ANAND INSTITUTE OF HIGHER TECHNOLOGY

OLD MAHABALIPURAM ROAD, KALASALINGAM NAGAR

KAZHIPATTUR, NEAR CHENNAI-603 103

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB MANUAL

Subject Code : CS2405

Subject Name : OBJECT ORIENTED PROGRAMMING LABORATORY

Degree : B.E

Branch : COMPUTER SCIENCE AND ENGINEERING

Year : II

Semester : III

Regulation : 2024

INDEX

Ex. No. 1.A	SEQUENTIAL SEARCH
Date:	

Aim:

To write a Java program to solve problems by using Sequential Search.

Algorithm:

Step1: start
Step 2: Get the array elements. Step 3:
Traverse the array.
Step 4: Match the key element with array element.
Step 5: If key element is found, return the index position of the array element. Step 6:
If key element is not found, print element not found message.
Step 7: stop

PROGRAM

//Exno:1(a) Sequential search

```
import java.util.Scanner;
class LinearSearch
{
public static void main(String[] args)
{
int c,n,search,array[];
Scanner in = new Scanner(System.in);
System.out.println("Enter number of elements");
n=in.nextInt();
array=new int[n];
System.out.println("Enter those " + n + " elements");
for(c=0;c<n;c++)
array[c]=in.nextInt();
System.out.println("Enter value to find");
search=in.nextInt();
for(c=0;c<n;c++)
{
if(array[c]==search)
{
System.out.println(search + " is present at location " + (c + 1) + "."); break;
}
}
if(c==n)
System.out.println(search + " isn't present in array.");
}
}
```

Output:

```
javac LinearSearch.java java
LinearSearch
Enter number of elements 5 Enter
those 5 elements 3
5
7
8
10
Enter value to find 8
8 is present at location 4.
```

Result:

Thus the Java application to solve the problems by using Sequential Search was developed successfully.

Ex. No.1.B	BINARY SEARCH
Date:	

Aim:

To write a Java program to solve problems by using Binary Search.

Algorithm:

Step1: Start
Step 2: Get the array elements.
Step 3: Calculate the mid element of the collection. Step 4:
Compare the key items with the mid element.
Step 5: If key item < mid element, then the key is in the upper half of the collection. Hence you need search in the upper half (mid +1).
Step 6: Else if key item = middle element, then we return the mid index position for the key found.
Step 7: Else key item > mid element, then the key lies in the lower half of the collection. Thus repeat binary search on the lower (right) half of the collection.
Step 8: Stop

PROGRAM

```
//Exno:1(b) Binary search
import java.util.Scanner; class
BinarySearchExample
{
public static void main(String args[])
{
int counter, num, item, array[], first, last, middle;
Scanner input = new Scanner(System.in);
System.out.println("Enter number of elements:"); num
= input.nextInt();
array = new int[num];
System.out.println("Enter " + num + " integers"); for
(counter = 0; counter < num; counter++) array[counter]
= input.nextInt(); System.out.println("Enter the search
value:"); item = input.nextInt();
first = 0;
last = num - 1;
middle = (first + last)/2; while
(first <= last)
{
if (array[middle] < item) first =
 middle + 1;
else if(array[middle] == item)
{
System.out.println(item + " found at location " + (middle + 1) + "."); break;
}
else
{
last = middle - 1;
}
middle = (first + last)/2;
}
if ( first> last )
System.out.println(item + " is not found.\n");
}
}
```

Output:

```
javac BinarySearchExample.java java
BinarySearchExample Enter number
of elements:
5
Enter 5 integers
8
9
12
15
16
Enter the search value: 9
9 found at location 2.
```

Result:

Thus the Java application to solve the problems by using Binary Search was developed successfully.

Ex. No.1.C

Date:

QUADRATIC SORTING ALGORITHM

SELECTION SORT

Aim:

To write a Java program to solve problems by using Selection Sort.

Algorithm:

Step 1: Start
Step 2: Initialize minimum value (min_index) to location 0.
Step 3: Traverse the array to find the minimum element in the array.
Step 4: While traversing if any element smaller than min_index is found then swap both the values.
Step 5: Then, increment min_index to point to the next element. Step
6: Repeat until the array is sorted.
Step 7: Stop

PROGRAM

```
//Exno:1(c) Selection sort import
java.util.Scanner; public class
selectionsort
{
public static void main(String args[])
{
int size, i, j, temp;
int arr[] = new int[50];
Scanner sc = new Scanner(System.in);
System.out.print("Enter Array Size : "); size =
sc.nextInt();
System.out.print("Enter Array Elements : \n");
for(i=0; i<size; i++)
{
arr[i] = sc.nextInt();
}
for(i=0; i<size; i++)
{for(j=i+1; j<size; j++)
{
if(arr[i] >arr[j])
{
```

```
temp    =    arr[i];
arr[i]    =    arr[j];
arr[j] = temp;
}
}}
System.out.print("sorted array >>\n");
for(i=0; i<size; i++)
{
System.out.print(arr[i]+ " ");
}}}
```

Output:

```
javac selectionsort.java
java     selectionsort
Enter Array Size : 6 Enter Array Elements : 7
2
9
4
6
3
sorted array >> 2 3 4 6 7 9
```

Result:

Thus the java program for selection sort was developed and the output was verified successfully.

Ex. No.1.D **QUADRATIC SORTING ALGORITHM**

Date: **INSERTION SORT**

Aim:

To write a Java program to solve problems by using Insertion Sort.

Algorithm:

Step1: Start
Step 2: Get the elements
Step 3: If the element is the first element, assume that it is already sorted. Step 4:
Pick the next element, and store it separately in a key.
Step 5: Now, compare the element with all elements in the sorted array.
Step 6: If the element in the sorted array is smaller than the current element, then move to the next element. Else, shift greater elements in the array towards the right.
Step 7: Insert the value.
Step 8: Repeat until the array is sorted. Step 9:
Stop

PROGRAM

//Exno:1(d) Insertion sort

```
import java.util.Scanner;
public class InsertionSort
{
public static void main(String[] args)
{
int n, i, j, element;
Scanner scan = new Scanner(System.in);
System.out.print("Enter the Size of Array: "); n =
scan.nextInt();
int[] arr = new int[n]; System.out.print("Enter "
+n+ " Elements: "); for(i=0; i<n; i++)
arr[i] = scan.nextInt();
for(i=1; i<n; i++)
{
element = arr[i];
for(j=(i-1); j>=0 &&arr[j]>element; j--)
arr[j+1] = arr[j];
arr[j+1] = element;
}
System.out.println("\nThe new sorted array is: "); for(i=0; i<n; i++)
System.out.print(arr[i]+ " ");
}
}
```

Output:

```
javac InsertionSort.java
java     InsertionSort
Enter the Size of Array: 6 Enter 6 Elements: 3
6
9
7
4
2
The new sorted array is: 2 3
4 6 7 9
```

Result:

Thus the java program for insertion sort was developed and the output was verified successfully.

Ex. No.2.A **STACK**

Date:

Aim:
To Develop stack data structures using classes and objects.

Algorithm:
The stack supports the following operations:

1. push inserts an item at the top of the stack (i.e., above its current top element).
2. pop removes the object at the top of the stack and returns that object from the function. The stack size will be decremented by one.
3. isEmpty tests if the stack is empty or not.
4. isFull tests if the stack is full or not.
5. peek returns the object at the top of the stack without removing it from the stack or modifying the stack in any way.
6. size returns the total number of elements present in the stack.

PROGRAM

```
//Exno:2(a)Stack import
java.util.Stack; class Main
{
public static void main(String[] args)
{
Stack<String> stack = new Stack<String>();

stack.push("A");  //  Insert  `A`  into  the  stack
stack.push("B");  //  Insert  `B`  into  the  stack
stack.push("C");  //  Insert  `C`  into  the  stack
stack.push("D"); // Insert `D` into the stack

// prints the top of the stack (`D`) System.out.println("The top
element is " + stack.peek());

stack.pop();      // removing the top element (`D`)
stack.pop();      // removing the next top (`C)

// prints the top of the stack (`B`) System.out.println("The top
element is " + stack.peek());

// returns the total number of elements present in the stack System.out.println("The
stack size is " + stack.size());

// check if the stack is empty if
(stack.empty()) {
System.out.println("The stack is empty");
}
else {
System.out.println("The stack is not empty");
}
}
}
```

Output

The top element is D The
top element is B The stack
size is 2 The stack is not

empty

I OMoARcPSD|283 59 27

Result:

Thus the Java program for stack data structure using classes and objects was developed and the output was verified successfully.

Ex. No.2.B **QUEUE**

Date:

Aim:

To develop QUEUE data structures using classes and objects.

Algorithm:

The queue supports the following core operations:

1. Enqueue: Inserts an item at the rear of the queue.
2. Dequeue: Removes the object from the front of the queue and returns it, thereby decrementing queue size by one.
3. Peek: Returns the object at the front of the queue without removing it.
4. IsEmpty: Tests if the queue is empty or not.
5. Size: Returns the total number of elements present in the queue.

PROGRAM

//Exno:2(b)Queue

```
import java.util.LinkedList;
import java.util.Queue; public
class QueueExample
{
public static void main(String[] args)
{
Queue<Integer> q= new LinkedList<>();

// Adds elements {0, 1, 2, 3, 4} to
// the queue
for(int i = 0; i< 5; i++)
q.add(i);

// Display contents of the queue.
System.out.println("Elements of queue " + q);

// To remove the head of queue. int
removedele = q.remove();
System.out.println("removed element-"+ removedele);
System.out.println(q);
// To view the head of queue int
head = q.peek();
System.out.println("head of queue="+head);
// Rest all methods of collection
// interface like size and contains
// can be used with this
// implementation. int
 size = q.size();
System.out.println("Size of queue="+size);
}
}
```

Output

```
javac QueueExample.java
java QueueExample
Elements of queue [0, 1, 2, 3, 4]
removed element-0
[1, 2, 3, 4]
```

head of queue=1
Size of queue=4

I OMoARcPSD283 59 27

Result:

Thus the Java program for Queue data structure using classes and objects was developed and the output was verified successfully.

ExNo.3 **GENERATINGPAY SLIPS FOR THE EMPLOYEES**

Date:

Aim:

To develop a java application for generating pay slips for the employees with their gross and net salary.

Algorithm:

Step 1. Start

Step 2. Get the information about the employee – Employee name. Employee id, address, mail id and mobile number

Step 3:Using function calculate the salary

Grosssalary=(0.97f*bpay)+(0.10f*bpay)+(0.12f*bpay)+(0.001f*bpay)+bpay;

Netsalary=grosssalary-((0.12f*bpay)+(0.001f*bpay)

Step 4:Using switch case, get the choice (Professor, Associate Professor, Assistant Professor,Programmer) and the required function called.

Step 6:Using function calculate the salary by the input as bpay. Step 7:Choose the employee and print the details.

Step 8:Stop.

PROGRAM

//Exno:3 GENERATING PAY SLIPS FOR THE EMPLOYEE

```
import java.util.*;
import java.io.*; class
Employee
{
int id;
float grosssalary, netsalary,bpay; String
name, address, mail, mno; Scanner get =
new Scanner(System.in); Employee()
{
System.out.println("Enter Name of the Employee:"); name =
get.next();
System.out.println("Enter id:"); id =
get.nextInt();
System.out.println("Enter Address of the Employee:");
address = get.next();
System.out.println("Enter mailid of the Employee:"); mail =
 get.next();
System.out.println("Enter mobile no. of the Employee:"); mno
= get.next();
}
void salary()
{
grosssalary= (0.97f*bpay)+(0.10f*bpay)+(0.12f*bpay) + (0.001f*bpay)+bpay;
netsalary= grosssalary-((0.12f*bpay) + (0.001f*bpay));
}
void display()
{
System.out.println("Employee Name: "+name);
System.out.println("ID            : "+id);
System.out.println("Mail Id               : "+mail);
System.out.println("Address                       : "+address);
System.out.println("Mobile No. : "+mno);
System.out.println("Gross Salary : "+grosssalary);
System.out.println("Net Salary : "+netsalary);
}
}
class Programmer extends Employee
{
```

```
Programmer()
{
```

```
System.out.println("Enter Basic Pay:"); bpay
= get.nextFloat();
salary();
}
void display()
{
System.out.println("\n ==============\n Programmer\n ================\n");
super.display();
}
}
class AssistantProfessor extends Employee
{
AssistantProfessor()
{
System.out.println("Enter Basic Pay:"); bpay
= get.nextFloat();
salary();
}
void display()
{
System.out.println(" ====================\n Assistant Professor\n=========\n");
super.display();
}
}
class AssociateProfessor extends Employee
{
AssociateProfessor()
{
System.out.println("Enter Basic Pay:"); bpay
= get.nextFloat();
salary();
}
void display()
{
System.out.println(" ================\n Associate Professor\n============\n");
super.display();
}
}
class Professor extends Employee
{
```

```
Professor()
{
System.out.println("Enter Basic Pay:");
```

```
bpay = get.nextFloat();
salary();
}
void display()
{
System.out.println(" ==============\n Professor\n ==============\n");
super.display();
}
}
class Employees
{
public static void main(String args[]) throws Exception
{
BufferedReaderbr = new BufferedReader(new InputStreamReader(System.in));
System.out.println(" -------------\n PAY SLIPS \n          \n 1. Professor \n 2. Associate
Professor\n 3. Assistant Professor\n 4. Programmer");
System.out.printf("\n Choose the type of Employee: "); int
choice=Integer.parseInt(br.readLine()); switch(choice)
{
case 1:
System.out.println(" ==============\n Enter Professor Details \n========"+"\n"); Professor
ob1 = new Professor();
 ob1.display(); break;
case 2:
System.out.println(" ==========\n Enter Associate Professor Details \n ====="+"\n");
AssociateProfessor ob2 = new AssociateProfessor();
 ob2.display(); break;
 case 3:
System.out.println(" ========\n Enter Assistant Professor Details\n ======="+"\n");
AssistantProfessor ob3 = new AssistantProfessor();
 ob3.display(); break;
case 4:
System.out.println(" ========\n Enter Programmer Details \n=========="+"\n");
Programmer ob4 = new Programmer();
 ob4.display(); break;
 default:
System.out.println("Enter correct choice");
break;
}
```

}
}

OUTPUT

D:\demo>javac Employees.java D:\demo>java
Employees

PAY SLIPS

1. Professor
2. AssociateProfessor
3. Assistant Professor
4. Programmer

Choose the type of Employee: 1
==========
Enter Professor Details
==========

Enter Name of the Employee: Dina Enter
id:125
Enter Address of the Employee: nagarkoil Enter
mailid of the Employee: din@gmail.com Enter
mobile no. of the Employee: 9445861253 Enter
Basic Pay: 20000

==========
Professor
==========
Employee Name: Dina ID
125
Mail Id: din@gmail.com Address :
nagarkoil Mobile No. : 9445861253
Gross Salary : 43820.0 Net
Salary : 41400.0 D:\demo>

Result:

Thus the java application for generating pay slips for the employees with their gross and netsalary was developed successfully.

ExNo.4 FINDING AREA OF DIFFERENT SHAPES USING ABSTRACT CLASS

Date:

Aim:

To write a JAVA program for finding area of different shapes using abstract class concept.

Algorithm:

Step 1:Start.
Step 2: Create an abstract class named Shape that contains two integers and an empty method named print Area()
Step 3:Get the value of length and breadth of rectangle.
Step 4:Calculate the area of rectangle using area=length * breadth Step
5:Print the area of rectangle.
Step 6:Get the value of length anf height of triangle.
Step 7:Calculate the area of triangle using area = 0.5 * length * height Step
8:Print the area of circle.
Step 9:Get the value of radius.
Step 10:Calculate the area of circle using area =Math.PI * radius * radius. Step
11:Print the area of circle.
Step 12:Stop.

PROGRAM

//Exno:4 Finding Area of Different Shapes Using Abstract Class

```
import java.util.*;
abstract class Shape
{
int length = 0; int
height = 0;
public abstract void printArea();
}
class Rectangle extends Shape
{
int area = 0;
public void printArea()
{
System.out.println("\n Rectangle\n       ");
Scanner input = new Scanner(System.in);
 System.out.printf("Enter Length of Rectangle : ");
this.length = input.nextInt(); System.out.printf("Enter
Breadth of Rectangle : "); this.height = input.nextInt();
this.area = this.length*this.height;
System.out.println("Area of the Rectangle is : " + this.area);
}
}
class Triangle extends Shape
{
double area = 0.0; public
void printArea()
{
System.out.println("\n Triangle\n         ");
Scanner input = new Scanner(System.in);
 System.out.printf("Enter Length of Triangle : ");
this.length = input.nextInt(); System.out.printf("Enter
Height of Triangle : "); this.height = input.nextInt();
this.area = 0.5 * this.length * this.height; System.out.println("Area
of the Triangle is : " + this.area);
}
}
class Circle extends Shape
{
```

```
double area = 0.0;
```

```
public void printArea()

{
System.out.println("\n Circle\n            ");
Scanner input = new Scanner(System.in);
System.out.printf("Enter Radius of Circle : "); this.length =
input.nextInt();
this.area = Math.PI * this.length * this.length;
System.out.println("Area of the Circle is : "+this.area);
 }
}
class Absclass
{
public static void main(String[] args)
{
 System.out.println("\n                \nFinding Area\n");
Shape rt = new Rectangle();
 rt.printArea();
Shape tr = new Triangle();
tr.printArea();
Shape cr = new Circle();
cr.printArea();
}
}
```

OUTPUT

```
D:\demo>javac Absclass.java D:\demo>java
Absclass
---------

Finding Area Rectangle
Enter Length of Rectangle : 20
Enter Breadth of Rectangle : 10
Area of the Rectangle is : 200

 Triangle
Enter Length of Triangle : 10
Enter Height of Triangle       20
Area of the Triangle is : 100.0

 Circle
Enter Radius of Circle : 5
```

Area of the Circle is : 78.53981633974483

Result:

Thus the JAVA program for finding

area of different shapes using abstract class concept was written, executed and the output was verified successfully.

Ex. No. 5 FINDING AREA OF DIFFERENT SHAPES USING INTERFACE

Date:

Aim:

To write a java program for finding area of different shapes using interface concept.

Algorithm:

Step 1: Start.
Step 2: Create an interface named Shape that contains an empty method named print Area() Step 3:
Get the value of length and breadth of rectangle.
Step 4: Calculate the area of rectangle using area=length * breadth Step 5:
Print the area of rectangle.
Step 6: Get the value of length and height of triangle.
Step 7: Calculate the area of triangle using area = 0.5 * length * height Step 8:
Print the area of circle.
Step 9: Get the value of radius.
Step 10: Calculate the area of circle using area =Math.PI * radius * radius. Step 11:
Print the area of circle.
Step 12: Stop.

PROGRAM

//Exno:5 Finding Area of Different Shapes Using Interface

```
import java.util.*;
interface Shape
{
void printArea();
}
class Rectangle implements Shape
{
int area, length,height; public
void printArea()
{
System.out.println("\n Rectangle\n"); Scanner
input = new Scanner(System.in);
System.out.printf("Enter Length of Rectangle : ");
this.length = input.nextInt(); System.out.printf("Enter
Breadth of Rectangle : "); this.height = input.nextInt();
this.area = this.length*this.height;
System.out.println("Area of the Rectangle is : " + this.area);
}
}
class Triangle implements Shape
{
double area = 0.0; int
length,height ;
public void printArea()
{
System.out.println("\n Triangle\n"); Scanner
input = new Scanner(System.in);
System.out.printf("Enter Length of Triangle : ");
this.length = input.nextInt(); System.out.printf("Enter
Height of Triangle : "); this.height = input.nextInt();
this.area = 0.5 * this.length * this.height; System.out.println("Area
of the Triangle is : " + this.area);
}
}
class Circle implements Shape
{
double area = 0.0; int
```

length;

```
    public void printArea()
{

System.out.println("\n Circle\n          "); Scanner
input = new Scanner(System.in);
System.out.printf("Enter Radius of Circle : ");
this.length = input.nextInt();
this.area = Math.PI * this.length * this.length;
System.out.println("Area of the Circle is : "+this.area);

}
}
class demoforinterface
{
public static void main(String[] args)
{
System.out.println("\n                \nFinding Area\n");
Shape rt = new Rectangle();
rt.printArea();
Shape tr = new Triangle();
 tr.printArea();
Shape cr = new Circle();
cr.printArea();
}
}
```

OUTPUT

```
D:\demo>javac demoforinterface.java
D:\demo>java demoforinterface
---------

Finding Area Rectangle
Enter Length of Rectangle : 3 Enter
Breadth of Rectangle : 2 Area of the
Rectangle is : 6 Triangle
Enter Length of Triangle : 6 Enter
Height of Triangle : 9 Area of the
Triangle is : 27.0 Circle
Enter Radius of Circle : 5
Area of the Circle is : 78.53981633974483
```

Result:

Thus the JAVA program for finding area of different shapes using interface concept was written, executed and the output was verified successfully.

Ex.No.6	**USER DEFINED EXCEPTION HANDLING**
Date:	

Aim:

To write a Java program to implement user defined exception handling.

Algorithm:

Step1: Start.
Step2: Create a class MyException that extends Exception class Step
3: Get an input
Step4: Check whether the input is greater than or equal to zero.
a) If yes, print the number.
b) Otherwise, handle the exception using try-catch block
Step 5: Stop.

PROGRAM

```
//Exno:6 User defined exception handling
import java.io.*; import
java.util.*;
class MyException extends Exception
{
MyException(String message)
{
super(message);
}
}
class demo
{
public static void main(String a[])
{
Scanner s=new Scanner(System.in); try
{
System.out.println ("Enter a positive number:"); int
no=s.nextInt();
if(no<0)
{
throw new MyException("Number must be positive");
}
System.out.println("Number:"+no);
}
catch(MyException e)
{

System.out.println("Caught the Exception"); System.out.println(e.getMessage());
System.out.println ("Exception Handled          !");
}
catch(Exception e)
{
System.out.println("Enter numbers...Exception Handled!");
}
}
}
```

OUTPUT RUN1

Enter a positive number:
-6
Caught the Exception
Number must be positive
Exception Handled !

RUN2

Enter a positive number: 6
Number:6

Result:

Thus the Java program to implement user defined exception handling was implemented and the

output was verified successfully.

Ex.No.7 Date:	**MULTI-THREADED APPLICATION**

Aim:

To write a java program that implements a multi-threaded application that has three threads. First thread generates a random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.

Algorithm:

Step1:Start.
Step2:Create three classes.Two classes (even,odd) implements Runnable and one thread extends Thread Class.
Step3:First thread generates a random integer every 1 second.
Step4: The integer value is even, these condition thread computes the square of the number and prints.
Step 5: If the value is odd, the third thread will print the value of cube of the number. Step6: Stop

PROGRAM

//Exno:7 Multi-threaded application

```
import java.util.*;
class even implements Runnable
{
public int x; public
even(int x)
{
this.x= x;
}
public void run()
{
System.out.println("Thread1:"+x+"is EVEN and Square of"+x+"is:"+x*x);
}
}
class odd implements Runnable
{
public int x; public
 odd(int x)
{
this.x= x;
}

public void run()
{
System.out.println("Thread2:"+x+"is ODD and Cube of"+x+"is:"+x*x*x);
}
}
class A extends Thread
{
public void run()
{
int num=0;
Random r=new Random(); try
{
for(int i=0;i <5; i++)
{
num=r.nextInt(100);
```

```
System.out.println("Thread-1:GeneratedNumberis"+num);  if
(num % 2 == 0)
{
```

```
    Thread t1=new Thread(new even(num));
    t1.start();
}
else
{
Thread t2=new Thread(new odd(num));
t2.start();
}
//Thread.sleep(1000);
System.out.println("");
}
}
catch(Exception ex)
{
System.out.println(ex.getMessage());
}
}
}
public class ThreeThreads
{
public static void main(String[] args)
{
A x=new A();
x.start();
}
}
```

OUTPUT

```
Thread-1: Generated Number is 39
Thread-1: Generated Number is 48
Thread-1: Generated Number is 48
Thread-1: Generated Number is 67
Thread-1: Generated Number is 45
Thread2:48is ODD and Cube of 48 is:110592 Thread1:48is
EVEN and Square of 48 is:2304 Thread2:45is ODD and
Cube of 45 is:91125 Thread1:45is EVEN and Square of 45
is:2025 Thread1:48is EVEN and Square of 48 is:2304
Thread2:48is ODD and Cube of 48 is:110592 Thread2:39is
ODD and Cube of 39 is:59319 Thread2:67is ODD and
Cube of 67 is:300763 Thread1:67is EVEN and Square of
```

67 is:4489 Thread1:39is EVEN and Square of 39 is:1521

Result:

Thus the java program that implements a multi-threaded application that has three threads was written and the output was verified successfully.

Ex. No. 8	FILE HANDLING
Date:	

Aim:

To write a Java program to perform file operations.

Algorithm:

Step 1:Start.
Step 2:Get the input as filename.
Step 3:If the filename exists, then
a) Print the details about the file (Is File?, Is Directory?, Is Readable?, Is Writable?, Type, Length of the file.
b) Otherwise, print “File does not exist”
Step 4:Stop.

PROGRAM

```
//Exno:8 File Handling
import java.util.Scanner;
import java.io.File; import
java.io.*;
import java.util.*; class
Main
{
public static String getFileExtension(File f1)
{
String fileName = f1.getName();
if(fileName.lastIndexOf(".") != -1 && fileName.lastIndexOf(".") != 0) return
fileName.substring(fileName.lastIndexOf(".")+1);
else
return "Folder";
}
public static void main(String[] args)
{
Scanner input=new Scanner(System.in);
System.out.print("\nEnter the filename: ");
String s=input.nextLine();
File f1=new File(s);
 if(f1.exists())
{
try
 {
FileWriter myWriter = new FileWriter(f1);
myWriter.write("AIHT provides quality Education in Engineering and Higher Technology and
catering to the needs of the student community of Metropolitan city"); myWriter.close();
System.out.println("Successfully wrote to the file.");
Scanner myReader = new Scanner(f1);
while (myReader.hasNextLine())
 {
 String data = myReader.nextLine();
System.out.println("       File content ");
System.out.println(data);
}
}
```

```
catch(Exception e)
{
```

```
    System.out.println(e);
    }
System.out.println("\nDETAILS ABOUT THE FILE");System.out.println("       ");

System.out.println(" File exists in : "+f1.getAbsolutePath());
System.out.println("\n Is file?:"+f1.isFile()); System.out.println(" Is
Directory?                              :"+f1.isDirectory());
System.out.println("\n Is Readable?              :"+f1.canRead());
System.out.println(" Is Writable?       :"+f1.canWrite());
System.out.println("\n Type :"+getFileExtension(f1));
System.out.println("\n Length of the File :"+f1.length()+" Bytes");
}
else
{
System.out.println("File does not exist");
}
}
}
```

OUTPUT:

```
Enter the filename: jj.txt Successfully wrote to the file.
----File content----
AIHT provides quality Education in Engineering and Higher Technology and catering to the
needs of the student community of Metropolitan city
DETAILS ABOUT THE FILE
File exists in : D:\ASB\jj.txt Is file? :true Is
Directory? :false
Is Readable?    :true Is Writable? :true
Type :txt
Length of the File :140 Bytes
```

Result:

Thus the Java program for reading a file name from the user and displaying the information

about the file was written and the output was verified successfully.

Ex. No. 9	FINDING MAXIMUM AND MINIMUM USING GENERIC CLASSES
Date:	

Aim:

To write a java program for finding the maximum and minimum value from the given type of elements using a generic class.

Algorithm:

Step 1: Start.
Step 2: Create a class Myclass to implement generic class and generic methods. Step 3:
Get the set of values belonging to specific data type.
Step 4: Create the objects of the class to hold Integer,Character and Double values.
Step 5: Create the methods to compare the values and find the maximum value and minimum value stored in the array.
Step 6: Invoke the methods with integer, character or double values.
Step 7: The maximum and minimum value of Integer ,character and double is displayed based on the data type passed to the method.
Step 8: Stop.

PROGRAM

//Exno:9 Finding Maximum and Minimum using Generic Classes

```
class MyClass<T extends Comparable<T>>
{
T[] vals;
MyClass(T[] obj)
{
vals = obj;
}
public T min()
{
T v = vals[0];
for(int i=1; i < vals.length; i++)
 if(vals[i].compareTo(v) < 0)
v = vals[i]; return v;
}
public T max()
{
T v = vals[0];
for(int i=1; i < vals.length;i++)
if(vals[i].compareTo(v) > 0)
 v = vals[i]; return v;
}
}
class gendemo
{
public static void main(String args[])
{
int i;
Integer inums[]={10,2,5,4,6,1}; Character
chs[]={'j','p','x','a','n','d'};
Double d[]={20.2,45.4,71.6,88.3,54.6,10.4};
MyClass<Integer> iob = new MyClass<Integer>(inums); MyClass<Character> cob
= new MyClass<Character>(chs); MyClass<Double>dob = new
MyClass<Double>(d); System.out.println("Max value in inums: " + iob.max());
System.out.println("Min value in inums: " + iob.min()); System.out.println("Max
value in chs: " + cob.max()); System.out.println("Min value in chs: " + cob.min());
```

```
System.out.println("Max value in d: " + dob.max());
System.out.println("Min value in d: " + dob.min());
}
}
```

OUTPUT:

```
Max value in inums: 10Min value in
inums: 1 Max value in chs: x Min
value in chs: a Max value in d: 88.3
Min value in d:
10.4
```

Result:

Thus the java program for finding the maximum and minimum value from the given type of

elements was implemented using a generic class and executed successfully.

Ex. No. 10	**JAVAFX CONTROLS, LAYOUTS AND**
Date:	**MENUS**

Aim:

To write a java program to develop applications using JavaFX controls, layouts and menus.

Algorithm:

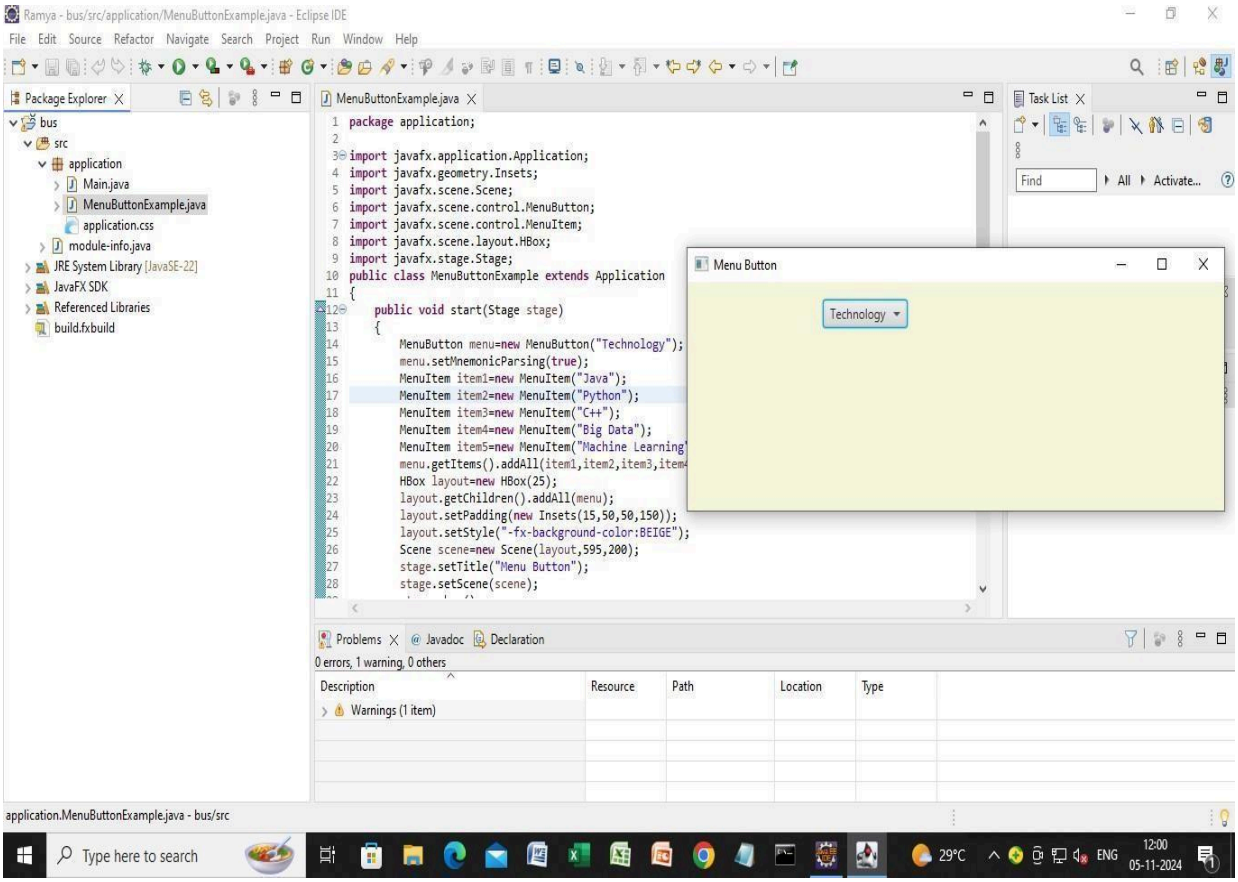
Step 1: Start
Step 2: Create a menu Button named "Technology".
Step 3: Create necessary menu items and add it add it to the menu. Step 4:
Add the choice box to the scene.
Step 5: Set the stage and launch. Step
6: Stop

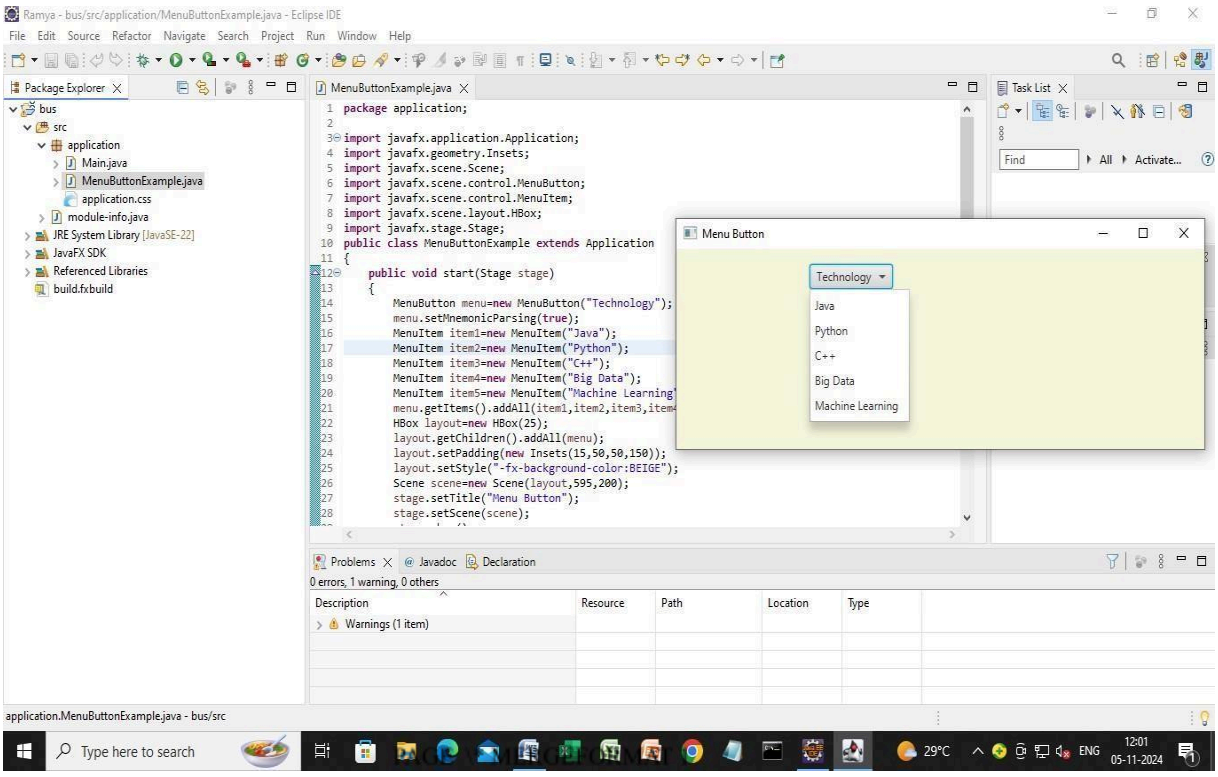
PROGRAM

//Exno:10 Applications using JavaFX controls, layouts and menus

```
import javafx.application.Application; import
javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.MenuButton; import
javafx.scene.control.MenuItem; import
javafx.scene.layout.HBox;
import javafx.stage.Stage;
public class MenuButtonExample extends Application
{
public void start(Stage stage)
{
MenuButton menu = new MenuButton("Technology");
menu.setMnemonicParsing(true);
MenuItem item1 = new MenuItem("Java"); MenuItem
item2 = new MenuItem("Python"); MenuItem item3 =
new MenuItem("C++"); MenuItem item4 = new
MenuItem("Big Data");
MenuItem item5 = new MenuItem("Machine Learning");
menu.getItems().addAll(item1, item2, item3, item4, item5);
HBox layout = new HBox(25);
layout.getChildren().addAll(menu);
layout.setPadding(new Insets(15, 50, 50, 150));
layout.setStyle("-fx-background-color: BEIGE");
Scene scene = new Scene(layout, 595, 200);
stage.setTitle("Menu Button"); stage.setScene(scene);
stage.show();
}
public static void main(String args[])
{
launch(args);
}
}
```

OUTPUT


Ramya - bus/src/application/MenuButtonExample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
bus
src
application
Main.java
MenuButtonExample.java
application.css
module-info.java
JRE System Library [JavaSE-22]
JavaFX SDK
Referenced Libraries
build.fxbuild
MenuButtonExample.java
package application;
import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.MenuButton;
import javafx.scene.control.MenuItem;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
public class MenuButtonExample extends Application
{
public void start(Stage stage)
{
MenuButton menu=new MenuButton("Technology");
menu.setMnemonicParsing(true);
MenuItem item1=new MenuItem("Java");
MenuItem item2=new MenuItem("Python");
MenuItem item3=new MenuItem("C++");
MenuItem item4=new MenuItem("Big Data");
MenuItem item5=new MenuItem("Machine Learning
menu.getItems().addAll(item1,item2,item3,item
HBox layout=new HBox(25);
layout.getChildren().addAll(menu);
layout.setPadding(new Insets(15,50,50,150));
layout.setStyle("-fx-background-color:BEIGE");
Scene scene=new Scene(layout,595,200);
stage.setTitle("Menu Button");
stage.setScene(scene);
Menu Button
Technology
Task List
Find
All
Activate...
Problems
Javadoc
Declaration
0 errors, 1 warning, 0 others
Description
Resource
Path
Location
Type
Warnings (1 item)
application.MenuButtonExample.java - bus/src
Type here to search
29°C
ENG
12:00
05-11-2024


Ramya - bus/src/application/MenuButtonExample.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
bus
src
application
Main.java
MenuButtonExample.java
application.css
module-info.java
JRE System Library [JavaSE-22]
JavaFX SDK
Referenced Libraries
build.fxbuild
MenuButtonExample.java
package application;
import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.MenuButton;
import javafx.scene.control.MenuItem;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;
public class MenuButtonExample extends Application
{
public void start(Stage stage)
{
MenuButton menu=new MenuButton("Technology");
menu.setMnemonicParsing(true);
MenuItem item1=new MenuItem("Java");
MenuItem item2=new MenuItem("Python");
MenuItem item3=new MenuItem("C++");
MenuItem item4=new MenuItem("Big Data");
MenuItem item5=new MenuItem("Machine Learning
menu.getItems().addAll(item1,item2,item3,item
HBox layout=new HBox(25);
layout.getChildren().addAll(menu);
layout.setPadding(new Insets(15,50,50,150));
layout.setStyle("-fx-background-color:BEIGE");
Scene scene=new Scene(layout,595,200);
stage.setTitle("Menu Button");
stage.setScene(scene);
Menu Button
Technology
Java
Python
C++
Big Data
Machine Learning
Task List
Find
All
Activate...
Problems
Javadoc
Declaration
0 errors, 1 warning, 0 others
Description
Resource
Path
Location
Type
Warnings (1 item)
application.MenuButtonExample.java - bus/src
Type here to search
29°C
ENG
12:01
05-11-2024

Result:

Thus the application using JavaFX controls, layouts and menus was developed and the output was verified