Object Oriented Programming

K. Varsha
B191437
R.No: 27
E3-CSE-C1 (311)

ASSIGNMENT-1

1) Introduction to OOPS:

Define object oriented Programming (OOP) Paradigms and explain the basic concepts of behind it. Highlight the differences between procedure- oriented Programming (pop) and object oriented Programming?

Object oriented Programming (OOP) is a programming Paradigm that uses objects to represent real-world entities and concepts to represent real-world entities and concepts of OOPS. OOP offers Several benefits, including reusability, encapsulation and modularity. Here are Some common basic concepts of OOP. They are:

Basic concepts of Object Oriented Programming:-

1. Objects: objects are nothing but real (or) abstract items that contain data to define the object and methods that can manipulate the information.

2. classes:

Classes is a group of objects that has the Same properties and behaviour.

3. Encapsulation:

Encapsulation is the process of wrapping up data & functions to a Single unit. It is the most striking feature of the class.

4. Abstraction:

Abstraction represents essential features. It doesn't represent the background details and the explanation classes use the concept of abstraction.

5. Inheritance:

Inheritance is the property where by one class extends another class's properties, including additional methods and Variable.

6. Polymorphism:

In geek terms, polymorphism means the ability to take more than one form. An operation may exhibit different behaviour.

pop vs oop

Procedural oriented programming (pop) and object oriented programming (oop) are two different programmings

1) Approach: pop is based on the concept of writing procedures (or) functions the operate data. while oop is based on the concept of creating objects that contain both data & behaviour.

2) Data and Behaviour: In POP, data & behaviour are Separate and distinct, while in oop, data and behaviour are closed related and encapsulated with objects

3) code reusability: oop provides better code reusability because objects can be reused, different parts of the code.

4) complexity management:

OOP provides better complexity management. Because, it allows the creation of modula, reusable code that can be easily modified and also extended. while POP can lead to complex and hard to maintain code.

5) Inheritance:

OOP Supports inheritance, which allows objects to inherit properties and behaviour from other objects while POP does not.

6) Polymorphism:

OOP Supports polymorphism which allows objects to take on multiple forms and behave differently in different contexts, while pop does not.

Overall oop provides more flexibility, modular and reusable approach to programming, while pop is more straight forward.

2) Java Features and Environment:

Discuss the Variable history of java, its Key features and the jave development Kit (JDK). Explain the Significance of the java API.

History of Java:

Java is a high-level, object Oriented Programming language developed by Sun microSystems. Mike cheridan & Patrick naughton and was originally designed for programming consumer electronics like set top boxes.

The official release of java took place in 1995, and over the years, it has become one of the most widely uset programming languages in the world. Java has undergone several updates and revisions with Java2 (released in 1998) being a significant milestone that introduced the java2 platform, a standard edition (J2SE), java, platform enterprise edition (J2EE) and Java2 platform, micro Edition (J2ME).

In 2009, Sun microSystems was acquired by Oracle corporation and Oracle continues to maintain and also to develop java.

Key features in Java

1. Platform Independence: Java Programs can run on any device that has the java Virtual machine (JVM) installed. This is possible due to the principle, which allows code to the written once and execution on any platform.

2. Object Oriented: Java is a pure object Oriented programming language, which means that everything in java is an object.

3. Multi threading:

Java supports multi threading, allowing concurrent execution of two (or) more parts of a program for maximum utilization of CPU.

4. Security:

Java is designed with security features to protect Systems from harmful activities. The JVM plays a crucial role in this by providing a secure execution environment.

5. Robustness:

Java is Known for its strong memory management, exception handling and garbage collection, which contribute to the robustness of java applications.

6. Portability:

Java programs can be easily moved from one computer System to another without any type of modification.

* Java Development Kit:

The java development kit is a Software development kit used to develop the java applications it Includes the Java Runtime Environment (JRE) an interpreter/loader, a compiler (javac), an archiever (jar) a documentation generator. It provides everything a program needs to compile, run, debug and deploy java.

Java API and Environment:

The java API (Application Programming Interface) is a vast collection of classes and packages that provide pre-built functionality to java developers. It includes libraries for Network. It includes the JVM and the Java API, plays a crucial role in making java.

3) Describe the various java tokens, including Keywords, literals, identifiers and Separators. Discuss the different types of operations in java and their respective functionalities.

Java tokens:

In java, a program is made up of various tokens, which are the Smallest units of program that are to the compiler.

1. Keywords:

Keywords are reserved words in java, that have Special meanings & cannot be used as identifiers (variable names, class names) etc.

Ex: 'include', 'class', 'if', 'else', 'while'.

2. Literals:

Literals are constant values that are used in java. There are several types of literals.

Integer literals: Examples include '10', '-5' and '0xFF'.

Floating pointin literals: Examples 3.14 and -0.009.

Boolean literals: 'true' and 'false'.

character literals: Single character enclosed in double quotes 'A' or '5'.

String literals: "Hello, java!"

3. Identifiers:

Identifiers are used as names for classes, methods, variables and other user defined items. They must follow certain rules such as starting with letters being case-sensitive.

Ex: include 'main', 'count', 'myClass'.

4) Operators

Operators perform various operations on variables and values. They can be classified into several types.

(a) Arithmetic Operators:

'+' (addition), '-' (subtraction), '*' (multiplication), '/' (division), % (modulo)

(b) Relational operators:

'==' (equals to) '!=' (not equal to) '<' (less than)

'>' (greater than) '<=' (less than or equal to) '>=' (greater than equal to)

(C) Logical operators:

'&&' (logical AND) '||' (logical OR) '!' (logical NOT)

(d) Assignment Operators:

'=' assignment

'+=', '-=', '*=', '/=', '%=' (compound assignment)

(e) Bitwise Operators:

'&' (Bitwise AND), '|' (Bitwise OR), '^' (Bitwise XOR), '~' (Bitwise NOT)

'<<' (left shift), '>>' (Right shift), '>>>' (unsigned right shift)

f) Increment / Decrement Operators:

'++' (increment)

'--' (decrement)

g) conditional Operator (Ternary Operator):

'?:' (conditional operator)

5) Separators:

Separators are used to Separate tokens in java.

→ ';' (Semicolon) – terminates statements.

→ ',' (comma) – Separates items in a list

→ '.' (dot) – access members of a class (or) package

→ '()' (parenthesis) – encloses method parameters and control structures.

→ '{}' (braces) – defines a block of code.

4) Explain the different decision making statements (if-else), switch) and looping constructs (for, which, do-while) in java. And provide examples illustrating each?

* Decision - Making statements:

1) If -else statement:

The 'if-else' statement allows you to make decisions based on a condition if the condition is true the code inside the 'if' block is executed, otherwise the code inside the else block is executed.

<u>Ex:</u>
```
int num=10;
 if (num>0){
      System.out.Println ("positive number");
 }
 else{
      System.out.println ("Negative number");
 }
```

2) Switch Statement:

The `switch' statement is used to select one of many code blocks to be executed it is often used when you have a variable with multiple possible values.

Ex:
```
int dayOfWeek=3;
    String dayName;
    Switch (dayOfWeek){
         case 1:
               dayName= "Sunday";
               break;
         case 2:
               dayName= "Monday";
               break;
         default:
               dayName= "Invalid day';
    }
    System.out.println ('Day is:' +dayName);
```

Looping constructs:

1. For loop:

The 'for' loop is used when you know the no.of iterations in advance It consists of three parts: Initialization, condition and Iteration expression.

Ex:
```
for (int i=1; i<=5; i++)
    {
        System.out.Println("Iteration :"+i);
    }
```

2) While loop

The 'while' loop repeatedly executes a block of code as long as given condition is true.

Ex:
```
int count=0;
while (count<3){
      System.out.Println( "Count" +count);
      Count ++;
}
```

3) do-while loop:

The do-while loop is similar to the while loop, but it guarantees and that the block of code is executed atleast once because the condition is checked after the loop body.

Exn
```
int num;
Scanner read=new Scanner(System.in);
do{
    System.out.println("Enter a positive number:");
    num=read.nextInt();
}while (num<=0);
System.out.println("You entered a positive number:" +num);
```

Arrays and Strings

Discuss the concepts of one-dimensional arrays, multi-dimensional arrays & their Processing in java.

Explain the String class methods and the usage of String Buffer class?

One-dimensional Arrays:

In java, a one-dimensional array is a collection of elements of the same data type, arranged in a linear sequence.

Exn
```
//Declaration & Initialization of an Integer array.
int nums[]={1,2,3,4,5};
//Accessing elements
int e=~~ma~~nums[2]; //value is 3.
```

Processing Arrays:

Arrays in java can be processed using loops. for example, using a for loop to iterate through the elements of an array.

Exn
```
int nums[]={1,2,3,4,5};
for(int i=0; i<nums.length ;i++){
    System.out.println("nums[i]);
}
```

String class Methods:

The 'String' class in java provides a set of methods for manipulating strings Some common methods include,

→ length(): Returns the length of the String.

→ charAt (int index): returns the character at the specific index.

→ Substring (int begin, int end): returns a Substring within the specified range.

→ concat (String str): concatenates the specified string to the end of the current string.

→ indexOf (String str): returns the index of the first occurence of the specified substring.

Ex:
```
String msg = "Hello, java!";
int length = msg. length();
char firstChar = msg. charAt (0);
String sub = msg. Substring (7);
int index = msg. indexOf ("Java");
```

String Buffer class:

The "StringBuffer" class is part of the 'java.lang' package and provides a mutable sequence of characters unlike the 'string' class, 'String Buffer' objects can be modified after creation.

Ex:
```
StringBuffer Sb = new StringBuffer("Hello");
    Sb.append ("Java");    // Hello Java
    Sb. insert (6, ",");   // Hello, Java
    Sb.delete (5,7);       // HelloJava
```

Using 'String Buffer' is more efficient when frequent modifications to a string are needed, as it avoids the creation of new objects with each modification. However, for single-threaded applications, the 'String Buffer' class is often prefered, as it is not Synchronized and therefore, faster is a non-thread Safe context.