INDEX

S.NO	Name	Date
1	Quadratic Equation	12/12/2023
2	SGPA Calculation	19/12/2023
3	Book class object Program	26/12/2023
4	Abstract Class - Shape	2/1/2024
5	Bank Account	9/1/2024
6	Strings	16/1/2024
7	Package - CIE, SEE	23/1/2024
8	Exception Handling	30/1/2024
9	Threads	6/2/2024
10	IPC and Deadlock	13/2/2024

Lab 1 - 12/12/2023

UKA12-12-23

Quadratic Equation

```
import java.util.Scanner;
class Quadratic
{   Scanner scan = new Scanner(System.in);
public void compute(int a, int b, int c)
 {
   while(a==0)
   {
    System.out.println("Enter a non zero value of a:");
    a=scan.nextInt();
   }
   int d = b*b - 4*a*c;
   double r1=0.0, r2=0.0;
   if(d==0)
    {
     r1 = -b/(2*a);
     System.out.println("Roots are real and equal.");
     System.out.println("Root 1 = Root 2 = " + r1);
   }
  else if(d>0)
  {
   r1=((-b) + Math.sqrt(d))/(2*a);
   r2=((-b) - Math.sqrt(d))/(2*a);
   System.out.println("Roots are real and distinct.");
   System.out.println("Root 1 = " + r1 + "Root 2="+r2);
  }
else if(d<0)
{
 System.out.println("Roots are imaginary.");
```

```
r1 = -b/(2*a);
r2 = Math.sqrt(-d)/(2*a);
System.out.println("Root 1 = "+r1+" i "+r2);
System.out.println("Root 2 = "+r1+" - i "+r2);
}
}
public static void main(String args[])
{
  int a, b, c;
  System.out.println("Enter the co-efficients a,
                      b and c of a quadratic equati
  a = scan.nextInt();
  b = scan.nextInt();
  c = scan.nextInt();
  Quadratic q = new Quadratic();
  q.compute(a, b, c);
}
}
```


Output 1:

Enter the co-efficients of a, b and c of a quadratic equation:

5 6 1

Roots are real and distinct

Root 1 = -0.2 Root 2 = -1.0

Output 2:

Enter the co-efficients of a, b and c of a quadratic equation:

1 -4 4

Roots are real and equal
Root 1 = Root 2 = 2.0

Output 3:
Enter the coefficients of a, b and c of a quadratic equation:
-1 1
Roots are imaginary
Root 1 = 0.0 + i 0.8660254037844386
Root 2 = 0.0 - i 0.8660254037844386
•Sneha N Shastri - 1BM22CS283

3) Parse Int demo

Find area of rectangle and verify the same with various inputs (length, breadth)

```
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("Area = " + area);
}
}
```

Output:
1 2
Length of rectangle : 1
Breadth of rectangle : 2
Area = 2

② Lab 2 - 19/12/2023 [19-12-23]

1. Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

$$SGPA = \frac{\sum (\text{course credits})(\text{Grade Points})}{\sum (\text{course Credits})}$$

```
public class Subject
{
 int subjectmarks;
 int credits;
 int grade;
}

import java.util.Scanner;
public class Student
{
Subject sub[];
String name;
String usn;
double sgpa;
Scanner scan=new Scanner(System.in);
Student()
{
sub = new Subject[8];
for(int i=0; i<8; i++)
{
 sub[i] = new Subject();
 scan= new Scanner(System.in);
```

```
public void getStudentDetails()
{
System.out.println("Enter name: ");
name = scan.nextLine();
System.out.println("Enter USN: ");
usn = scan.nextLine();
}
public void getMarks()
{
for(int i=0; i<8; i++)
{
System.out.println("Enter Subject "+(i+1)+" marks: ");
sub[i].subjectmarks = scan.nextInt();
System.out.println("Enter Subject "+(i+1)+" credits: ");
sub[i].credits = scan.nextInt();
if(sub[i].subjectmarks==100)
sub[i].grade = 10;
else if(sub[i].subjectmarks<40)
sub[i].grade = 0;
else
sub[i].grade = (sub[i].subjectmarks/10)+1;
}
}
public void computeSGPA()
{
int sumc=0;
double prod=0;
for(int i=0;i<8;i++)
{
```

```java
sumc=sumc+sub[i].credits;
prod=prod+(sub[i].grade * sub[i].credits);
}
sgpa=prod/sumc;
}
}

public class ~~SGPT~~ Sgpa
{
public static void main(String args[])
{
Student s1=new Student();
s1.getStudentDetails();
s1.getMarks();
s1.computeSGPA();
System.out.println("Name:"+s1.name);
System.out.println("USN:"+s1.usn);
System.out.println("S.No \t Subject Marks \t Credits
                    \t Grade");

for(int i=0;i<8;i++)
{
System.out.println((i+1)+"\t"+s1.sub[i].subject
              marks+"\t"+s1.sub[i].credits
              +"\t"+s1.sub[i].grade);

}
System.out.println("SGPA="+s1.~~grade~~ sgpa);
System.out.println("- - - - - - - - - - -");
```

System.out.println("Sneha N Shastri - 1BM22CS283");

}

}

Output:

```
Enter name:
Sneha
Enter USN:
1BM22CS283
Enter Subject 1 Marks:
90
Enter Subject 1 Credits:
4
Enter Subject 2 Marks:
97
Enter Subject 2 Credits:
4
Enter Subject 3 Marks:
87
Enter Subject 3 Credits:
3
Enter Subject 4 Marks:
88
Enter Subject 4 Credits:
3
Enter Subject 5 Marks:
89
Enter Subject 5 credits:
3
```

Enter Subject 6 Marks:
97
Enter Subject 6 credits:
3
Enter Subject 7 Marks:
90
Enter Subject 7 credits:
1
Enter Subject 8 Marks:
95
Enter Subject 8 Credits:
1

Name: Sneha
USN: 1BM22CS283

S.No	Subject Marks	Credits	Grade
1	90	4	10
2	97	4	10
3	87	3	9
4	88	3	9
5	89	3	9
6	97	3	10
7	90	1	10
8	95	1	10

SGPA = 9.5909

Sneha N Shastri - 1BM22CS283

3) Lab 3-26-12-2023

26-12-23

Book class-object program

```
import java.util.Scanner;
class Books
{
String name;
String author;
int price;
int numPages;

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

}

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

```
Class Main
{
public static void main (String args[])
{
Scanner scan = new Scanner (System.in);
int n, price, numPages;
String name, author;
n = scan.nextInt();
price = scan.nextInt();
numPages = scan.nextInt();
name = scan.next();
author = scan.next
Books b[] = new Books [n];
~~System.out.println(~~
for (int i=0; i<n; i++)
{
System.out.println("~~Boo~~Enter details of Book "+(i+1)+":");
System.out.println("Enter name of book: ");
name = scan.next~~Line()~~;
author = scan.next~~Line()~~;
~~price = scan.nextLine();~~
System.out.println("Enter price of book: ");
price = scan.nextInt();
System.out.println("Enter no. of pages: ");
numPages = scan.nextInt();
b[i] = new Books (name, author, price, numPages);
```

<- System.out.println("Enter no. of books: \n");

<- System.out.println("Enter author name: ");

⑤ Develop a java program to create an abstract class named Shape that contains 2 integers and an empty method named printArea(). Provide 3 classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each of the classes contain only method printArea() that prints area of the given shape. [Dat: 21.1.23

2-1-23

```
import java.util.Scanner;
class InputScanner
{
  Scanner scan;
  InputScanner()
  {
    scan = new Scanner(System.in);
  }
}

abstract class Shape extends InputScanner
{
  double a;
  double b;

  abstract void getInput();
  abstract void printArea();
}

class Rectangle extends Shape
{
  void getInput()
  {
```

```
InputScanner is = new InputScanner();
System.out.println("Enter value of a:");
a=scan.nextDouble();
System.out.println("Enter value of b:");
b=scan.nextDouble();

}
void printArea()
{
double area = a*b;
System.out.println("The area of rectangle is:"+area);

}
}

class Triangle extends Shape
{
void getInput()
{
    Scanner
InputScanner is=new InputScanner();
System.out.println("Enter value of a (base):");
a=scan.nextDouble();
System.out.println("Enter value of b (height):");
b=scan.nextDouble();
}
Void printArea()
{
double area = 0.5*a*b;
System.out.println("The area of triangle is:"+area
}
```

```
}
class Circle extends Shape
{
  double r;
  void getInput()
  {
   InputScanner is= new InputScanner();
   System.out.println("Enter the value of radius:");
   r=Scan.nextDouble();
  }
  Void printArea()
  {
   double area= 3.14 * r * r;
   System.out.println("The area of circle is:"+area);
  }
}

class MainShape
{  public static void main(String args[]) {
 Rectangle r= new Rectangle();
 Triangle t= new Triangle();
 Circle c= new Circle();
 r.getInput();
 r.printArea();
 t.getInput();
 t.printArea();
 c.getInput();
 c.printArea(); } }
```

Output:

Enter value of a:

6

Enter value of b:

2.5

The area of rectangle is 15.0

Enter value of a(base):

6

Enter value of b(base):

8

The area of triangle is:24.0

Enter value of radius:

3

The area of circle is 28.2599999

⑤ Lab 5 – 09/01/2024

Develop a Java program to create class ~~Bank~~ Account that maintains two kinds of account for its customers one saving and other current account.

Savings account provides – compound interest & withdrawal facilities. No cheque book facility

Current account provides – cheque book facility but no interest. Service charge imposed if balance falls below minimum balance.

Create a class Account that stores customer name, acc no, type of acc. From this derive classes Curr-Act and Sav-Acct. Include methods to achieve:

a. Accept deposit from customer and update balance
b. Display balance
c. Compute & deposit interest
d. Permit withdrawal and update balance

```
import java.util.Scanner;
class Bank Account
{
  String cname;
  long accno;
  String type;
  Account( String cn, long ac, String t)
  {
      cname = cn;
      accno = ac;
      type = t;
```

```
}
import java.util.Scanner;
class CurrAct extends Account
{
double balance;
CurrAct ( String cn, long ac, String t, double b)
 {
  super( cn, ac, t);
  balance = b;
 }
public void operations()
 {
 Scanner scan = new Scanner (System.in);
 System.out.println("Enter name : ");
 String s = scan.nextLine();
 System.out.println("Enter account number : ");
 long n = scan.nextLong();
 System.out.println("
do { int c;
System.out.println("Enter choice :");
System.out.println("1. Deposit ");
System.out.println("2. Display balance ");
System.out.println("3. Withdrawal ");
System.out.println("4. Exit ");
c = scan.nextInt();
switch (c)
{
  case 1:
  System.out.println("Enter deposit amount :");
```

```
double deposit = scan.nextDouble();
balance += deposit;
break;
case 2:
System.out.println("Balance is: " + balance);
break;
case 3:
System.out.println("Enter wi
if (balance < 100)
{ System.out.println("Less than minimum balance.
                    Fine of Rs.5");
  balance -= 5;
}
else
 {
  System.out.println("Enter withdrawal amount:");
  double amt = scan.nextDouble();
  balance -= amt;
  System.out.println("Withdrawal Successful. Current
                    Balance = " + balance);
 }
 break;
case 4:
 System.out.println("Thank You.");
 break;
default:
 System.out.println("Invalid choice.");
} } while (c!=4); }
}
```

```
class SavAcct extends Account
{
double balance;
SavAcct (String cn, long ac, String t, double b)
 {
  super (cn, ac, t);
  balance = b;

 }
public void operations()
 {
 Scanner scan = new Scanner(System.in);
 do
 {
  int c;
  System.out.println("1. Deposit.");
  System.out.println("2. Withdraw.");
  System.out.println("3. Compute & deposit interest.");
  System.out.println("4. Display ~~details~~ balance.");
  System.out.println("5. Exit.");

 Switch(c)
 {

  case 1:
  System.out.println("Enter deposit amount:");
  double deposit = scan.nextDouble();
  balance += deposit;
  break;
 Case 2:
 if(balance < 100)
  {
   System.out.println("Less than minimum balance
                       Fine of Rs.5");
```

```
balance -= 5;
}
else
{
    System.out.println("Enter withdrawal amount:");
    double amt = scan.nextDouble();
    balance -= amt;
    System.out.println(amt + " has "Withdrawal
                        successful. Balance = " + balance);
}
break;
Case 3:
    double r = 6.0/100.0;
    System.out.println("Enter duration of account
                        holding.");
    int t=scan.nextInt();
    double interest = balance * Math.pow((1+r), t) -
                        balance;
    balance = balance + interest;
    System.out.println("Bataan Interest: " + interest);
    System.out.println("Balance:" + balance);
    breas;
Case 4:
    System.out.println("Balance: "+balance);
    break;
Case 5:
    System.out.println("Thank You
    break;
default:
```

```
System.out.println("Invalid choice.");

}while(c!=5);


}

class main
{
public static void main(String args[])
{
Scanner scan = new Scanner(System.in);
System.out.println("Enter name:");
String s=scan.nextLine();
System.out.println("Enter account number:");
long ac=scan.nextLong();
System.out.println("Enter account type: Savings/
                    Current");
String t=scan.nextLine();
System.out.println("-----------");
System.out.println("Details:");
System.out.println("Name:"+s+"\n"+
                   "AccNo:"+ac+"\n"+
                   "Acc Type:"+t);

if(t.equalsIgnoreCase("Savings"))
 {
   SavAcct sv=new SavAcct(s,ac,t,
System.out.println("Enter account balance:");
 double b=scan.nextDouble();
if(t.equalsIgnoreCase("Savings"))
{
```

```
SavAcct sv= new SavAcct (s, ac, t, b);
sv.operations();
}
else if (t.equalsIgnoreCase("Current"))
{
  CurrAct ct=new CurrAct (s, ac, t, b);
  ct.operations();
}
}
```

Lab-6 - Strings

Q 6-10

```
import java.util.Scanner;
class StringDemo2
{
public static void main(String args[])
{
//6
System.out.println("Bmsce equals Bmsce = "+
                   ("Bmsce".equals("Bmsce")));
System.out.println("Bmsce equals college = "+
                   ("Bmsce".equals("college")));
System.out.println("Bmsce equals BMSCE = "+
                   ("Bmsce".equals("BMSCE")));

//7
String S1 = "Bmsce college";
String S2 = "Welcome to Bmsce College of Engineering";
if (S1.regionMatches(0,S2,0,13))
System.out.println("substring matched");
else
System.out.println("not matched");

//8
System.out.println("startsWith() demo:");
System.out.println("Does eclipse start with
                   ecl?:" + ("eclipse".startsWith("ecl")));
System.out.println("Does eclipse start with
                   lec?:" + ("eclipse".endsWith("lec")));
```

```
//9
System.out.println("endsWith() demo:");
System.out.println("Does eclipse end with
                    pse?:" + ("eclipse".endsWith
                    ("pse")));
System.out.println("Does eclipse end with ecl?:
                    +("eclipse".endsWith("ecl")));

//10
System.out.println("equals versus ==");
String o="Beautiful";
String p="Beautiful";
System.out.println("String 1:"+o);
System.out.println("String 2:"+p);
System.out.println("Equals:"+(o.equals(p)));
System.out.println("==" + (o==p));
}
}
```

Output:

```
Bmsce equals Bmsce = true
Bmsce equals College = false
Bmsce equals BMSCE = false
Not matched
```

startsWith demo():
does eclipse start with ecl? true
does eclipse ~~start~~ not start with lec? false

endsWith demo():
does eclipse ~~start~~ end with pse? true
does eclipse end with ecl? false

signals versus ==
String 1: Beautiful
String 2: Beautiful
Equals: true
== true

~~intsa~~

Q. 19
Abstract class Fly
Methods fly(), makesound()
Subclasses - Eagle, Hawk

```
abstract class Fly
{
  public abstract void fly();
  public abstract void makesound();
}

class Eagle extends Fly
{
  public void fly()
  {
    System.out.println("The eagle flies high.");
  }
```

```
public void makesound()
 {
    System.out.println("The eagle emits a high pitched whistle.");

 }
}

class Hawk extends Fly
 {
  public void fly()
  {
    System.out.println("The hawk flies lower");
  }
  public void makesound()
  {
    System.out.println("The hawk screeches keaaar");

  }
}

class Birdmain
  {
    public static void main(String args[])
    {
      System.out.println("Invoking eagle!");
      Eagle eg=new Eagle();
      eg.fly();
      eg.makesound();
```

```
System.out.println("Invoking hawk!");
Hawk hk = new Hawk();
hk.fly();
hk.makesound();
}
}
```

Output:

Invoking eagle!
The eagle flies high
The eagle emits a high pitched whistle.
Invoking hawk!
The hawk flies lower
The hawk screeches keaaar

21. Implementation of Stack using Generics:

```
public class Stack<E>
{
  E stck[];
  int top;
  final int SIZE = 10;
  Stack()
  {
   stck = (E[]) new Object[SIZE];
   top = -1;
}
```

```
void push (E item)
{
  if (top == SIZE-1)
  System.out.println ("Stack is full");
  else
  stck [++top] = item;
  }

E pop ()
{
  if(top <0)
  {
    System.out.println("Stack underflow");
    return null;
  }
  else
  return stck [top--];
  }
}

import java.util.Scanner;
public class TestStack
{
  public static void main (String args[])
  {
```

```
Stack <Integer> myStack1 = new Stack
Stack <Integer>();

Stack <Double> myStack2 = new Stack <Double >();
Scanner s = new Scanner (System.in);
System.out.println ("Enter elements into Integer stack:")
for(int i=0; i<5; i++)
{ int n=s.nextInt();
  mystack1.push (n);
}
System.out.println ("Enter elements into the double
                     stack:");

for(int i=0; i<5; i++)
{
  double m=s.nextDouble();
  mystack2.push(m);
}
System.out.println ("Elements of stack 1");
for (int i=0; i<5; i++)
{
  System.out.println (mystack1.pop());
}

for(int i.
System.out.println ("Elements of stack 2");
for (int i=0; i<5; i++)
{
```

```
System.out.println(mystack2.pop());
}
s.close();
}
}
```

Output:

```
Enter elements into the integer stack:
1 2 3 4 5
Enter elements into the double stack:
8.0 9.0 10.5 11.6 12.7

Elements of stack 1:
5
4
3
2
1

Elements of stack 2:
12.7
11.6
10.5
9.0
8.0
```

16/1/24

17-23-01-2024

Package Package CIE → Classes → Student
                                → Internals

Package SEE → External

Create the above Packages and classes to implement package concept

```
package CIE;
import java.util.Scanner;
public class Student
{
String usn;
String name;
int sem;

public void inputStudentDetails()
{
Scanner scan = new Scanner(System.in);
System.out.println("Enter name:");
name = scan.next();
System.out.println("Enter usn:");
usn = scan.next();
System.out.println("Enter sem:");
sem = scan.nextInt();
}

public void displayStudentDetails()
{
System.out.println("Name:"+name);
System.out.println("USN:"+usn);
```

```
System.out.println("Sem:" + sem);
}
}
```

```
package CIE;
import java.util.Scanner;
public class Internals extends Student
{ protected
  int marks[] = new int[5];
   public void inputCIEMarks()
    {
     Scanner scan = new Scanner(System.in);
     System.out.println("Enter internal marks of
                         5 subjects :");
    for (int i=0; i<marks.length; i++)
    {
      marks[i] = scan.nextInt();
    }
  }
}
```

```
package SEE;
Import CIE.Internals;
import java.util.Scanner;
public class External extends Internals
{
  protected int marks[];
```

```
protected int finalMarks[];

public External()
{
 marks = new int[5];
 finalmarks = new int[5];

}
public void inputSEEMarks()
{
Scanner s = new Scanner(System.in);
for(int i=0; i<5; i++)
{
 System.out.println("subject "+(i+1)+" marks:");
 marks[i] = s.nextInt();
}
}

public void calculateFinalMarks()
{
 for(int i=0;i<5;i++)
 {
  finalMarks[i] = marks[i]/2 + super.marks[i];
 }
}

public void displayFinalMarks()
{
   displayStudentDetails();
   for(int i=0; i<5;i++)
   System.out.println("subject "+(i+1)+":"+
                      finalMarks[i]);
```

```
}
}

import SEE.External;
class MainMarks
{
public static void main(String args[])
    {
      int numofstudents = 2;
      External finalMarks[] = new External[numOfStudents];
      for(int i=0 ; i<numOfStudents; i++)
       {
        finalMarks[i]= new External();
        finalMarks[i].inputStudentDetails();
        System.out.println("Enter CIE marks");
        finalMarks[i].inputCIEMarks();
        System.out.println("Enter SEE Marks");
        finalMarks[i].inputSEEMarks();
       }
      System.out.println("Displaying data:\n");
      for(int i=0;i<numofstudents;i++)
      {
       finalMarks[i].calculateFinalMarks();
       finalMarks[i].displayFinalMarks();
}}}
```

Output:

Enter name:
Sneha
Enter usn:
1BM22CS283
Enter sem:
3
Enter CIE marks
Enter internal marks of 5 subjects:
50
49
48
47
46
Enter SEE marks
Subject 1 marks: 99
Subject 2 marks: 98
Subject 3 marks: 90
Subject 4 marks: 95
Subject 5 marks: 96

Enter name:
Siri
Enter usn:
1BM22CS280
Enter sem:
3
Enter CIE marks
Enter internal marks of 5 subjects:
50
45

46
47
49

Enter SEE marks
Subject 1 marks: 90
Subject 2 marks: 99
Subject 3 marks: 8
Subject 4 marks: 98
Subject 5 marks: 90

Displaying data:

Name: Sneha
USN: 1BM22CS283
Sem: 3
Subject 1: 99
Subject 2: 98
Subject 3: 93
Subject 4: 94
Subject 5: 94

30/1/24

Name: Siri
USN: 1BM22CS280
Sem: 3
Subject 1: 95
Subject 2: 94
Subject 3: 50
Subject 4: 96
Subject 5: 94

30/1/2024

Write a program that demonstrates 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<0. In Son class, implement a constructor that cases both father and son's age and throws an exception if son's age >= father's age.

```
class WrongAge extends Exception
{
 WrongAge(String s)
 {
  super(s);
 }
}

import java.util.Scanner;
class Father
int age;
Scanner scan= new Scanner(System.in);
Father() throws WrongAge
{
 System.out.println("Enter Father's age:");
 age =scan.nextInt();
 if(age < 0)
 throw new WrongAge("Age cannot be negative.");
```

```
}
}

import java.util.Scanner;
class Son extends Father
{
  int sonage;
  Scanner scan = new Scanner(system.in);

  Son() throws WrongAge
  {
    if(sonage>=age)
    throw newWrongAge("Son's age cannot be greate
                       than father's age.");
  }

  public void display()
  {
    System.out.println("Father's age:"+age);
    System.out.println("Son's age:" + son
                                      age)
    System.out.println("Sneha N Shastri - 1BM22CS281
  }
}

class AgeMain
{
public static void main(String args[])throws Exception
{
  Son s=new Son();
```

```
    check();
    display();
  }
}
```

Output:

```
Enter father's age:
16
Enter son's age:
25
Exception in thread "main" WrongAge: Son's age cannot be greater than father's age.

Enter father's age:
-5
Exception in thread "main" WrongAge: Age cannot be negative.

Enter father's age:
50
Enter son's age:
-2
Exception in thread "main" WrongAge: Age cannot be negative

Enter father's age:
50
Enter son's age:
20
Father's age: 50
Son's age: 20
```

30/11/24

Sneha N Shastri - 1BM22CS283

6/2/24

(8) Write a program which creates two threads, one displaying "BMS College of Engineering" once every ten seconds and another displaying "CSE" once every two seconds

```
class NewThread implements Runnable
{
Thread t;
NewThread()
{
  t = new Thread(this, "NThread");
  System.out.println("CT:" + t);
  t.start();
  }
public void run()
{
 try
 {
  for(int n = 5; n > 0; n--)
  {
    System.out.println("CSE");
    Thread.sleep(2000);
  }
 }
catch (InterruptedException ie)
{
```

```
System.out.println("CSE thread interrupted");
}
System.out.println("CSE thread quitting");
}
}
class Thread2
{
public static void main(String args[])
{
new NewThread();
System.out.println("Back in Main");
try
{
for(int n=5;n>0;n--)
{
System.out.println("BMS college of Engineering");
Thread.sleep(10000);
}
}
catch(InterruptedException ie)
{
System.out.println("BMS thread interrupted");
}
System.out.println("BMS thread quitting");
System.out.println("Sneha N Shastri - 1BM22CS283");
}
}
```

Output:

CT: Thread [# 29, NThread, 5, main]
Back in main
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE thread quiting
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS thread quitting
Sneha N Shastri - 1BM22CS283

7) Demonstrate Inter process communication and deadlock

Inter Process Communication

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

Synchronized void put(int n)
{
while (valueSet)
   try
   {
```

```
System.out.println("Producer waiting");
wait();
} catch(InterruptedException e)
{
 System.out.println("Interrupted Exception caught");
 }

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

// Producer class
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<=5)
{
q.put(i++);
}
}
}
```

```
// Consumer class
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<=5)
{
int x = q.get();
System.out.println("consumed: " + x);
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.");
        System.out.println("Sneha N Shastri - 1BM22CS283");
    }
}
```

Output:

```
Press Control-C to stop.
Sneha N Shastri - 1BM22CS283
Put : 0
Intimate Consumer
Producer waiting
Got : 0
Intimate Producer
Put : 1
Intimate Consumer
Producer waiting
Consumed : 0
Got : 1
```

Intimate Producer
Consumed : 1
Put : 2
Intimate Consumer
Producer waiting
Got : 2
Intimate Producer
Consumed = 2
Put : 3
Intimate Consumer
Producer waiting
Got : 3
Intimate Producer
Consumed : 3
Put : 4
Intimate Consumer
Producer waiting
Got : 4
Intimate Producer
Consumed : 4
Put : 5
Intimate Consumer
Got : 5
Intimate Producer
Consumed : 5

5. 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 = new Thread(this, "Racing Thread");
t.start();
a.foo(b);
System.out.println("Back in main thread");
}

public void run()
{
  b.bar(a);
  System.out.println("Back in other thread");
}  <- System.out.println("Sneha N Shastri-1BM22CS283");

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

Output:

```
RacingThread entered B.bar
MainThread entered A.foo
MainThread trying to call B.last()
Inside A.last
Back in main thread
RacingThread trying to call A.last()
Inside A.last
Back in other thread
Sneha N Shastri
-1BM22CS283
```

13.02.24