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:**

**a)OOPs:**

Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods). A common feature of objects is that procedures (or methods) are attached to them and can access and modify the object's data fields. In this brand of OOP, there is usually a special name such as this or self used to refer to the current object. In OOP, computer programs are designed by making them out of objects that interact with one another.OOP languages are diverse, but the most popular ones are class-based, meaning that objects are instances of classes, which also determine their types.

**b)Comparision**

**Python**

Python is a high-level language. It fully supports object-oriented programming. Python is not a pure object-oriented language. • Python is an interpreted language whereas Java is not an interpreted language, it is a compiled language. • Python is a scripting language whereas JAVA is a low-level implementation language. • Python is easy to use whereas JAVA is not as simple as Python. Programmers prefer to use python instead of Java because python contains less line of code whereas Java is just opposite to it. • Python programs are much shorter than JAVA programs. • Python is widely used in companies for building projects as its programs are shorter whereas JAVA is rarely used in companies for projects because it is difficult to use. • Python supports dynamic typing which is very useful for the programmers because they need to write less code because of which their time is saved and which is user-friendly as well as programmer-friendly. But in the case of JAVA, developers are required to define the type of each variable before using it which consumes the programmer’s lots of time. • Many large organizations like Google, Yahoo, NASA, etc. are making use of Python. But Python programs are generally expected to run slower than Java programs. • Java has much better library support for some use cases than Python which is the biggest advantage of JAVA. • Python is very much slower than Java.

**C++**

Java was basically derived from C++.C++ is both procedural and object-oriented programming language whereas Java is a pure objectoriented 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) Introduction to JDK, JRE, JVM, javadoc,command line argument**

1.JDK (Java Development Kit) is a Kit that provides the environment to develop and execute(run) the Java program. JDK is a kit(or package) that includes two things • Development Tools(to provide an environment to develop your java programs) • JRE (to execute your java program).

2. JRE (Java Runtime Environment) is an installation package that provides an environment to only run(not develop) the java program(or application)onto your machine. JRE is only used by those who only want to run Java programs that are end-users of your system.

3. JVM (Java Virtual Machine) is a very important part of both JDK and JRE because it is contained or inbuilt in both. Whatever Java program you run using JRE or JDK goes into JVM and JVM is responsible for executing the java program line by line, hence it is also known as an interpreter. Now let us discuss the components of JRE in order to understand its importance of it and perceive how it actually works. For this let us discuss components. The components of JRE are as follows:

Deployment technologies, including deployment, Java Web Start, and Java Plug-in. • User interface toolkits, including Abstract Window Toolkit (AWT), Swing, Java 2D, Accessibility, Image I/O, Print Service, Sound, drag, and drop (DnD), and input methods. • Integration libraries, including Interface Definition Language (IDL), Java Database Connectivity (JDBC), Java Naming and Directory Interface (JNDI), Remote Method Invocation (RMI), Remote Method Invocation Over Internet Inter-Orb Protocol (RMI-IIOP), and scripting. • Other base libraries, including international support, input/output (I/O), extension mechanism, Beans, Java Management Extensions (JMX), Java Native Interface (JNI), Math, Networking, Override Mechanism, Security, Serialization, and Java for XML Processing (XML JAXP). • Lang and util base libraries, including lang and util, management, versioning, zip, instrument, reflection, Collections, Concurrency Utilities, Java Archive (JAR), Logging, Preferences API, Ref Objects, and Regular Expressions. • Java Virtual Machine (JVM), including Java HotSpot Client and Server Virtual Machines

4. Javadoc Javadoc (originally cased JavaDoc) is a documentation generator created by Sun Microsystems for the Java language (now owned by Oracle Corporation) for generating API documentation in HTML format from Java source code.

5. command line argument • 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. You can pass N (1,2,3 and so on) numbers of arguments from the command prompt

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". startOz("ozymandias") → "oz" startOz("bzoo") → "z" startOz("oxx") → "o"**

**ANS:**

import java.util.Scanner;

public class p1 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter String:");

        String A = sc.nextLine();

        startOZ(A);

    }

    static void startOZ(String a) {

        if (a.charAt(0) == 'O' || a.charAt(0) == 'o') {

            System.out.print(a.charAt(0));

        }

        if (a.charAt(1) == 'z' || a.charAt(1) == 'Z') {

            System.out.print(a.charAt(1));

        }

    }

}

1. **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**

**ANS:**

import java.util.Scanner;

public class p2 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter first number:");

        int a = sc.nextInt();

        System.out.println("Enter second number:");

        int b = sc.nextInt();

        boolean c = lastDigit(a, b);

        System.out.println(c);

    }

    static boolean lastDigit(int p, int q) {

        if (p % 10 == q % 10)

            return true;

        else

            return false;

    }

}

1. **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**

**ANS:**

import java.util.Scanner;

public class p3 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter size of array : ");

        int a = sc.nextInt();

        int[] b = new int[a];

        System.out.println("Enter array :");

        for (int i = 0; i < a; i++) {

            b[i] = sc.nextInt();

        }

        boolean c = array123(b);

        System.out.println(c);

    }

    static boolean array123(int b[]) {

        int count1 = 0;

        for (int i = 0; i < b.length; i++) {

            if (b[i] == 1 && b[i + 1] == 2 && b[i + 2] == 3) {

                count1++;

            }

            if (i == b.length - 2) {

                break;

            }

        }

        if (count1 >= 1)

            return true;

        else

            return false;

    }

}

1. **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**

**ANS:**

import java.util.Scanner;

import static java.lang.Math.min;

public class p4 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter first string : ");

        String a = sc.next();

        System.out.println("Enter second string : ");

        String b = sc.next();

        int c = stringMatch(a, b);

        System.out.println(c);

    }

    static int stringMatch(String x, String y) {

        int count = 0;

        int f = min(x.length(), y.length());

        for (int i = 0; i < f - 1; i++) {

            String z = x.substring(i, i + 2);

            String t = y.substring(i, i + 2);

            if (z.equals(t)) {

                count++;

            }

        }

        return count;

    }

}

1. **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"]**

**ANS:**

import java.util.Scanner;

public class p5 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter size of array : ");

        int a = sc.nextInt();

        String[] b = new String[a];

        System.out.println("Enter array : ");

        for (int i = 0; i < b.length; i++) {

            b[i] = sc.next();

        }

        System.out.println("Enter string you want to remove");

        String c = sc.next();

        wordsWithout(b, c);

    }

    static void wordsWithout(String x[], String y) {

        int count = 0;

        for (int i = 0; i < x.length; i++) {

            if (x[i].equals(y)) {

                count++;

            }

        }

        String[] d = new String[x.length - count];

        int j = 0;

        for (int i = 0; i < x.length; i++) {

            if (x[i].equals(y)) {

            } else {

                d[j] = x[i];

                j++;

            }

        }

        System.out.println("New Array is :");

        for (String element : d) {

            System.out.println(element);

        }

    }

}

1. **Display numbers in a pyramid pattern.**

**ANS:**

import java.util.Scanner;

public class p6 {

    private static final String string = " ";

    public static void main(String[] args) {

        int number = 0;

        for (int row = 0; row <= 7; row++) {

            for (int column = 1; column <= 7 - row; column++) {

                System.out.printf("%4s", string);

            }

            for (int column = 0; column <= row; column++) {

                number = (int) Math.pow(2, column);

                System.out.printf("%4d", number);

            }

            for (int column = row - 1; column >= 0; column--) {

                number = (int) Math.pow(2, column);

                System.out.printf("%4d", number);

            }

            System.out.println();

        }

    }

}

1. **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 two dimensional array. Each row records a student’s answers to the questions, as shown in the following array. Students’ Answers to the Questions:**

**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.**

**ANS:**

import java.util.*\**;

class p7 {

    public static void main(String args[]) {

        p7 a = new p7();

        a.getdata();

        a.check();

    }

    String key[] = { "D", "B", "D", "C", "C", "D", "A", "E", "A", "D" };

    String mcq[][] = new String[8][10];

    String store[] = new String[10];

    int marks[] = new int[8];

    void getdata() {

        Scanner sc = new Scanner(System.in);

        for (int i = 0; i < 8; i++) {

            System.out.print("Enter Marks for Student Number " + (i + 1) + ": ");

            store[i] = sc.nextLine();

            mcq[i] = store[i].split(" ");

        }

    }

    void check() {

        for (int i = 0; i < 8; i++) {

            marks[i] = 0;

        }

        for (int i = 0; i < 8; i++) {

            for (int j = 0; j < 10; j++) {

                if (mcq[i][j].compareTo(key[j]) == 0) {

                    marks[i]++;

                }

            }

        }

        for (int i = 0; i < 8; i++) {

            System.out.println("Student Number " + (i + 1) + " has gained " + marks[i] + " marks");

        }

    }

}

1. **The problem is to check whether a given Sudoku solution is correct.**

**ANS:**

import java.util.Scanner;

public class p8 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int[][] a = new int[9][9];

        int f, ans = 0;

        for (int i = 0; i < 9; i++) {

            for (int j = 0; j < 9; j++) {

                a[i][j] = sc.nextInt();

            }

        }

        for (int i = 0; i < 9; i++) {

            for (int j = 0; j < 9; j++) {

                ans = ans ^ a[i][j];

            }

        }

        if (ans == 1)

            System.out.println("YES");

        else

            System.out.println("NO");

    }

}

1. **Implement Caesar Cipher.**

**ANS:**

import java.util.Scanner;

public class p9 {

    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));

        sc.close();

    }

}

**GITHUB LINK:**

**https://github.com/ocmodi21/JAVA**