**CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY**

**CHANDUBHAI S PATEL INSTITUTE OF TECHNOLOGY**

**Name:- Patel Vraj**

**ID:- 21CE105**

**CSPIT -CE**

|  |  |
| --- | --- |
|  | **Practical -1** |
| **Practical 1.1** | Introduction to Object Oriented Concepts, comparison of Java with other object oriented programming languages. Introduction to JDK, JRE, JVM, javadoc, command line argument |
| **Ans:-** | OOP Concept:- Object-oriented programming (OOP) is a programming paradigm based on the concept of objects, which are data structures that contain data, in the form of fields (or attributes) and code, in the form of procedures, (or methods).  Comparison:-  **Java**is one of the most popular and widely used programming language and platform. A platform is an environment that helps to develop and run programs written in any programming language.  Java is fast, reliable and secure. From desktop to web applications, scientific supercomputers to gaming consoles, cell phones to the Internet, Java is used in every nook and corner.  C++ :-   * Java was basically derived from C++. * C++ is both procedural and object-oriented programming language whereas Java is a pure object-oriented language. * Both the languages have different objectives which means it has many differences too. * The main objective of C++ is to design a system of programming. * Java doesn’t support operator overloading but C++ does support it. * C++ also extends the C programming language whereas Java is basically created to support network computing. * Java doesn’t support structures and unions where C++ does support it. * Java is much slower than C++ in terms of execution. * C++ libraries are simple and also they are robust. It also provides container and associative arrays. But Java contains a powerful cross-platform library. * In Java, there is an automatic garbage collection whereas this is not the case in C++. In C++ all objects are destroyed manually with the help of the code. * C++ supports pointers which are variables which store addresses of other variables. But Java does not have any kind of variable which stores addresses of other variables. * C++ executes its programs very fast compared to Java.   C :-   * C is very much like C++(which was used to derive Java). In fact, C++ is an updated form of C. * C is a structure or procedure-oriented language whereas Java is an object-oriented programming language. * Execution time for programs written in C is very less when compared to Java. * C supports pointers whereas Java does not support variables for storing addresses of other variables. * C cannot handle exceptions in its program whereas Java is very good at handling exceptions.   JDK :-  The Java Development Kit (JDK) is a cross-platformed software development environment that offers a collection of tools and libraries necessary for developing Java-based software applications and applets. It is a core package used in Java, along with the JVM (Java Virtual Machine) and the JRE (Java Runtime Environment).  JVM :-The JVM is a specification, and can have different implementations, as long as they adhere to the specs. Oracle has its own JVM implementation (called the HotSpot JVM), the IBM has its own (the J9 JVM, for example).  JRE:- The Java Runtime Environment (JRE) is software that Java programs require to run correctly. Java is a computer language that powers many current web and mobile applications. The JRE is the underlying technology that communicates between the Java program and the operating system.  Javadoc:- Javadoc comments help the Javadoc Tool to generate better documentation. A Javadoc comment contains two parts. It contains the description of the code and Javadoc tags for specific meta-data. These comments are placed before the class, the field, or the method that we want to add information to.  Command Line Arguments :- The java command-line argument is an argument i.e. passed at the time of running the java program. The arguments passed from the console can be received in the java program and it can be used as an input. So, it provides a convenient way to check the behavior of the program for the different values. |
| **Practical 1.2** | Given a string, return a string made of the first 2 chars (if present), however include first char only if it is 'o' and include the second only if it is 'z', so "ozymandias" yields "oz".  startOz("ozymandias") → "oz"  startOz("bzoo") → "z"  startOz("oxx") → "o" |
| **Code** | //Problem 1 : Given a string, return a string made of the first 2 chars (if present),  // however include first char only  // if it is 'o' and include the second only if it is 'z', so "ozymandias" yields "oz".  // This program is made by 21CE105 Patel Vraj  import java.util.Scanner;  public class P1\_2 {      public static String startOz(String str){          if (str.startsWith("oz"))          return "oz";    else if (str.startsWith("o"))          return "o";    else if (str.startsWith("z",1))          return "z";    else          return "";      }      public static void main(String[] args) {          Scanner sc = new Scanner(System.in);          String s1 = sc.next();          String s2 = sc.next();          String s3 = sc.next();          System.out.println(startOz(s1));          System.out.println(startOz(s2));          System.out.println(startOz(s3));          System.out.println("This program is made by 21CE105 Patel Vraj");      }  } |
| **Output** |  |
| **GitHub Link** | https://github.com/PatelVraj10/java-practical-file/upload |
| **Practical 1.3** | Given two non-negative int values, return true if they have the same last digit, such as with 27 and 57. Note that the % "mod" operator computes remainders, so 17 % 10 is 7.  lastDigit(7, 17) → true  lastDigit(6, 17) → false  lastDigit(3, 113) → true |
| **Code** | // This program is made by 21CE105 Patel Vraj  import java.util.Scanner;  public class P1\_3 {      public static void main(String[] args) {          Scanzer sc=new Scanner(System.in);          for(int i=0;i<3;i++) {              int a = sc.nextInt();              int b = sc.nextInt();              if (a % 10 == b % 10) {                  System.out.println("true");              }              else {                  System.out.println("false");              }          }          System.out.println("This program is made by 21CE105 Patel Vraj");      }  } |
| **Output** |  |
| **GitHub** | https://github.com/PatelVraj10/java-practical-file/upload |
| **Practical 1.4** | Given an array of ints, return true if the sequence of numbers 1, 2, 3 appears in the array somewhere.  array123([1, 1, 2, 3, 1]) → true  array123([1, 1, 2, 4, 1]) → false  array123([1, 1, 2, 1, 2, 3]) → true |
| **Code** | // tHis program is made by 21CE105 Patel Vraj  import java.util.Scanner;  public class P1\_4 {      public static boolean arrOneTeoThree(int []arr){          for(int i=0;i<arr.length-1;i++){              if(arr[i]==1 && arr[i+1]==2 && arr[i+2]==3){                  return true;              }          }return false;      }      public static void main(String[] args) {          int [] arr1 = { 1,1,2,3,1};          int [] arr2 = {1,1,2,4,1};          int [] arr3 = {1,1,2,1,2,3};          System.out.println(arrOneTeoThree(arr1));          System.out.println(arrOneTeoThree(arr2));          System.out.println(arrOneTeoThree(arr3));          System.out.println("THis program is made by 21CE105 Patel Vraj");      }  } |
| **Output** |  |
| **GitHub** | https://github.com/PatelVraj10/java-practical-file/upload |
| **Practical 1.5** | Given 2 strings, a and b, return the number of the positions where they contain the same length 2 substring. So "xxcaazz" and "xxbaaz" yields 3, since the "xx", "aa", and "az" substrings appear in the same place in both strings. stringMatch("xxcaazz", "xxbaaz") → 3  stringMatch("abc", "abc") → 2  stringMatch("abc", "axc") → 0 |
| **Code** | // This program is made by 21CE105 Patel Vraj  public class P1\_5  {      public static int stringMatch(String str1, String str2)      {          String s1, s2;          int cnt = 0;          for(int i = 0; i< (str1.length()) - 1; i++)          {              if(i< (str1.length()) - 2)              {                  s1 = str1.substring(i, i + 2);                  s2 = str2.substring(i, i + 2);              }              else              {                  s1 = str1.substring(i);                  s2 = str2.substring(i);              }              if(s1.equals(s2))                  cnt++;          }          return cnt;      }      public static void main(String[] args)      {          System.out.println(stringMatch("xxcaazz", "xxbaaz"));          System.out.println(stringMatch("abc", "abc"));          System.out.println(stringMatch("abc", "axc"));          System.out.println("This program is made by 21CE105 Patel Vraj");      }  } |
| **Output** |  |
| **GitHub** | https://github.com/PatelVraj10/java-practical-file/upload |
| **Practical 1.6** | Given an array of strings, return a new array without the strings that are equal to the target string. One approach is to count the occurrences of the target string, make a new array of the correct length, and then copy over the correct strings.  wordsWithout(["a", "b", "c", "a"], "a") → ["b", "c"]  wordsWithout(["a", "b", "c", "a"], "b") → ["a", "c", "a"]  wordsWithout(["a", "b", "c", "a"], "c") → ["a", "b", "a"] |
| **Code** | // This program is made by 21CE105 Patel Vraj  import java.util.\*;  public class Ass6Targetmaqin {      public static void main(String[] args) {          Scanner sc = new Scanner(System.in);          System.out.print("Enter the String: ");          String s1 = sc.nextLine();          System.out.print("Enter the target string: ");          String s2 = sc.nextLine();          int a = s1.length(),b = 0;          for(int i=0;i<a;i++)          {              if(s1.charAt(i)==s2.charAt(0))              b++;          }          char[] target = new char[a-b];          int j = 0;          for(int i=0;i<a;i++)          {              if(s1.charAt(i)!=s2.charAt(0))              {                  target[j]=s1.charAt(i);                  j++;              }          }          System.out.print("The New string is ");          System.out.println(target);          System.out.println("This program is made by 21CE105 Patel Vraj");      }  } |
| **Output** |  |
| **GitHub** | https://github.com/PatelVraj10/java-practical-file/upload |
| **Practical 1.7** | Display numbers in a pyramid pattern.  1  1 2 1  1 2 4 2 1  1 2 4 8 4 2 1  1 2 4 8 16 8 4 2 1  1 2 4 8 16 32 16 8 4 2 1  1 2 4 8 16 32 64 32 16 8 4 2 1  1 2 4 8 16 32 64 128 64 32 16 8 4 2 1 |
| **Code** | //This program is made by 21CE105 Patel Vraj  public class P1\_7 {      public static void pyramid() {          int ix = 30;          for (int i = 1; i <= 128; i = i \* 2) {  // x is the number printed  //it gets the value from i,              for (int g = ix; g >= 0; g--) {                  System.out.print(" ");              }              for (int x2 = 1; x2 <= i - 1; x2 = x2 \* 2) {                  System.out.print("   ");                  System.out.print(x2);              }              for (int x = i; x >= 1; x = x / 2) {                  System.out.print("   ");                  System.out.print(x);              }              ix = ix - 4;              System.out.println();          }      }      public static void main(String[] args) {          pyramid();          System.out.println("This program is made by 21CE105 Patel Vraj");      }  } |
| **Output** |  |
| **GitHub** | https://github.com/PatelVraj10/java-practical-file/upload |
| **Practical 1.8** | The problem is to write a program that will grade multiple-choice tests. Assume there are eight students and ten questions, and the answers are stored in a twodimensional array. Each row records a student’s answers to the questions, as shown in the following array. Students’ Answers to the Questions: 0 1 2 3 4 5 6 7 8 9  Student 0 A B A C C D E E A D  Student 1 D B A B C A E E A D  Student 2 E D D A C B E E A D  Student 3 C B A E D C E E A D  Student 4 A B D C C D E E A D  Student 5 B B E C C D E E A D  Student 6 B B A C C D E E A D  Student 7 E B E C C D E E A D  The key is stored in a one-dimensional array:  Key to the Questions: 0 1 2 3 4 5 6 7 8 9  Key D B D C C D A E A D  Your program grades the test and displays the result. It compares each student’s answers with the key, counts the number of correct answers, and displays it. |
|  |
| **Code**  **Output** | // This program is made by 21CE105 Patel Vraj  import java.util.\*;  public class Ass8StudentAnsMain {      public static void main(String[] args) {          Scanner sc = new Scanner(System.in);          char[][] stu = new char[8][11];          for(int i=0;i<8;i++)          {              System.out.print("Enter the answers of Student "+i+": ");              stu[i][0] = (char)i;              for(int j=1;j<11;j++)              {                  char a = sc.next().charAt(0);                  stu[i][j] = a;              }          }          char key[] = {'D','B','D','C','C','D','A','E','A','D'};          int ans[][] = new int[8][2];          int c=0;          for(int i=0;i<8;i++)          {              ans[i][0] = stu[i][0];              for(int j=1;j<11;j++)              {                  if(stu[i][j]==key[j-1])                  {                      c++;                  }              }              ans[i][1]=c;              c=0;          }          System.out.println();          for(int i=0;i<8;i++)          {              System.out.println("Number of correct answers by Student "+ans[i][0]+" is "+ans[i][1]);          }          System.out.println("This program is made by 21CE105 Patel Vraj");      }    }  OUTPUT: |
| **GitHub** | https://github.com/PatelVraj10/java-practical-file/upload |
| **Practical 1.9** |  |
| **Code** | // his program is made by 21CE105 Patel Vraj  import java.io.\*;  import java.util.\*;  public class P1\_9{          static int N = 9;          // Function to check if all elements of the board[][] array store value in the range[1, 9]          static boolean isinRange(int[][] board)          {              for(int i = 0; i < N; i++)        // Traverse board[][] array              {                  for(int j = 0; j < N; j++)                  {                      // Check if board[i][j] lies in the range                      if (board[i][j] <= 0 || board[i][j] > 9)                      {                          return false;                      }                  }              }              return true;          }          // Function to check if the solution of sudoku puzzle is valid or not          static boolean isValidSudoku(int board[][])          {              if (isinRange(board) == false)   // Check if all elements of board[][] stores value in the range[1, 9]              {                  return false;              }              boolean[] unique = new boolean[N + 1];    // Stores unique value from 1 to N              for(int i = 0; i < N; i++)                // Traverse each row of the given array              {                  Arrays.fill(unique, false);       // Initialize unique[] array to false                  for(int j = 0; j < N; j++)            // Traverse each column of current row                  {                      int Z = board[i][j];              // Stores the value of board[i][j]                      if (unique[Z])                    // Check if current row stores duplicate value                      {                          return false;                      }                      unique[Z] = true;                  }              }              for(int i = 0; i < N; i++)               // Traverse each column of the given array              {                  Arrays.fill(unique, false);      // Initialize unique[] array to false                  for(int j = 0; j < N; j++)           // Traverse each row of current column                  {                      int Z = board[j][i];             // Stores the value of board[j][i]                      if (unique[Z])                   // Check if current column stores duplicate value                      {                          return false;                      }                      unique[Z] = true;                  }              }              for(int i = 0; i < N - 2; i += 3)            // Traverse each block of size 3 \* 3 in board[][] array              {                  for(int j = 0; j < N - 2; j += 3)        // j stores first column of each 3 \* 3 block                  {                      Arrays.fill(unique, false);      // Initialize unique[] array to false                      for(int k = 0; k < 3; k++)            // Traverse current block                      {                          for(int l = 0; l < 3; l++)                          {                              int X = i + k;          // Stores row number of current block                              int Y = j + l;          // Stores column number of current block                              int Z = board[X][Y];    // Stores the value of board[X][Y]                              if (unique[Z])          // Check if current block stores duplicate value                              {                                  return false;                              }                              unique[Z] = true;                          }                      }                  }              }              // If all conditions satisfied              return true;          }          public static void main(String[] args)          {              int[][] board = { { 5, 3, 4, 6, 7, 8, 9, 1, 2 },                      { 6, 7, 2, 1, 9, 5, 3, 4, 8 },                      { 1, 9, 8, 3, 4, 2, 5, 6, 7 },                      { 8, 5, 9, 7, 6, 1, 4, 2, 3 },                      { 4, 2, 6, 8, 5, 3, 7, 9, 1 },                      { 7, 1, 3, 9, 2, 4, 8, 5, 6 },                      { 9, 6, 1, 5, 3, 7, 2, 8, 4 },                      { 2, 8, 7, 4, 1, 9, 6, 3, 5 },                      { 3, 4, 5, 2, 8, 6, 1, 7, 9 } };              if (isValidSudoku(board))              {                  System.out.println("Valid");              }              else              {                  System.out.println("Not Valid");              }              System.out.println("This program is made by 21CE105 Patel Vraj");          }      } |
| **Output** |  |
| **GitHub** | https://github.com/PatelVraj10/java-practical-file/upload |
| **Practical 1.10** | Implement Caesar Cipher. |
| **Code** | // This program is made by 21CE105 Patel Vraj  import java.util.Scanner;  public class P1\_10 {          public static final String ALPHABET = "abcdefghijklmnopqrstuvwxyz";          public static String encrypt(String plainText, int shiftKey)          {              plainText = plainText.toLowerCase();              String cipherText = "";              for (int i = 0; i < plainText.length(); i++)              {                  int charPosition = ALPHABET.indexOf(plainText.charAt(i));                  int keyVal = (shiftKey + charPosition) % 26;                  char replaceVal = ALPHABET.charAt(keyVal);                  cipherText += replaceVal;              }              return cipherText;          }          public static String decrypt(String cipherText, int shiftKey)          {              cipherText = cipherText.toLowerCase();              String plainText = "";              for (int i = 0; i < cipherText.length(); i++)              {                  int charPosition = ALPHABET.indexOf(cipherText.charAt(i));                  int keyVal = (charPosition - shiftKey) % 26;                  if (keyVal < 0)                  {                      keyVal = ALPHABET.length() + keyVal;                  }                  char replaceVal = ALPHABET.charAt(keyVal);                  plainText += replaceVal;              }              return plainText;          }          public static void main(String[] args)          {              Scanner sc = new Scanner(System.in);              System.out.println("Enter the String for Encryption: ");              String message = new String();              message = sc.next();              System.out.println(encrypt(message, 3));              System.out.println(decrypt(encrypt(message, 3), 3));              System.out.println("This program is made by 21CE105 Patel Vraj");              sc.close();          }      } |
| **Output** |  |
| **GitHub** | https://github.com/PatelVraj10/java-practical-file/upload |