```
class Q {
  int n;
  boolean valueSet = false;
  Synchronized int get() {
  while (!valueSet)
  try {
        wait();
  } catch (InterruptedException e) {
      System.out.println("InterruptedException caught");}
      System.out.println("Got: " + n);
      valueSet = false;
      notify();
      return n;
      }

Synchronized void put (int n) {
  while (valueSet)
  try {
      wait();
  } catch (InterruptedException e) {
      System.out.println("Put " + n);
      notify();
      } }

class Producer implements Runnable {
    Q q;
    Producer (Q q) {
        this.q = q;
        new Thread (this, "Producer ").start();
    }
```

Date
Page

```
public void run (){
        int i=0;
        while (i <15) {
                q.put (i++);
        }
    }
}
class consumer implements Runnable {
a q;
consumer (a q) {
    this.q = q;
    new Thread (this, "consumer" ). start ();
}
public void run(){
  int i=0;
  while (i < 10){
    int r= q.get();
    i++;
  }
}
}
class p(fixed) {
    public static void main (String args [] ) {
     a q = new a();
     new producer (q);
     new consumer (q);
     System. out. println(" Press control - to stop "); }
}
```

Output: Put : 0
Got : 0
Put : 1
Got : 1
Put : 2

Got: 2	Got: 4	Got: 6	Got: 8
Put: 3	Put: 5	Put: 7	Put: 9
Got: 3	Got: 5	Got: 7	Got: 9
Put: 4	Put: 6	Put: 8	Put: 10
			Got: 10

Deadlock:

```
class A {
Synchronized void foo (B b){
String name = Thread.currentThread().getName();
System.out.println(name + "entered A.foo ");
try {
    Thread.sleep(1000); }
catch (Exception e){
    System.out.println("A Interrupted "); }
    System.out.println(name + " trying to call B.last()");
    b.last(); }
    void last(){
        System.out.println("Inside A.last ");
        }
    }

    class B {
    synchronized void bar(A a) {
        String name Thread.currentThread().getName();
        System.out.println(name + "entered B.bar ");
        try {
            Thread.sleep(1000); }
        catch (Exception e){
            System.out.println("B Interrupted ");
        }
```

```
System.out.println (name + "trying to call A.last()");
a.last(); }

void last(){
    System.out.println("Inside A.last");
  }

}
class Deadlock implements Runnable
  {
    A a = new A();
    B b = new B();
    Deadlock(){
      Thread.currentThread().setName("Main thread");
      Thread t = newThread(this, "Racing thread");
      t.start();
      a.foo(b);
      System.out.println("Back in main thread");
    }
```

Output:-

```
Main thread entered A.foo
Racing Thread entered B.bar
Main thread trying to call B.last()
Inside A.last.
Back in main thread
Racing thread trying to call A.last()
Inside A.last
Back in other thread
```

Got: 2	Got: 4	Got: 6	Got: 8
Put: 3	Put: 5	Put: 7	Put: 9
Got: 3	Got: 5	Got: 7	Got: 9
Put: 4	Put: 6	Put: 8	Put: 10
			Got: 10

Deadlock:

```
class A {
Synchronized void foo (B b){
String name = Thread.currentThread().getName();
System.out.println(name + "entered A.foo ");
try {
    Thread.sleep(1000); }
catch (Exception e){
    System.out.println("A Interrupted "); }
    System.out.println(name + " trying to call B.last()");
    b.last(); }
    void last(){
        System.out.println("Inside A.last ");
        }
    }

    class B {
    synchronized void bar(A a) {
        String name Thread.currentThread().getName();
        System.out.println(name + "entered B.bar ");
        try {
            Thread.sleep(1000); }
        catch (Exception e){
            System.out.println("B Interrupted ");
        }
```

```
System.out.println (name + "trying to call A.last()");
a.last(); }

void last(){
    System.out.println("Inside A.last");
  }

}
class Deadlock implements Runnable
  {
    A a = new A();
    B b = new B();
    Deadlock(){
      Thread.currentThread().setName("Main thread");
      Thread t = newThread(this, "Racing thread");
      t.start();
      a.foo(b);
      System.out.println("Back in main thread");
    }
```

Output:-

```
Main thread entered A.foo
Racing Thread entered B.bar
Main thread trying to call B.last()
Inside A.last.
Back in main thread
Racing thread trying to call A.last()
Inside A.last
Back in other thread
```

2nd lab:

```
import java.util.Scanner;

class subject{
int subjectMarks, credits, grade;}
class Student {
    String name;
    String usn;
    double SGPA;
    Scanner s;
    subject subjects[];

Student()
{
int i;
subjects = new subject[9];
for(i=0;i<8;i++)
subjects[i] = new subject();
s = new Scanner(System.in);
}

public void getStudentDetails(){
System.out.println("Enter student name:");
name=s.nextLine();

System.out.println("Enter Student USN:");
usn=s.nextLine();}

public void getMarks(){
int i;
for(i=0;i<8;i++){
System.out.println("Enter marks of subject"+(i+1)+":");
subjects[i].subjectMarks= s.nextInt();
if(subjects[i].subjectMarks>=40&&subjects[i].subjectMarks<=100){
subjects[i].grade=calculateGrade(subjects[i].subjectMarks);}
```

```
else{
System.out.println("Invalid Marks. Marks should be between 40 and 100");}
System.out.println("enter credits:");
subjects[i].credits=s.nextInt();
}
}

public int calculateGrade(int marks){
if (marks>=90)
return 10;
else if(marks>=70&&marks<=80)
return 9;
else if(marks>=60&&marks<=70)
return 8;
else if(marks>=50&&marks<=60)
return 7;
else
return 6;
}

public void computeSGPA() {
        int totalscore = 0;
        int totalcred = 0;

        for (int i = 0; i < 8; i++) {
            totalscore += subjects[i].grade * subjects[i].credits;
            totalcred += subjects[i].credits;
        }
        SGPA = (double) totalscore / (double) totalcred;
    }
}

public class Stud{
public static void main(String args[]){
Student s1=new Student();
```

```
public void computeSGPA() {
        int totalscore = 0;
        int totalcred = 0;

        for (int i = 0; i < 8; i++) {
            totalscore += subjects[i].grade * subjects[i].credits;
            totalcred += subjects[i].credits;
        }
        SGPA = (double) totalscore / (double) totalcred;
    }
}

public class Stud{
public static void main(String args[]){
Student s1=new Student();

s1.getStudentDetails();

s1.getMarks();
s1.computeSGPA();

System.out.println("Student name:"+s1.name);
System.out.println("Student usn:"+s1.usn);
System.out.println("Student sgpa:"+s1.SGPA);}
}
```

3rd code:

```
import java.util.Scanner;

class Book {
   String name;
   String author;
   int price;
   int numPages;


   Book(String name, String author, int price, int numPages) {
      this.name = name;
      this.author = author;
      this.price = price;
      this.numPages = numPages;
   }


   public String toString() {
      String bookDetails = "Book name: " + this.name + "\n"
            + "Author name: " + this.author + "\n"
```

```
            + "Price: " + this.price + "\n"
            + "Number of pages: " + this.numPages + "\n";
        return bookDetails;
    }
}

public class Main {
    public static void main(String args[]) {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter the number of books: ");
        int n = s.nextInt();


        Book[] books = new Book[n];


        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for Book " + (i + 1));
            System.out.println("Name: ");
            String name = s.next();
            System.out.println("Author: ");
            String author = s.next();
            System.out.println("Price: ");
            int price = s.nextInt();
            System.out.println("Number of pages: ");
            int numPages = s.nextInt();

            // Create a new Book object with the entered details
            books[i] = new Book(name, author, price, numPages);
        }


        System.out.println("\nDetails of the books:");
        for (int i = 0; i < n; i++) {
            System.out.println("Book " + (i + 1) + ":\n" + books[i].toString());
        }
    }
}
```

4th code:

```
import java.util.Scanner;

class InputScanner

{

    Scanner s = new Scanner(System.in);
```

```
    double getInput (String prompt){
        System.out.println(prompt);
        return s.nextDouble();
    }
}
abstract class Shape extends InputScanner{
    double side1.side2;
    abstract void area();
}

class Rectangle extends Shape{
    Rectangle()
    {
        side_1=getInput("Enter length of rectangle");
        side_2=getInput("Enter breadth of rectangle");
    }
    void area()
    {
        double area = side_1*side_2;
        System.out.println("Area of rectangle = "+area);
    }
}
class Triangle extends Shape{
    Triangle()
    {
        side_1=getInput("Enter Base of Triangle");
        side_2=getInput("Enter Height of Triangle");
    }
    void area()
```

```
    {
        double area = side_1*side_2/2;
        System.out.println("Area of Triangle = "+area);
    }
}
class Circle extends Shape{
    Circle()
    {
        side_1=getInput("Enter the radius of the circle: ");
    }
    void area()
    {
        double area=Math.pi*side_1*side_1;
        System.out.println("Area of the circe = "+area);
    }
}
class main1{
    public static void main (String args[]){
        Rectangle rectangle = new Rectangle();
        Triangle triangle = new Triangle();
        Circle circle = new Circle();

        rectangle.area();
        triangle.area();
        circle.area();
    }
}
```

5th code:

```
import java.util.Scanner;

class Account {
    String customerName;
    long accountNumber;
    String accountType;
    double balance;

    public Account(String customerName, long 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. Updated balance: $" + balance);
    }

    public void displayBalance() {
        System.out.println("Account Balance: $" + balance);
    }
}

class CurAccount extends Account {
    double minBalance;
    double serviceCharge;
```

```
    public CurAccount(String customerName, long accountNumber, double balance, double minBalance,
double serviceCharge) {
        super(customerName, accountNumber, "Current", balance);
        this.minBalance = minBalance;
        this.serviceCharge = serviceCharge;
    }

    public void checkMinBalance() {
        if (balance < minBalance) {
            balance -= serviceCharge;
            System.out.println("Minimum balance not maintained. Service charge of $" + serviceCharge + "
imposed.");
            displayBalance();
        }
    }

    public void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient funds. Withdrawal failed.");
        } else {
            balance -= amount;
            System.out.println("Withdrawal of $" + amount + " successful. Updated balance: $" + balance);
            checkMinBalance();
        }
    }
}

class SavAccount extends Account {
    double interestRate;
```

```
    public SavAccount(String customerName, long accountNumber, double balance, double interestRate) {
        super(customerName, accountNumber, "Savings", balance);
        this.interestRate = interestRate;
    }

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

    public void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient funds. Withdrawal failed.");
        } else {
            balance -= amount;
            System.out.println("Withdrawal of $" + amount + " successful. Updated balance: $" + balance);
        }
    }
}

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

        CurAccount currentAccount = new CurAccount("John Doe", 123456789, 1000, 500, 10);
        SavAccount savingsAccount = new SavAccount("Jane Doe", 987654321, 2000, 5);
```

```
int choice;
do {
    System.out.println("\nSelect an option:");
    System.out.println("1. Deposit");
    System.out.println("2. Display Balance");
    System.out.println("3. Compute Interest (Savings Account only)");
    System.out.println("4. Withdraw");
    System.out.println("5. Exit");
    System.out.print("Enter your choice: ");
    choice = scanner.nextInt();

    switch (choice) {
        case 1:
            System.out.print("Enter amount to deposit: ");
            double depositAmount = scanner.nextDouble();
            System.out.print("Select account (1. Current, 2. Savings): ");
            int accountType = scanner.nextInt();
            if (accountType == 1) {
                currentAccount.deposit(depositAmount);
            } else if (accountType == 2) {
                savingsAccount.deposit(depositAmount);
            } else {
                System.out.println("Invalid account type.");
            }
            break;
        case 2:
            System.out.print("Select account (1. Current, 2. Savings): ");
            int accType = scanner.nextInt();
```

```
    if (accType == 1) {
        currentAccount.displayBalance();
    } else if (accType == 2) {
        savingsAccount.displayBalance();
    } else {
        System.out.println("Invalid account type.");
    }
    break;
case 3:
    if (savingsAccount instanceof SavAccount) {
        ((SavAccount) savingsAccount).computeInterest();
    } else {
        System.out.println("Invalid option for current account.");
    }
    break;
case 4:
    System.out.print("Enter amount to withdraw: ");
    double withdrawAmount = scanner.nextDouble();
    System.out.print("Select account (1. Current, 2. Savings): ");
    int accTyp = scanner.nextInt();
    if (accTyp == 1) {
        currentAccount.withdraw(withdrawAmount);
    } else if (accTyp == 2) {
        savingsAccount.withdraw(withdrawAmount);
    } else {
        System.out.println("Invalid account type.");
    }
    break;
case 5:
```

```
            System.out.println("Exiting the program. Thank you!");
            break;
          default:
            System.out.println("Invalid choice. Please enter a valid option.");
        }
    } while (choice != 5);

    scanner.close();
  }
}
```

6th code:

1.

```
public class StringConstructorDemo {
        public static void main(String[] args) {
        String str1 = "Hello, World!"
        System.out.println("String created using a string literal: " + str1);
        }
}
public class StringConstructorDemo {
        public static void main(String[] args) {
                char[] charArray = {'H', 'e', 'l', 'l', 'o'};
                String str2 = new String(charArray);
                System.out.println("String created using the new keyword and char array: " + str2);
        }
}
public class StringConstructorDemo {
        public static void main(String[] args) {
                byte[] byteArray = {72, 101, 108, 108, 111};
```

```
                String str3 = new String(byteArray);
                System.out.println("String created using getBytes method: " + str3);
                }
}
public class StringConstructorDemo {
        public static void main(String[] args) {
                StringBuilder stringBuilder = new StringBuilder("Java");
                String str4 = new String(stringBuilder);
                System.out.println("String created using StringBuilder: " + str4);
        }
}
```

2.

```
public class StringDemo {
        public static void main(String[] args) {
        String exampleString = "Hello, World!"
        int length = exampleString.length();
        System.out.println("String Length: " + length);
        String stringLiteral1 = "Java"
        String stringLiteral2 = "Java" // Reusing the string literal
        System.out.println("String Literal 1: " + stringLiteral1);
        System.out.println("String Literal 2: " + stringLiteral2);
        System.out.println("Are String Literals Equal? " + (stringLiteral1 == stringLiteral2));
        String firstName = "John"
        String lastName = "Doe"
        String fullName = firstName + " " + lastName;
        System.out.println("Concatenated String: " + fullName);
        }
}
```

3.

```
class Person {
	private String name;
	private int age;
	public Person(String name, int age) {
	this.name = name;
	this.age = age;
}
@Override
public String toString() {
		return "Person{name='" + name + "', age=" + age + '}';
	}
}
public class ToStringDemo {
	public static void main(String[] args) {
		Person person = new Person("John Doe", 25);
		System.out.println(person); // Output: Person{name='John Doe', age=25}
	}
}
```

4.

```
public class SubstringExtraction {
	public static void main(String[] args) {
		String originalString = "Welcome to Bmsce college"
		char[] extractedChars = new char[5];
		originalString.getChars(11, 16, extractedChars, 0);
		String extractedString = new String(extractedChars);
		System.out.println("Extracted Substring: " + extractedString);
	}
}
```

5.

```
public class GetBytesDemo {
	public static void main(String[] args) {
		String originalString = "Hello, World!"
		byte[] byteArray = originalString.getBytes();
		System.out.println("Byte Array: " + byteArray);
		System.out.print("Bytes: ");
		for (byte b : byteArray) {
			System.out.print(b + " ");
		}
	}
}
public class ToCharArrayDemo {
	public static void main(String[] args) {
		String originalString = "Java Programming"
		char[] charArray = originalString.toCharArray();
		System.out.println("Char Array: " + charArray);
		System.out.print("Chars: ");
		for (char c : charArray) {
			System.out.print(c + " ");
		}
	}
}
```

6.

```
public class StringComparison {
	public static void main(String[] args) {String str1 = "Bmsce"
	String str2 = "College"
	String str3 = "BMSCE"
	System.out.println("Using equals(): Bmsce equals Bmsce -> " + str1.equals("Bmsce"));
```

```
        System.out.println("Using equals(): Bmsce equals College -> " + str1.equals(str2));
        System.out.println("Using equals(): Bmsce equals BMSCE -> " + str1.equals(str3));
        System.out.println("Using equalsIgnoreCase(): Bmsce equalsIgnoreCase BMSCE -> " +
        str1.equalsIgnoreCase(str3));
}
```

7.

```
public class RegionMatchesExample {
        public static void main(String[] args) {
                String mainString = "Welcome to Bmsce College of Engineering"
                String subString = "Bmsce College"
                boolean isMatched = mainString.regionMatches(11, subString, 0, subString.length());
                if (isMatched) {
                        System.out.println("Substring is matched.");
                } else {
                        System.out.println("Substring is not matched.");
                }
        }
}
```

8.

```
public class StartsWithExample {
        public static void main(String[] args) {
                String mainString = "Hello, World!"
                boolean startsWithHello = mainString.startsWith("Hello");
                boolean startsWithJava = mainString.startsWith("Java");
                System.out.println("Starts with 'Hello': " + startsWithHello); // Should be true
                System.out.println("Starts with 'Java': " + startsWithJava); // Should be false
        }
}
```

9.

```
public class EndsWithExample {
	public static void main(String[] args) {
		String mainString = "Hello, World!"
		boolean endsWithWorld = mainString.endsWith("World!");
		boolean endsWithJava = mainString.endsWith("Java");
		System.out.println("Ends with 'World!': " + endsWithWorld); // Should be true
		System.out.println("Ends with 'Java': " + endsWithJava); // Should be false
	}
}
```

10.

```
public class EqualsVsDoubleEquals {
	public static void main(String[] args) {
		String str1 = "Hello"
		String str2 = "Hello"
		String str3 = new String("Hello");
		boolean equalsResult1 = str1.equals(str2); // true
		boolean equalsResult2 = str1.equals(str3); // true
		boolean doubleEqualsResult1 = (str1 == str2); // true (due to string pooling)
		boolean doubleEqualsResult2 = (str1 == str3); // false (different objects)
		System.out.println("Using equals(): " + equalsResult1 + ", " + equalsResult2);
		System.out.println("Using ==: " + doubleEqualsResult1 + ", " + doubleEqualsResult2);
	}
}
```

11.

```
import java.util.Arrays;
public class AlphabeticalSorting {
```

```
        public static void main(String[] args) {
                String[] words = {"van", "watch", "ball", "cat", "xmas", "yatch", "zee", "apple", "ice",
                "jug", "kite", "lift", "man", "net", "orange", "dog", "ent", "free", "gun", "hen", "parrot",
                "queen", "ring", "star", "tree", "umbrella"};
                 System.out.println("Original Array: " + Arrays.toString(words));
                System.out.println("Sorted Array: " + Arrays.toString(words));
        }
}
```

12.

```
import java.util.Arrays;
public class NumberSorting {
        public static void main(String[] args) {
                String[] numbers = {"10", "9", "8", "7", "6", "5", "4", "3", "2", "1"};
                Arrays.sort(numbers);
                System.out.println("Sorted Numbers:");
                for (String number : numbers) {
                        System.out.println(number);
                }
        }
}
```

13.

```
public class StringReplaceExample {
        public static void main(String[] args) {
                String originalString = "Thwas was a test. Thwas was, too."
                String targetSubstring = "was"
                String replacementString = "is"
                int index = originalString.indexOf(targetSubstring);
```

```
                StringBuilder result = new StringBuilder();
                while (index != -1) {
                result.append(originalString.substring(0, index));
                result.append(replacementString);
                index += targetSubstring.length();
                originalString = originalString.substring(index);
                index = originalString.indexOf(targetSubstring);
        }
                result.append(originalString);
                System.out.println("Original String: " + originalString);
                System.out.println("After Replacement: " + result.toString());
        }
}
```

14.

```
public class StringConcatenationDemo {
        public static void main(String[] args) {
                String s1 = "hello";
                String s2 = "world"; //
                String result = s1.concat(s2);
                 System.out.println("Concatenated String: " + result); } }
```

15.

```
public class StringReplaceDemo {
   public static void main(String[] args) {
      String originalString = "Welcome to College";
      String modifiedString = originalString.replace("College", "Commege");
      System.out.println("Original String: " + originalString);
      System.out.println("Modified String: " + modifiedString);
```

```
    }
```

16.

```
public class StringTrimDemo {
    public static void main(String[] args) {
        String originalString = " Hello Friends ";

        String trimmedString = originalString.trim();

        System.out.println("Original String: '" + originalString + "'");
        System.out.println("Trimmed String: '" + trimmedString + "'");
    }
}
```

17.

```
import java.util.Arrays;
import java.util.Comparator;
import java.util.Scanner;

class Student {
    private int registrationNumber;
    private String fullName;
    private short semester;
    private float cgpa;

    public Student() {
}
    public Student(int registrationNumber, String fullName, short semester, float cgpa) {
        this.registrationNumber = registrationNumber;
        this.fullName = fullName;
```

```
    this.semester = semester;
    this.cgpa = cgpa;
}   public void display() {
    System.out.println("Registration Number: " + registrationNumber);
    System.out.println("Full Name: " + fullName);
    System.out.println("Semester: " + semester);
    System.out.println("CGPA: " + cgpa);
    System.out.println("------------------------");
}
public int getRegistrationNumber() {
    return registrationNumber;
}

public String getFullName() {
    return fullName;
}

public short getSemester() {
    return semester;
}

public float getCgpa() {
    return cgpa;
}
public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    Student[] students = new Student[5];
    for (int i = 0; i < 5; i++) {
        System.out.println("Enter details for student " + (i + 1) + ":");
```

```
        System.out.print("Registration Number: ");
        int regNumber = scanner.nextInt();
        scanner.nextLine(); // Consume newline
        System.out.print("Full Name: ");
        String name = scanner.nextLine();
        System.out.print("Semester: ");
        short semester = scanner.nextShort();
        System.out.print("CGPA: ");
        float cgpa = scanner.nextFloat();

        students[i] = new Student(regNumber, name, semester, cgpa);
    }
    System.out.println("\nStudent Records:");
    for (Student student : students) {
        student.display();
    }
    sortByCGPA(students);
    System.out.println("\nStudent Records (Sorted by CGPA):");
    for (Student student : students) {
        student.display();
    }
    sortByName(students);
    System.out.println("\nStudent Records (Sorted by Name):");
    for (Student student : students) {
        student.display();
    }
}
private static void sortByCGPA(Student[] students) {
    Arrays.sort(students, Comparator.comparingDouble(Student::getCgpa).reversed());
```

```
    }
    private static void sortByName(Student[] students) {
        Arrays.sort(students, Comparator.comparing(Student::getFullName));
    }
}
```

18.

```
public class StringBufferDemo {
    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Hello, World!");
        stringBuffer.setLength(5);
        System.out.println("After setLength(5): " + stringBuffer);
        char charAtIndex = stringBuffer.charAt(1);
        System.out.println("charAt(1): " + charAtIndex);
        stringBuffer.setCharAt(1, 'a');
        System.out.println("After setCharAt(1, 'a'): " + stringBuffer);
        char[] charArray = new char[5];
        stringBuffer.getChars(0, 5, charArray, 0);
        System.out.println("getChars(0, 5, charArray, 0): " + new String(charArray));
    }
}
public class StringBufferDemo2 {
    public static void main(String[] args) {
        StringBuffer stringBuffer = new StringBuffer("Hello");
        stringBuffer.append(" World!");
        System.out.println("After append: " + stringBuffer);
        stringBuffer.insert(6, ", Java");
        System.out.println("After insert(6, \", Java\"): " + stringBuffer);
        stringBuffer.reverse();
        System.out.println("After reverse: " + stringBuffer);
```

```
        stringBuffer.delete(5, 11);
        System.out.println("After delete(5, 11): " + stringBuffer);
        stringBuffer.deleteCharAt(1);
        System.out.println("After deleteCharAt(1): " + stringBuffer);
    }
}public class StringBufferDemo3 { public static void main(String[] args) { StringBuffer stringBuffer = new
StringBuffer("Hello, Java!"); // replace() method stringBuffer.replace(7, 12, "World");
System.out.println("After replace(7, 12, \"World\"): "
```

19.

```
abstract class Bird {
    public abstract void fly();
    public abstract void makeSound();
}
class Eagle extends Bird {
    @Override
    public void fly() {
        System.out.println("Eagle flies high in the sky.");
    }
    @Override
    public void makeSound() {
        System.out.println("Eagle screeches loudly.");
    }
}
class Hawk extends Bird {
    @Override
    public void fly() {
        System.out.println("Hawk soars gracefully through the air.");
    }
```

```
    @Override
    public void makeSound() {
        System.out.println("Hawk emits a piercing cry.");
    }
}
public class BirdTest {
    public static void main(String[] args) {
        // Create instances of Eagle and Hawk
        Eagle eagle = new Eagle();
        Hawk hawk = new Hawk();
        System.out.println("Eagle:");
        eagle.fly();
        eagle.makeSound();
        System.out.println();
        System.out.println("Hawk:");
        hawk.fly();
        hawk.makeSound();
    }
}
```

20.

```
abstract class Shape {
    public abstract double calculateArea();
    public abstract double calculatePerimeter();
}

class Circle extends Shape {
    private double radius;
    public Circle(double radius) {
        this.radius = radius;
```

```
    }

    @Override
    public double calculateArea() {
        return Math.PI * radius * radius;
    }

    @Override
    public double calculatePerimeter() {
        return 2 * Math.PI * radius;
    }
}
class Triangle extends Shape {
    private double side1, side2, side3;
    public Triangle(double side1, double side2, double side3) {
        this.side1 = side1;
        this.side2 = side2;
        this.side3 = side3;
    }

    @Override
    public double calculateArea() {
 triangle
        double s = (side1 + side2 + side3) / 2.0;
        return Math.sqrt(s * (s - side1) * (s - side2) * (s - side3));
    }
    @Override
    public double calculatePerimeter() {
        return side1 + side2 + side3;
```

```
    }
}
public class ShapeTest {
    public static void main(String[] args) {
        // Create instances of Circle and Triangle
        Circle circle = new Circle(5.0);
        Triangle triangle = new Triangle(3.0, 4.0, 5.0);
 for Circle
        System.out.println("Circle:");
        System.out.println("Area: " + circle.calculateArea());
        System.out.println("Perimeter: " + circle.calculatePerimeter());
        System.out.println();
 for Triangle
        System.out.println("Triangle:");
        System.out.println("Area: " + triangle.calculateArea());
        System.out.println("Perimeter: " + triangle.calculatePerimeter());
    }
}
```

7th code:

```
package CIE;


public class Internals {
    private int[] internalMarks = new int[5];



    public Internals() {


    }
```

```
    public void setInternalMarks(int[] internalMarks) {
        this.internalMarks = internalMarks;
    }



    public int[] getInternalMarks() {
        return internalMarks;
    }
}

package CIE;

public class Student {
    public String usn;
    public String name;
    public int sem;



    public Student() {
        this("", "", 0);
    }



    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
```

```
    }

    public void setUsn(String usn) {
        this.usn = usn;
    }

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

    public void setSem(int sem) {
        this.sem = sem;
    }

    public String getUsn() {
        return usn;
    }

    public String getName() {
        return name;
    }

    public int getSem() {
        return sem;
    }
}
```

```
package SEE;
import CIE.Student;

public class External extends Student {
   public int[] seeMarks = new int[5];
public External() {
      this("", "", 0, new int[5]);
   }
  public External(String usn, String name, int sem, int[] seeMarks) {
      super(usn, name, sem);
      this.seeMarks = seeMarks;
   }
public void setSeeMarks(int[] seeMarks) {
      this.seeMarks = seeMarks;
   }
 public int[] getSeeMarks() {
      return seeMarks;
   }
}
import CIE.Student;
import CIE.Internals;
import SEE.External;
import java.util.Scanner;

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

      // Allow the user to enter the number of students
```

```
System.out.print("Enter the number of students: ");
int n = scanner.nextInt();

Student[] students = new Student[n];
Internals[] internals = new Internals[n];
External[] externals = new External[n];

// Initialize students, internals, and externals
for (int i = 0; i < n; i++) {
    students[i] = new Student();
    System.out.print("Enter USN for student " + (i + 1) + ": ");
    students[i].setUsn(scanner.next());

    System.out.print("Enter name for student " + (i + 1) + ": ");
    students[i].setName(scanner.next());

    System.out.print("Enter semester for student " + (i + 1) + ": ");
    students[i].setSem(scanner.nextInt());

    internals[i] = new Internals();
    // Assuming a simple method to input internal marks with validation
    internals[i].setInternalMarks(inputMarksWithValidation("internal", i, scanner, 0, 50));

    externals[i] = new External(students[i].getUsn(), students[i].getName(), students[i].getSem(), new
int[5]);
    // Assuming a simple method to input external marks with validation
    externals[i].setSeeMarks(inputMarksWithValidation("external", i, scanner, 0, 100));

    // Calculate final marks for the ith student and display
```

```
            int[] finalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                finalMarks[j] = internals[i].getInternalMarks()[j] + externals[i].getSeeMarks()[j] / 2;
            }

            System.out.println("Student " + (i + 1) + " Final Marks: " +
                    finalMarks[0] + ", " + finalMarks[1] + ", " + finalMarks[2] + ", " +
                    finalMarks[3] + ", " + finalMarks[4]);
        }

        scanner.close();
    }

    private static int[] inputMarksWithValidation(String type, int studentIndex, Scanner scanner, int min,
int max) {
        int[] marks = new int[5];
        System.out.println("Enter " + type + " marks for student " + (studentIndex + 1) + ": ");
        for (int i = 0; i < 5; i++) {
            int mark;
            do {
                System.out.print("Subject " + (i + 1) + ": ");
                mark = scanner.nextInt();
                if (mark < 0 || mark > max) {
                    System.out.println("Invalid input. " + type + " marks should be between 0 and " + max + ".
Please try again.");
                }
            } while (mark < 0 || mark > max);
            marks[i] = mark;
        }
```

```
        return marks;
    }
}
```

8th code:

```
import java.util.Scanner;
class WrongAge extends Exception{
WrongAge(String error){
System.out.println(error);
}
}

class Father{
int age;
Father(int age) throws WrongAge{
if(age<0)
throw new WrongAge("Father's age cannot be negative");
this.age=age;
}
}

class Son extends Father{
int age;
Son(int age,int s_age) throws WrongAge{
super(age);
if(s_age>=age)
throw new WrongAge("Son'age cannot be greater than Father's age");
this.age=s_age;
}
```

```
}

class LabQ7{
public static void main(String args[]){
Scanner sc=new Scanner(System.in);
try{
System.out.println("Enter the Father's age:");
int f_age=sc.nextInt();
System.out.println("Enter the Son's age:");
int s_age=sc.nextInt();
Son a=new Son(f_age,s_age);
System.out.println("Father's age:"+f_age);
System.out.println("Son's age:"+s_age);
}
catch(WrongAge e){
System.out.println("Wrong Age entered");}
catch(Exception ee){
System.out.println("Unexpected error :"+ee);}
}
}
```

8th code (Threads):

```
class DisplayThread extends Thread {
   private String message;
   private int interval;
   private boolean running = true;
   public DisplayThread(String message, int interval) {
      this.message = message;
      this.interval = interval;
```

```
    }
    public void run() {
        while (running) {
            System.out.println(message);
            try {
                Thread.sleep(interval);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
    public void stopThread() {
        running = false;
    }
}
public class ThreadExample {
    public static void main(String[] args) {
        DisplayThread bmsThread = new DisplayThread("BMS College of Engineering", 10000);
        DisplayThread cseThread = new DisplayThread("CSE", 2000);
        bmsThread.start();
        cseThread.start();
        System.out.println("Press Enter to stop the threads...");
        try {
            System.in.read();
        } catch (Exception e) {
            e.printStackTrace();
        }
        bmsThread.stopThread();
        cseThread.stopThread();
```

```
    }
}
```

9th code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

class SwingDemo{
    SwingDemo(){
        // create jframe container
        JFrame jfrm = new JFrame(&quot;Divider App&quot;);
        jfrm.setSize(275, 150);
        jfrm.setLayout(new FlowLayout());
        // to terminate on close
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // text label
        JLabel jlab = new JLabel(&quot;Enter the divider and divident:&quot;);

        // add text field for both numbers
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);

        // calc button
        JButton button = new JButton(&quot;Calculate&quot;);

        // labels
        JLabel err = new JLabel();
        JLabel alab = new JLabel();
```

```
JLabel blab = new JLabel();
JLabel anslab = new JLabel();



// add in order :)
jfrm.add(err);  // to display error bois
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

ActionListener l = new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        System.out.println(&quot;Action event from a text field&quot;);
    }
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);

button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        try{
            int a = Integer.parseInt(ajtf.getText());
            int b = Integer.parseInt(bjtf.getText());
            int ans = a/b;
```

```
            alab.setText(&quot;\nA = &quot; + a);
            blab.setText(&quot;\nB = &quot; + b);
            anslab.setText(&quot;\nAns = &quot;+ ans);
        }
        catch(NumberFormatException e){
            alab.setText(&quot;&quot;);
            blab.setText(&quot;&quot;);
            anslab.setText(&quot;&quot;);
            err.setText(&quot;Enter Only Integers!&quot;);
        }
        catch(ArithmeticException e){
            alab.setText(&quot;&quot;);

            blab.setText(&quot;&quot;);
            anslab.setText(&quot;&quot;);
            err.setText(&quot;B should be NON zero!&quot;);
        }
    }
  });

  // display frame
  jfrm.setVisible(true);
}

public static void main(String args[]){
  // create frame on event dispatching thread
  SwingUtilities.invokeLater(new Runnable(){
    public void run(){
        new SwingDemo();
```

```
            }
        });
    }
}
```

10th code:

```
class Q {
    int n;
    boolean valueSet = false;

    synchronized int get() {
        while (!valueSet)
            try {
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        System.out.println("Got: " + n);
        valueSet = false;
        notify();
        return n;
    }

    synchronized void put(int n) {
        while (valueSet)
            try {
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
```

```
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        notify();
    }
}

class Producer implements Runnable {
    Q q;

    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }

    public void run() {
        int i = 0;
        while (i < 15) {
            q.put(i++);
        }
    }
}

class Consumer implements Runnable {
    Q q;

    Consumer(Q q) {
        this.q = q;
```

```
        new Thread(this, "Consumer").start();
    }

    public void run() {
        int i = 0;
        while (i < 15) {
            int r = q.get();
            i++;
        }
    }
}

class PCFixed {
    public static void main(String args[]) {
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop.");
    }
}
```

10th code (DEADLOCK):

```
class A {
    synchronized void foo(B b) {
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered A.foo");
        try {
            Thread.sleep(1000);
```

```
        } catch (Exception e) {
            System.out.println("A Interrupted");
        }
        System.out.println(name + " trying to call B.last()");
        b.last();
    }

    void last() {
        System.out.println("Inside A.last");
    }
}

class B {
    synchronized void bar(A a) {
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered B.bar");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            System.out.println("B Interrupted");
        }
        System.out.println(name + " trying to call A.last()");
        a.last();
    }

    void last() {
        System.out.println("Inside B.last");
    }
}
```

```
class Deadlock implements Runnable {
   A a = new A();
   B b = new B();

   Deadlock() {
      Thread.currentThread().setName("MainThread");
      Thread t = new Thread(this, "RacingThread");
      t.start();
      a.foo(b); // get lock on a in this thread.
      System.out.println("Back in main thread");
   }

   public void run() {
      b.bar(a); // get lock on b in other thread
      System.out.println("Back in other thread");
   }

   public static void main(String args[]) {
      new Deadlock();
   }
}
```