SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Ramapuram Campus, Bharathi Salai, Ramapuram, Chennai - 600089

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TUTORIAL MANUAL

DEGREE / BRANCH: B.TECH/CSE

III SEMESTER

21CSC203P – ADVANCED PROGRAMMING PRACTICE

Regulation – 2021

Academic Year 2023-2024(ODD SEM)

LIST OF EXPERIMENTS:

S.NO	EXPERIMENT NAME
1.	Simple Java program using control structures
2.	Implement Sum of series (1 + 2+ 3+…..n,1+1/2+1/3 +……..1/n,12 + 22+ 32 +…….n2) using Java
3.	Simple Java Program to implement functions
4.	Simple Java Program with Class and Objects
5.	Implement Java program - Simple Calculator using polymorphism
6.	Implement Java program - Pay Slip Generator using Inheritance
7.	Simple Java Program to implement threads
8.	Simple Java Programs with Java Data Base Connectivity (JDBC)
9.	Form Design with applet and Swing using Java
10.	Simple Python Program to implement functions
11.	Python program using control structures and arrays
12.	Implement Python program - TCP/UDP program using Sockets
13.	Construct NFA and DFA using Python
14.	Implement a Python program for the algebraic manipulations using symbolic paradigm
15.	Simple Python programs to implement event handling

Experiment No: 1

SIMPLE JAVA PROGRAM USING CONTROL STRUCTURES

Aim:

To develop a Java program to validate age using control Structures.

Algorithm:

Step 1: Import Scanner class to get the inputs from the user
Step 2: Create a Class AgeValidator and create a main function
Step 3: Use Scanner to get age as the input from the user
Step 4: Use if Condition to check the age of the User
Step 5: Print the results based on the age

Program:

```
import java.util.Scanner;

class AgeValidator {

   public static void main(String[] args) {

      Scanner sc = new Scanner(System.in);

            int age = sc.nextInt();

           if(age >= 18) {

         System.out.println("You are Eligible for Voting");

      } else {

         System.out.println("You are Not Eligible for Voting");

      }

   }

}
```

Output:

```
java -cp /tmp/qpe1JeX9Wm HelloWorld
Enter your Age  : 34
You are Eligible for Voting
```

Result:

Thus, the Java Program with Control Structures is implemented and executed.

Experiment No: 2

IMPLEMENT SUM OF SERIES (1 + 2+ 3+…..n,1+1/2+1/3 +……..1/n,12 + 22+ 32 +…….n2) USING JAVA

Aim:

To implement a Java Program to Calculate Sum of series of (1 + 2+ 3+…..n,1+1/2+1/3 +……..1/n, 1^2+ 2^2+ 3^2 +…….n^2)

Algorithm:

Step 1: Import Scanner class to get the inputs from the user
Step 2: Create a Class Sum of Series and create a main function
Step 3: Use Scanner to get N as the input from the user
Step 4: Calculate the Sum of 1 + 2+ 3+…..n, and display the result
Step 5: Calculate the Sum of 1+1/2+1/3 +……..1/n, and display the results
Step 6: Calculate the Sum of , 1^2+ 2^2+ 3^2 +…….n^2, and display the results
Step 7: End

Program:

```
import java.util.Scanner;

class SumOfSeries {

   public static void main(String[] args) {

      Scanner sc = new Scanner(System.in);

      System.out.print("Enter the Value of N  : ");

      int n = sc.nextInt();

       int firstSum = 0;

      for(int i = 1; i <= n; i++) {

         firstSum += i;

      }

       System.out.println("Sum of First n Natural Numbers are : " + firstSum);

            double secondSum = (double) 0;

      for(int i= 1; i <= n; i++) {

         secondSum = (double)1/i;

      }

            System.out.println("Sum of Series 1 + 1/2 + 1/3 .. + 1/n : " + secondSum );

            int thirdSum = 0;
```

```
        for(int i = 1; i <= n;  i++)  {
            thirdSum += Math.pow(i, 2);
        }
            System.out.println("Sum of Series  1^2 + 2^2 + .. + n^2 : " + thirdSum);
            }
}
```

Output:

```
java -cp /tmp/qpe1JeX9Wm HelloWorld
Enter the Value of N  : 7
Sum of First n Natural Numbers are : 28
Sum of Series 1 + 1/2 + 1/3 .. + 1/n : 0.14285714285714285
Sum of Series  1^2 + 2^2 + .. + n^2 : 140
```

Result:

The Java Program to calculate the Sum of Series is implemented and executed.

Experiment No: 3

SIMPLE JAVA PROGRAMS TO IMPLEMENT FUNCTIONS

Aim:

To implement a Simple Java Program to perform addition using Functions.

Algorithm:

Step 1: Import Scanner class to get the inputs from the user
Step 2: Create a Class Function and create a main function
Step 3: Use Scanner to get two numbers as the input from the user
Step 4: Create a static function add to calculate sum of two number
Step 5: Return the result
Step 6: Print the result
Step 7: End

Program:

```
import java.util.Scanner;

class Function {

   public static int add(int a, int b) {

      return a+b;    }

   public static void main(String[] args) {

      Scanner sc = new Scanner(System.in);

      System.out.print("Enter the Value of two number that you want to add  : ");

      int a = sc.nextInt();

      int b = sc.nextInt();

      int sum = add(a,b);

     System.out.println("Sum of " + a + " and " + b + " are : " + sum);

   }

}
```

Output:

```
Enter the Value of two number that you want to add  : 2 3
Sum of 2 and 3 are : 5
```

Result:

The Java Program to perform addition using functions is implemented and executed.

Experiment No: 4

SIMPLE JAVA PROGRAMS WITH CLASS AND OBJECTS

Aim:

To implement a Simple Java Program with Classes and Objects.

Algorithm:

Step 1: Create a Main class and a main Function
Step 2: Declare a new Student class
Step 3: Student Class will contain two Attribute name, studentId
Step 4: Create a constructor function inside the student class to initialize the name, studentId
Step 5: Create student object inside the main function of Main class
Step 6: Print name and studentId of the objects
Step 7: End

Program:

```
class Student {
   String name;
   int studentId;
      public Student(String name, int studentId) {
      this.name = name;
      this.studentId = studentId;
   }
}
class Main {
   public static void main(String[] args) {
      Student s1 = new Student("John", 123);
      Student s2 = new Student("Jack", 124);

      System.out.println("Name of Student1 is : " + s1.name + "\n");
      System.out.println("StudentId of Student1 is : " + s1.studentId + " \n");

      System.out.println("Name of Student2 is : " + s2.name + "\n");
      System.out.println("StudentId of Student2 is : " + s2.studentId);
```

```
    }
}
```

Output:

```
Name of Student1 is : John

StudentId of Student1 is : 123

Name of Student2 is : Jack
StudentId of Student2 is : 124
```

Result:

Thus, the Java Program is implemented and executed using with class and objects.

Experiment No: 5

IMPLEMENT JAVA PROGRAM - SIMPLE CALCULATOR USING POLYMORPHISM

Aim:

To implement a simple Calculator using Polymorphism in Java.

Algorithm:

Step1: Create a Main class and main function
Step 2: Create a class Addition with a method add
Step 3: Add method need to be overloaded with 1, 2, 3 arguments
Step 4: Create an object for Addition class
Step 5: Print the result of the all add methods
Step 6: End

Program:

```
class Addition {
   public int add(int a) {
      return a + 1;
   }
      public int add(int a, int b) {
      return a + b;
   }
   public int add(int a, int b, int c) {
      return a + b + c;
   }
}
class Main{
   public static void main(String[] args) {
            Addition obj = new Addition();
            int sum1 = obj.add(5);
      System.out.println("Result of 1st Method :  "  + sum1);
            int sum2 = obj.add(2, 3);
      System.out.println("Result of 2nd Method :  "  + sum2 );
```

```
            int sum3 = obj.add(2,2,3);
        System.out.println("Result of 3rd Method :  " + sum3);
    }
}
```

Output:

```
Result of 1st Method :  6
Result of 2nd Method :  5

Result of 3rd Method :  7
```

Result:

Thus, the Java Program to create a simple calculator with polymorphism is implemented and executed.

Experiment No: 6

JAVA PROGRAM TO GENERATE PAYSLIP USING INHERITANCE

Aim:

To develop a java application to generate pay slip for different category of employees using the concept of inheritance.

Algorithm:

Step 1: Create the class employee with name, Empid, address, mailid, mobileno as members.
Step 2: Inherit the classes programmer, asstprofessor, associateprofessor and professor from employee class.
Step 3: Add Basic Pay (BP) as the member of all the inherited classes.
Step 4: Calculate DA as 97% of BP, HRA as 10% of BP, PF as 12% of BP, Staff club fund as 0.1% of BP.
Step 5: Calculate gross salary and net salary.
Step 6: Generate payslip for all categories of employees.
Step 7: Create the objects for the inherited classes and invoke the necessary methods to display the Payslip.

Program:

```
import java.util.*;

class employee

{

 int empid;

 long mobile;

 String name, address, email;

 Scanner get = new Scanner(System.in);


void getdata()

{

 System.out.print("Enter Name of the Employee: ");

 name = get.nextLine();

 System.out.print("Enter Employee E-mail id: ");

 email = get.nextLine();

 System.out.print("Enter Address of the Employee: ");
```

```
 address = get.nextLine();
 System.out.print("Enter Employee id: ");
 empid = get.nextInt();
 System.out.print("Enter Mobile Number: ");
 mobile = get.nextLong();
 }

 void display()
 {
 System.out.println("\n");
 System.out.println("**************************************************");
 System.out.println("Employee Information");
 System.out.println("**************************************************");
 System.out.println("Employee Name: "+name);
 System.out.println("Employee id: "+empid);
 System.out.println("Mail id: "+email);
 System.out.println("Address: "+address);
 System.out.println("Mobile Number: "+mobile);
 }
}
class developer extends employee
{
double salary, bp, da, hra, pf, ta, net, gross;
void getdeveloper()
{
System.out.print("Enter Your Basic pay: ");
bp = get.nextDouble();
}
void calculatedev()
{
da=(0.15*bp);
```

```
hra=(0.20*bp);
pf=(0.12*bp);
ta=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-ta);
System.out.println("\n");
System.out.println("***********************************************");
System.out.println("PAY SLIP FOR Developer");
System.out.println("***********************************************");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("PF:Rs"+pf);
System.out.println("HRA:Rs"+hra);
System.out.println("ta:Rs"+ta);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}
class manager extends employee
{
double salary, bp, da, hra, pf, ta, net, gross;
void getmanage()
{
System.out.print("Enter Your Basic pay: ");
bp = get.nextDouble();
}
void calculatemng()
{
da=(0.15*bp);
hra=(0.20*bp);
pf=(0.12*bp);
```

```
ta=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-ta);
System.out.println("\n");
System.out.println("*********************************************");
System.out.println("PAY SLIP FOR Manager");
System.out.println("*********************************************");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("ta:Rs"+ta);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
 }
}
class designer extends employee
{
double salary, bp, da, hra, pf, ta, net, gross;
 void getdesign()
{
System.out.print("Enter Your Basic pay: ");
bp = get.nextDouble();
}
void calculatedesign()
{
da=(0.15*bp);
hra=(0.20*bp);
pf=(0.12*bp);
ta=(0.1*bp);
gross=(bp+da+hra);
```

```
net=(gross-pf-ta);
System.out.println("\n");
System.out.println("**********************************************");
System.out.println("PAY SLIP FOR Designer");
System.out.println("**********************************************");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("ta:Rs"+ta);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}
class accountant extends employee
{
double salary, bp, da, hra, pf, ta, net, gross;
void getaccountant()
{
System.out.print("Enter Your Basic pay: ");
bp = get.nextDouble();
}
void calculateaccountant()
{
da=(0.15*bp);
hra=(0.20*bp);
pf=(0.12*bp);
ta=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-ta);
System.out.println("\n");
```

```
System.out.println("**********************************************");
System.out.println("PAY SLIP FOR Accountant");
System.out.println("**********************************************");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("ta:Rs"+ta);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}
class Main
{
public static void main(String args[])
{
int choice,cont;
do
{
System.out.println("\n");
System.out.println("***** Welcome to Webeduclick Payroll Management System *****");
System.out.println(" 1. Developer \t 2. Manager \t 3. Designer \t 4. Accountant");
Scanner c = new Scanner(System.in);
choice=c.nextInt();
switch(choice)
{
case 1:
{
developer dev=new developer();
dev.getdata();
dev.getdeveloper();
```

```
dev.display();
dev.calculatedev();
break;
}
case 2:
{
manager mng=new manager();
mng.getdata();
mng.getmanage();
mng.display();
mng.calculatemng();
break;
}
case 3:
{
designer design=new designer();
design.getdata();
design.getdesign();
design.display();
design.calculatedesign();
break;
}
case 4:
{
accountant acc=new accountant();
acc.getdata();
acc.getaccountant();
acc.display();
acc.calculateaccountant();
break;
}
```

```
}
System.out.println("\n");
System.out.println("Do you want to quit press 0 to quit and press 1 to continue");
cont=c.nextInt();
}while(cont==1);
}
}
```

Output:

```
***** Welcome to Webeduclick Payroll Management System *****
 1. Developer    2. Manager      3. Designer     4. Accountant
1
Enter Name of the Employee: ABC
Enter Employee E-mail id: ABS@xyz.com
Enter Address of the Employee: Chennai,TN
Enter Employee id: 31
Enter Mobile Number: 1234567890
Enter Your Basic pay: 70000


***********************************************
Employee Information
***********************************************
Employee Name: ABC
Employee id: 31
Mail id: ABS@xyz.com
Address: Chennai,TN
Mobile Number: 1234567890


***********************************************
PAY SLIP FOR Developer
***********************************************
Basic Pay:Rs70000.0
DA:Rs10500.0
PF:Rs8400.0
HRA:Rs14000.0
ta:Rs7000.0
GROSS PAY:Rs94500.0
NET PAY:Rs79100.0


Do you want to quit press 0 to quit  and  press 1 to continue
```

Result:

Thus, the java application to generate pay slips for different categories of employees was implemented using inheritance and the program was executed successfully.

Experiment No: 7

SIMPLE JAVA PROGRAMS TO IMPLEMENT THREAD

Aim:

To demonstrate the concept of threading in Java and showcase how concurrent execution of tasks can be achieved using threads by simulating a cooking scenario.

Algorithm:

Step 1: Define an inner class CookThread inside the CookingExperiment class, extending Thread.
Step 2: Inside the CookThread class declare a private String variable name to represent the person's name.
Step 3: Create a constructor that takes the name as a parameter and assigns it to the name variable.
Step 4: Override the run() method inherited from the Thread class.
Step 5: Print a message indicating that the person has started cooking.

Program:

```
public class Main {
    public static void main(String[] args) {
        Thread person1 = new CookThread("Person 1");
        Thread person2 = new CookThread("Person 2");
        person1.start();
        person2.start();
    }
    static class CookThread extends Thread {
        private final String name;
        public CookThread(String name) {
            this.name = name;
        }
        public void run() {
            System.out.println(name + " started cooking.");
            for (int i = 1; i <= 5; i++) {
                System.out.println(name + " - Preparing ingredient " + i);
                try {
                    Thread.sleep(100);
                } catch (InterruptedException e) {
```

```
                e.printStackTrace();
            }
        }

        System.out.println(name + " finished cooking.");
    }
  }
}
```

Output:

```
Person 1 started cooking.
Person 2 started cooking.
Person 1 - Preparing ingredient 1
Person 2 - Preparing ingredient 1
Person 1 - Preparing ingredient 2
Person 2 - Preparing ingredient 2
Person 1 - Preparing ingredient 3
Person 2 - Preparing ingredient 3
Person 1 - Preparing ingredient 4
Person 2 - Preparing ingredient 4
Person 1 - Preparing ingredient 5
Person 2 - Preparing ingredient 5
Person 1 finished cooking.
Person 2 finished cooking.


...Program finished with exit code 0
Press ENTER to exit console.
```

Result:

Thus, the concept of threading in Java is executed successfully showcasing the concurrent execution of tasks using threads.

Experiment No: 8

SIMPLE JAVA PROGRAM WITH JAVA DATA BASSE CONNECTIVITY (JDBC)

Aim:

To create a Java program with Java Database Connectivity (JDBC) to manage a student database, allowing users to perform operations like creating and inserting student records to the database.

Algorithm:

Step 1: Database credentials
Step 2: Load and register the JDBC driver
Step 3: Create a connection to the database
Step 4: Create a statement
Step 5: Create the table "students" if it doesn't exist
Step 6: Insert a new student record
Step 7: Retrieve student records
Step 8: Process the result set
Step 9: Close the resources

Program:

```
import java.sql.*;

public class StudentDatabaseExample {

    public static void main(String[] args) {

        String dbUrl = "jdbc:mysql://localhost:3306/university";

        String dbUsername = "your_username";

        String dbPassword = "your_password";

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");

                    Connection connection = DriverManager.getConnection(dbUrl, dbUsername,
            dbPassword);

            Statement statement = connection.createStatement();

            String createTableQuery = "CREATE TABLE IF NOT EXISTS students (" +

                "id INT AUTO_INCREMENT PRIMARY KEY," +

                "name VARCHAR(100) NOT NULL," +

                "age INT NOT NULL," +
```

```
            "major VARCHAR(100) NOT NULL" +
            ")";
    statement.executeUpdate(createTableQuery);
    System.out.println("Table 'students' created successfully.");
            String insertQuery = "INSERT INTO students (name, age, major) VALUES
    ('John Doe', 20, 'Computer Science')";
    int rowsAffected = statement.executeUpdate(insertQuery);
    if (rowsAffected > 0) {
        System.out.println("New student record added successfully!");
    } else {
        System.out.println("Failed to add a new student record.");
    }
    String selectQuery = "SELECT id, name, age, major FROM students";
    ResultSet resultSet = statement.executeQuery(selectQuery);
    System.out.println("\nStudent Records:");
    while (resultSet.next()) {
        int id = resultSet.getInt("id");
        String name = resultSet.getString("name");
        int age = resultSet.getInt("age");
        String major = resultSet.getString("major");


        System.out.println("ID: " + id + ", Name: " + name + ", Age: " + age + ", Major: " +
    major);
    }
    resultSet.close();
    statement.close();
    connection.close();
} catch (ClassNotFoundException e) {
            System.err.println("Failed to load JDBC driver. Make sure you have added the
    JDBC library to the classpath.");
    e.printStackTrace();
} catch (SQLException e) {
```

```
            System.err.println("Failed to connect to the database or execute the query.");
            e.printStackTrace();
        }
    }
}
```

Output:

```
Table 'students' created successfully.
New student record added successfully!

Student Records:
ID: 1, Name: John Doe, Age: 20, Major: Computer Science
```

Result:

Thus, a Java program with Java Database Connectivity (JDBC) to manage a student database, allowing users to perform operations like creating and inserting student records to the database has been successfully executed and verified.

Experiment No: 9

FORM DESIGN WITH APPLET AND SWING USING JAVA

Aim:

To create a Java application using AWT Applet and Swing components to design a user-friendly data input form with interactive features.

Algorithm:

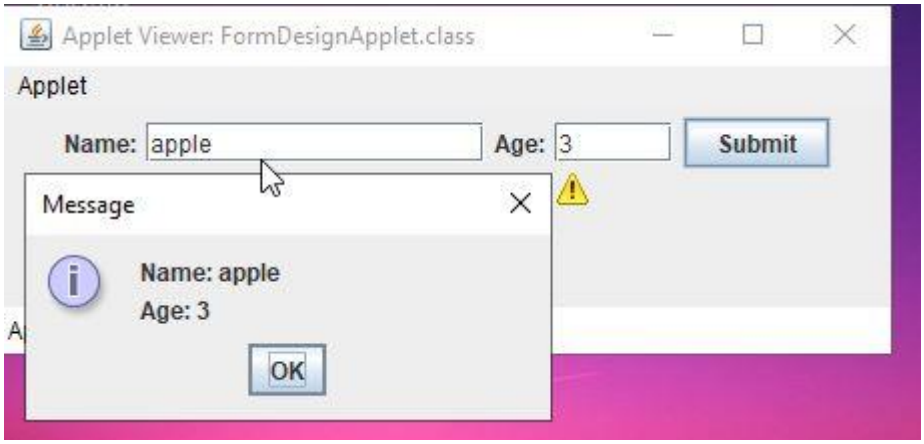
1. Set the layout for the Applet
2. Create Swing components for the form
3. Add components to the Applet
4. Action listener for the submit button
5. Perform any processing or validation with the submitted data here
6. For this example, we will just display the entered data in a dialog box

Program:

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class FormDesignApplet extends JApplet {

    public void init() {

        setLayout(new FlowLayout());

        JLabel nameLabel = new JLabel("Name:");

        JTextField nameTextField = new JTextField(15);

        JLabel ageLabel = new JLabel("Age:");

        JTextField ageTextField = new JTextField(5);

        JButton submitButton = new JButton("Submit");

        add(nameLabel);

        add(nameTextField);

        add(ageLabel);

        add(ageTextField);

        add(submitButton);
```

```
        submitButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String name = nameTextField.getText();
                String age = ageTextField.getText();
                        JOptionPane.showMessageDialog(null, "Name: " + name + "\nAge: " +
age);
            }
        });
    }
}
```

Output:



Result:

Thus, a Java application using AWT Applet and Swing components to design a user-friendly data input form with interactive features has been successfully executed and verified.

Experiment No: 10

SIMPLE PYTHON PROGRAM TO IMPLEMENT FUNCTIONS

Aim:

To implement functions in python to print the value of a decimal number in binary, hexadecimal, octal format.

Algorithm:

Step 1: Define three functions: decimal_into_binary(decimal_1), decimal_into_octal(decimal_1) decimal_into_hexadecimal(decimal_1)
Step 2: Input a decimal number from the user using the input() function. The input is converted to an integer using int() and stored in the variable decimal_1.
Step 3: Call the three defined functions with the input decimal number decimal_1 as arguments.
Step 4: The functions will convert the decimal number into its respective binary, octal, and hexadecimal representations and print the results

Program:

```
def decimal_into_binary(decimal_1):
    decimal = int(decimal_1)
    print ("The given decimal number", decimal, "in Binary number is: ", bin(decimal))

def decimal_into_octal(decimal_1):
    decimal = int(decimal_1)
    print ("The given decimal number", decimal, "in Octal number is: ", oct(decimal))

def decimal_into_hexadecimal(decimal_1):
    decimal = int(decimal_1)
    print ("The given decimal number", decimal, "in Hexadecimal number is: ", hex(decimal))

decimal_1 = int (input (" Enter the Decimal Number: "))
decimal_into_binary(decimal_1)
decimal_into_octal(decimal_1)
decimal_into_hexadecimal(decimal_1)
```

Output:

```
 Enter the Decimal Number: 5
The given decimal number 5 in Binary number is:  0b101
The given decimal number 5 in Octal number is:  0o5
The given decimal number 5 in Hexadecimal number is:  0x5


...Program finished with exit code 0
Press ENTER to exit console.
```

Result:

Thus, the functions are implemented successfully to print the value of a decimal number in binary, hexadecimal, octal format.

Experiment No: 11

PYTHON PROGRAM USING CONTROL STRUCTURES AND ARRAYS

Aim:

To write a Python program using control structures and arrays.

Control Structures:

i) if else

Algorithm:

Step 1: Start the program
Step 2: Get the three numbers
Step 3: Find the greatest of three numbers using if else by comparing each number with the other
Step 4: Display the greatest number
Step 5: Stop

Program:

```
# Python program to find the largest number among the three input numbers
# change the values of num1, num2 and num3
# for a different result

num1 = 10
num2 = 14
num3 = 12

# uncomment following lines to take three numbers from user

#num1 = float(input("Enter first number: "))
#num2 = float(input("Enter second number: "))
#num3 = float(input("Enter third number: "))

if (num1 >= num2) and (num1 >= num3):
   largest = num1
elif (num2 >= num1) and (num2 >= num3):
   largest = num2

else:
   largest = num3

print("The largest number is", largest)
```

Output:

```
The largest number is 14.0
```

ii) while

Algorithm:

Step 1: Start the program
Step 2: Get the input number
Step 3: Find the sum of all numbers until the user enters zero
Step 4: Display the sum of all the numbers once 0 is entered.
Step 5: Stop

Program:

```
# program to calculate the sum of numbers
# until the user enters zero
total = 0
number = int(input('Enter a number: '))

# add numbers until number is zero
while number != 0:
    total += number    # total = total + number
        # take integer input again
    number = int(input('Enter a number: '))

print('total =', total)
```

Output:

```
Enter a number: 34
Enter a number: 45
Enter a number: 67
Enter a number: 5
Enter a number: 98
Enter a number: 0
total = 249
```

iii) for

Algorithm:

Step 1: Start the program
Step 2: Declare the list
Step 3: For loop continues until we reach the last item in the sequence
Step 4: Stop

Program:

```
for x in 'Python':

    print(x)
```

Output:

P

y

t

h

o

n

Arrays:

Algorithm:

Step 1: Start the program
Step 2: Import array for array creation
Step 3: Create array with a data type
Step 4: Create array with another data type
Step 5: Array store multiple items of the same type together
Step 6: Stop

Program:

```
# Python program to demonstrate
# Creation of Array
# importing "array" for array creations
import array as arr

# creating an array with integer type
a = arr.array('i', [1, 2, 3])
# printing original array
print("The new created array is : ", end=" ")
for i in range(0, 3):
        print(a[i], end=" ")
print()

# creating an array with double type
b = arr.array('d', [2.5, 3.2, 3.3])

# printing original array
print("\nThe new created array is : ", end=" ")
for i in range(0, 3):
        print(b[i], end=" ")
```

Output:

The new created array is : 1 2 3

The new created array is : >
2.5 3.2 3.3

Result :

Thus, Python program using control structures and arrays was executed and output was taken.

Experiment No: 12

IMPLEMENT PYTHON PROGRAM - TCP/UDP PROGRAM USING SOCKETS

Aim:

To implement Python program - TCP/UDP program using Sockets.

Algorithm:

Step 1: The first step is to create a socket and use the socket() function to create a socket.
Step 2: Use the connect() function for connecting the socket to the server address.
Step 3: Transmit data between two communicating parties using read() and write() functions.

Program:

UDP Server using Python:

```
import socket

localIP     = "127.0.0.1"
localPort   = 20001
bufferSize  = 1024

msgFromServer       = "Hello UDP Client"
bytesToSend         = str.encode(msgFromServer)

# Create a datagram socket
UDPServerSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)

# Bind to address and ip
UDPServerSocket.bind((localIP, localPort))

print("UDP server up and listening")

# Listen for incoming datagrams
while(True):
    bytesAddressPair = UDPServerSocket.recvfrom(bufferSize)
    message = bytesAddressPair[0]
    address = bytesAddressPair[1]
    clientMsg = "Message from Client:{}".format(message)
    clientIP  = "Client IP Address:{}".format(address)

    print(clientMsg)
    print(clientIP)

    # Sending a reply to client
    UDPServerSocket.sendto(bytesToSend, address)
```

Output:

UDP server up and listening

Message from Client:b"Hello UDP Server"

Client IP Address:("127.0.0.1", 51696)

UDP Client using Python:

```
import socket
msgFromClient       = "Hello UDP Server"
bytesToSend         = str.encode(msgFromClient)
serverAddressPort   = ("127.0.0.1", 20001)
bufferSize          = 1024

# Create a UDP socket at client side
UDPClientSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)

# Send to server using created UDP socket
UDPClientSocket.sendto(bytesToSend, serverAddressPort)

msgFromServer = UDPClientSocket.recvfrom(bufferSize)

msg = "Message from Server {}".format(msgFromServer[0])

print(msg)
```

Output:

Message from Server b"Hello UDP Client"

Result:
Thus TCP/UDP program using Sockets was implemented and executed using Python.

Experiment No:13

CONSTRUCT NFA AND DFA USING PYTHON

Aim:

To implement symbolic programming paradigm in python.

13a. To write a program to convert DFA to NFA using python.

Algorithm:
Step 1: Initialize the transitions
Step 2: Copy the input in list
Step 3: Parse the string of a,b in 0,1 for simplicity
Step 4: Counter to remember the number of symbols read
Step 5: Set the final states
Step 6: Check for each possibility
Step 7: Move further only if you are at non-hypothetical state
Step 8: Read the last symbol and current state lies in the set of final states
Step 9: Input string for next transition is input[i+1:]
Step 10: Increment the counter
Step 11: Print the state

Program:

```
import sys

def main():
    transition = [[[0,1],[0]], [[4],[2]], [[4],[3]], [[4],[4]]]
    input = raw_input("enter the string: ")
    input = list(input)
    for index in range(len(input)):
        if input[index]=='a':
            input[index]='0'
        else:
            input[index]='1'
    final = "3"
    i=0
    trans(transition, input, final, start, i)
    print "rejected"
def trans(transition, input, final, state, i):
    for j in range (len(input)):
        for each in transition[state][int(input[j])]:
            if each < 4:
```

```
            state = each
            if j == len(input)-1 and (str(state) in final):
                print "accepted"
                sys.exit()
            trans(transition, input[i+1:], final, state, i)
        i = i+1
main()
```

Output:

enter the string: abb
accepted

enter the string: aaaabbbb
rejected

13b.Write a program to convert NFA to DFA

Algorithm:

Step 1: Take NFA input from User
Step 2: Creating a nested dictionary
Step 3: Assign the end states to the paths in dictionary\
Step 4: Print NFA
Step 5: Enter final state/states of NFA
Step 6: Create a nested dictionary in dfa
Step 7: Create a single string from all the elements of the list
Step 8: Compute the other rows of DFA transition table
Step 9: Create a temporay list
Step 10: Assign the state in DFA table
Step 11: Create a single string(new state) from all the elements of the list
Step 12: Assign the new state in the DFA table
Step 13: Remove the first element in the new_states_list
Step 14: Print the DFA created
Step 15: Step 1: Print Final states of DFA

Program:

```
import pandas as pd

nfa = {}

n = int(input("No. of states : "))

t = int(input("No. of transitions : "))

for i in range(n):

    state = input("state name : ")

    nfa[state] = {}
```

```
    for j in range(t):
        path = input("path : ")
        print("Enter end state from state {} travelling
        through path {} : ".format(state,path))
        reaching_state = [x for x in input().split()]
        nfa[state][path] = reaching_state
print("\nNFA :- \n")
print(nfa)
print("\nPrinting NFA table :- ")
nfa_table = pd.DataFrame(nfa)
print(nfa_table.transpose())
print("Enter final state of NFA : ")
nfa_final_state = [x for x in input().split()]
new_states_list = []
dfa = {}
keys_list = list(list(nfa.keys())[0])
path_list = list(nfa[keys_list[0]].keys())
dfa[keys_list[0]] = {}
for y in range(t):
    var = "".join(nfa[keys_list[0]][path_list[y]])
    dfa[keys_list[0]][path_list[y]] = var
    if var not in keys_list:
        new_states_list.append(var)
        keys_list.append(var)
while len(new_states_list) != 0:
    dfa[new_states_list[0]] = {}
    for _ in range(len(new_states_list[0])):
        for i in range(len(path_list)):
            temp = []
            for j in range(len(new_states_list[0])):
                temp += nfa[new_states_list[0][j]][path_list[i]]
            s = ""
```

```
            s = s.join(temp)
            if s not in keys_list:
                new_states_list.append(s)
                keys_list.append(s)
            dfa[new_states_list[0]][path_list[i]] = s
    new_states_list.remove(new_states_list[0])
print("\nDFA :- \n")
print(dfa)
print("\nPrinting DFA table :- ")
dfa_table = pd.DataFrame(dfa)
print(dfa_table.transpose())
dfa_states_list = list(dfa.keys())
dfa_final_states = []
for x in dfa_states_list:
    for i in x:
        if i in nfa_final_state:
            dfa_final_states.append(x)
            break
  print("\nFinal states of the DFA are : ",dfa_final_states)
```

Output:

```
 No. of states : 4
No. of transitions : 2
state name : A
path : a
Enter end state from state A travelling through path a :
A B
path : b
Enter end state from state A travelling through path b :
A
state name : B
path : a
```

```
Enter end state from state B travelling through path a :
C
path : b
Enter end state from state B travelling through path b :
C
state name : C
path : a
Enter end state from state C travelling through path a :
D
path : b
Enter end state from state C travelling through path b :
D
state name : D
path : a
Enter end state from state D travelling through path a :
path : b
Enter end state from state D travelling through path b :
NFA :-
{'A': {'a': ['A', 'B'], 'b': ['A']}, 'B': {'a': ['C'], 'b':
['C']}, 'C': {'a': ['D'], 'b': ['D']}, 'D': {'a': [], 'b': []}}
Printing NFA table :-
        a    b
A  [A, B]  [A]
B     [C]  [C]
C     [D]  [D]
D      []   []
Enter final state of NFA :
D
DFA :-
{'A': {'a': 'AB', 'b': 'A'}, 'AB': {'a': 'ABC', 'b': 'AC'},
'ABC': {'a': 'ABCD', 'b': 'ACD'}, 'AC': {'a': 'ABD', 'b':
```

'AD'}, 'ABCD': {'a': 'ABCD', 'b': 'ACD'}, 'ACD': {'a': 'ABD',

'b': 'AD'}, 'ABD': {'a': 'ABC', 'b': 'AC'}, 'AD': {'a': 'AB',

'b': 'A'}}

Printing DFA table :-

	a	b
A	AB	A
AB	ABC	AC
ABC	ABCD	ACD
AC	ABD	AD
ABCD	ABCD	ACD
ACD	ABD	AD
ABD	ABC	AC
AD	AB	A

Final states of the DFA are : ['ABCD', 'ACD', 'ABD', 'AD']

Result:

Thus, the Python program to implement the conversion of DFA to NFA and NFA to DFA have been executed successfully.

Experiment No: 14

IMPLEMENT A PYTHON PROGRAM FOR THE ALGEBRAIC MANIPULATIONS USING SYMBOLIC PARADIGM

Aim:

To implement symbolic programming paradigm in python for algebraic manipulations.

14 a. Write the commands to perform the operations on substitutions and expressions

Algorithm:

Step 1: Import sympy module

Step 2: Evaluate the expression using sympy command

Step 3: Print the result

Program:

```
from sympy import *
expr = cos(x) + 1
print( expr.subs(x, y))
x, y, z = symbols("x y z")
 print(expr = x**y)
print(expr)
expr = sin(2*x) + cos(2*x)
print( expand_trig(expr))
print(expr.subs(sin(2*x), 2*sin(x)*cos(x)))
expr = x**4 - 4*x**3 + 4*x**2 - 2*x + 3
 replacements = [(x**i, y**i) for i in range(5) if i % 2 == 0]
print( expr.subs(replacements))
str_expr = "x**2 + 3*x - 1/2"
 expr = sympify(str_expr)
expr
 expr.subs(x, 2)
expr = sqrt(8)
print( expr.evalf())
expr = cos(2*x)
 expr.evalf(subs={x: 2.4})
```

```
one = cos(1)**2 + sin(1)**2
print( (one - 1).evalf())
```

Output:

cos(y)+1

Xxy

2sin(x)cos(x)+2cos2(x)−1

2sin(x)cos(x)+cos(2x)

4x3−2x+y4+4y2+3

192

2.82842712474619

0.0874989834394464

x2+3x−12

14 b. To perform the following operations on matrices

Algorithm:

Step 1: Import matrix from sympy.matrices.
Step 2: Create the matrix
Step 3: Print the matrix
Step 4: Display the matrix
Step 5: Display 0th row
Step 6: Print first column
Step 7: Delete the first column from the matrix
Step 8: Insert the row into the matrix
Step 9: Generate two matrices
Step 10: Print addition of two matrices
Step 11: Print the multiplication of two matrices

Program:

```
from sympy.matrices import Matrix
 m=Matrix([[1,2,3],[2,3,1]])
print( m)
M=Matrix(2,3,[10,40,30,2,6,9])
Print( M)
```

```
Print(M.shape)
Print(M.row(0))
M.col(1)
M.row(1)[1:3]
Print( M)
M=Matrix(2,3,[10,40,30,2,6,9])
 M.col_del(1)
a=Matrix([[1,2,3],[2,3,1]])
 print(a)
a1=Matrix([[10,30]])
 a=M.row_insert(0,M1)
print(a)
a2=Matrix([40,6])
 a=M.col_insert(1,M2)
 print(a)
 M1=Matrix([[1,2,3],[3,2,1]])
 M2=Matrix([[4,5,6],[6,5,4]])
Print( M1+M2)
M1=Matrix([[1,2,3],[3,2,1]])
 M2=Matrix([[4,5],[6,6],[5,4]])
Print( M1*M2)
```

Output:

```
[1 2 3 2 3 1]
[10 40 30 2 6 9]
(2,3)
[10 40 30]
 [40 6]
[6, 9]
[10 30 2 6 9]

Matrix([[10, 30],[ 2, 9]])

[1 2 3 2 3 1]
```

[10 40 30 2 9]

[10 40 30 6 9]

[5 7 9 9 7 5]

14 c. Write the commands to find derivative, integration, limits, quadratic equation

Algorithm:

Step 1: Import sympy module
Step 2: Make a symbol
Step 3: Find the derivative of the expression
Step 4: Print the result
Step 5: Find the integration of the expression
Step 6: Print the result
Step 7: Find the limit of the expression
Step 8: Print the result
Step 9: Find the quadratic equation of the expression
Step 10: Print the result

Program:

```
from sympy import *

x = Symbol('x')

 #make the derivative of cos(x)*e ^ x

ans1 = diff(cos(x)*exp(x), x)

print("The derivative of the  sin(x)*e ^ x : ", ans1)

# Compute (e ^ x * sin(x)+ e ^ x * cos(x))dx

ans2 = integrate(exp(x)*sin(x) + exp(x)*cos(x), x)

print("The result of  integration is : ", ans2)

# Compute definite integral of sin(x ^ 2)dx

# in b / w interval of ? and ?? .

ans3 = integrate(sin(x**2), (x, -oo, oo))

print("The value of integration is : ", ans3)

# Find the limit of sin(x) / x given x tends to 0

ans4 = limit(sin(x)/x, x, 0)

print("limit is : ", ans4)

# Solve quadratic equation like, example : x ^ 2?2 = 0
```

```
ans5 = solve(x**2 - 2, x)
print("roots are : ", ans5)
```

Output:

```
from sympy import *
x = Symbol('x')
 #make the derivative of cos(x)*e ^ x
ans1 = diff(cos(x)*exp(x), x)
print("The derivative of the  sin(x)*e ^ x : ", ans1)
# Compute (e ^ x * sin(x)+ e ^ x * cos(x))dx
ans2 = integrate(exp(x)*sin(x) + exp(x)*cos(x), x)
print("The result of  integration is : ", ans2)
# Compute definite integral of sin(x ^ 2)dx
# in b / w interval of ? and ?? .
ans3 = integrate(sin(x**2), (x, -oo, oo))
print("The value of integration is : ", ans3)
# Find the limit of sin(x) / x given x tends to 0
ans4 = limit(sin(x)/x, x, 0)
print("limit is : ", ans4)
# Solve quadratic equation like, example : x ^ 2?2 = 0
ans5 = solve(x**2 - 2, x)
print("roots are : ", ans5)
```

Result:

Thus, the Python program to implement symbolic program have been written and executed successfully.

Experiment No: 15

SIMPLE PYTHON PROGRAMS TO IMPLEMENT EVENT HANDLING

Aim:

To implement various kind of event handling programs in python using mouse and keyboard

15 a. Program to implement left click and right click events

Algorithm:

Step 1: Import the package tkinter
Step 2: Define the right click event
Step 3: Print the right click event detail.
Step 4: Define the left click event
Step 5: Print the left click event detail.
Step 6: Define the middle click event
Step 7: Print the middle click event detail.
Step 8: Binding the event with buttons
Step 9: Create object and run main loop.
Step 10: Stop

Program :

```
From tkinter import *
root = Tk()
def rightclick(ev):
print("rightclick")
def leftclick(ev):
print("leftclick")
def middleclick(event):
print("middleclick")
frame = Frame(root,width=300,height=200)
frame.bind("<Button-1>",leftclick)
frame.bind("<Button-2>",middleclick)
frame.bind("<Button-3>",rightclick)
frame.pack()
root.mainloop()
```

Output :

leftclick

rightclick

leftclick

rightclick

leftclick

leftclick

15 b.Program to implement mouse events

Algorithm :

Step 1: Import tkinter package
Step 2: Create class App
Step 3: Initialize the constructor
Step 4: Set the frame with green color
Step 5: Bind button, double button and button release events
Step 6: Bind motion ,enter and leave events
Step 7: Create object and run main loop.
Step 8: Stop

Program :

```
Import tkinter as tk
class App(tk.Tk):
def __init__(self):
super().__init__()
frame = tk.Frame(self, bg="green",  height=100, width=100)
frame.bind("<Button-1>", self.print_event)
frame.bind("<Double-Button-1>", self.print_event)
frame.bind("<ButtonRelease-1>", self.print_event)
frame.bind("<B1-Motion>", self.print_event)
frame.bind("<Enter>", self.print_event)
frame.bind("<Leave>", self.print_event)
frame.pack(padx=50, pady=50)
def print_event(self, event):
position = "(x={}, y={})".format(event.x, event.y)
print(event.type, "event", position)
if __name__ == "__main__":
app = App()
app.mainloop()
```

Output :

Enter event (x=86, y=1)

Leave event (x=46, y=100)

Enter event (x=48, y=95)

Leave event (x=7, y=100)

Enter event (x=50, y=98)

Leave event (x=47, y=103)

Enter event (x=56, y=95)

Leave event (x=115, y=122)

15c. Program to capture keyboard events

Algorithm:

Step 1: Import tkinter package

Step 2: Define Key press and click events

Step 3: Bind the keypress and button events

Step 4: Create object and run main loop.

Step 5: Stop

Program :

```
From Tkinter import *
root = Tk()
def key(event):
print"pressed", repr(event.char)
def callback(event):
frame.focus_set()
print"clicked at", event.x, event.y
frame = Frame(root, width=100, height=100)
frame.bind("<Key>", key)
frame.bind("<Button-1>", callback)
frame.pack()
root.mainloop()
```

Output :

```
clicked at 61 21
```

pressed 'w'

pressed 'w'

pressed 'b'

pressed 'b'

pressed 'b'

pressed 'b'

pressed 'b'

clicked at 47 54

clicked at 47 54

clicked at 47 54

15d. **Program to implement the keypress event**

Algorithm :

Step 1: Import tkinter package
Step 2: Define class App
Step 3: Define focus in and key events
Step 4: Bind the events
Step 5: Create object and run main loop.
Step 6: Stop

Program :

```
Import tkinter as tk
class App(tk.Tk):
def __init__(self):
super().__init__()
entry = tk.Entry(self)
entry.bind("<FocusIn>", self.print_type)
entry.bind("<Key>", self.print_key)
entry.pack(padx=20, pady=20)

def print_type(self, event):
print(event.type)
def print_key(self, event):
args = event.keysym, event.keycode, event.char
print("Symbol: {}, Code: {}, Char: {}".format(*args))
```

```
if __name__ == "__main__":
app = App()
app.mainloop()
```

Output :

```
FocusIn
Symbol: s, Code: 83, Char: s
Symbol: d, Code: 68, Char: d
Symbol: a, Code: 65, Char: a
Symbol: Caps_Lock, Code: 20, Char:
Symbol: Caps_Lock, Code: 20, Char:
Symbol: v, Code: 86, Char: v
Symbol: c, Code: 67, Char: c
```

Result:

Thus, the Python program to implement various mouse click and keyboard events have been executed and implemented successfully.