Exp no-10

Interface in Java

The interface in Java is *a mechanism to achieve abstraction*. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java.

Syntax:

```
interface <interface_name>{

    // declare constant fields
    // declare methods that abstract
    // by default.
}
```

To declare an interface, use the interface keyword. It is used to provide total abstraction. That means all the methods in an interface are declared with an empty body and are public and all fields are public, static, and final by default. A class that implements an interface must implement all the methods declared in the interface. To implement the interface, use the implements keyword.

Exp no-07

Difference Between Static and non-static in Java

In order to grasp how classes, variables, and methods operate in Java, it is crucial to comprehend the notions of static and non-static. Non-static members are linked to specific class instances, whereas static members are connected to the class. In this section, we will contrast static versus non-static Java components, highlighting their differences and potential applications.

Associated with

Static: Static members (variables and methods) are associated with the class itself rather than with individual instances.

Non-Static: Non-static members are specific to each instance of a class, as they are tied to objects created from the class.

Memory Allocation

Static: Static members are allocated memory only once, at the time of class loading. They are shared among all instances of the class.

Non-Static: Non-static members have memory allocated separately for each instance of the class. Each object has its own copy of non-static members.

Accessing

Static: Static members can be accessed directly using the class name followed by the member name (e.g., ClassName.memberName). They are accessible from anywhere within the program.

Non-Static: Non-static members are accessed using an object reference followed by the member name (e.g., objectReference.memberName). They are specific to a particular instance of the class.

Initialization

Static: Static members are initialized when the class is loaded into memory, typically during program startup. Initialization happens only once.

Non-Static: Non-static members are initialized when each instance of the class is created, usually using the new keyword. Initialization occurs separately for each object.

Exp no-05

Java Arrays

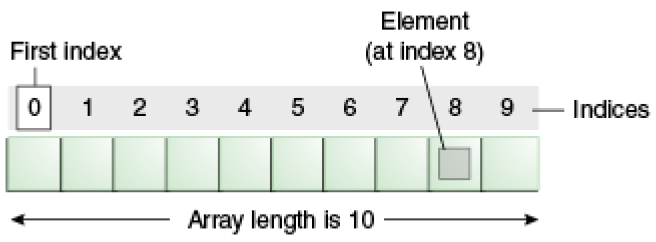
Normally, an array is a collection of similar type of elements which has contiguous memory location.

Java array is an object which contains elements of a similar data type. Additionally, The elements of an array are stored in a contiguous memory location. It is a data structure where we store similar elements. We can store only a fixed set of elements in a Java array.

Array in Java is index-based, the first element of the array is stored at the 0th index, 2nd element is stored on 1st index and so on.

In Java, array is an object of a dynamically generated class. Java array inherits the Object class, and implements the Serializable as well as Cloneable interfaces. We can store primitive values or objects in an array in Java. Like C/C++, we can also create single dimentional or multidimentional arrays in Java.

Moreover, Java provides the feature of anonymous arrays which is not available in C/C++.



Types of Array in java

There are two types of array.

- Single Dimensional Array
- Multidimensional Array

Exp no-04

Java String

In Java, string is basically an object that represents sequence of char values. An array of characters works same as Java string. For example:

```
1.  char[] ch={'j','a','v','a','t','p','o','i','n','t'};
2.  String s=new String(ch);
```

is same as:

```
1.  String s="javatpoint";
```

Java String class provides a lot of methods to perform operations on strings such as compare(), concat(), equals(), split(), length(), replace(), compareTo(), intern(), substring() etc.

What is String in Java?

Generally, String is a sequence of characters. But in Java, string is an object that represents a sequence of characters. The java.lang.String class is used to create a string object.

How to create a string object?

There are two ways to create String object:

1. By string literal
2. By new keyword

Exp no-06

Constructor in java-

In Java, a constructor is a block of codes similar to the method. It is called when an instance of the class is created. At the time of calling constructor, memory for the object is allocated in the memory.

Rules for creating Java constructor

There are two rules defined for the constructor.

1. Constructor name must be the same as its class name
2. A Constructor must have no explicit return type
3. A Java constructor cannot be abstract, static, final, and synchronized

Types of Java constructors

There are two types of constructors in Java:

1. Default constructor (no-arg constructor)
2. Parameterized constructor

Constructor overloading in Java

In Java, we can overload constructors like methods. The constructor overloading can be defined as the concept of having more than one constructor with different parameters so that every constructor can perform a different task.

Exp no-07

Inheritance in Java

Last Updated : 04 Mar, 2024

Java, Inheritance is an important pillar of OOP(Object-Oriented Programming). It is the mechanism in Java by which one class is allowed to inherit the features(fields and methods) of another class. In Java, Inheritance means creating new classes based on existing ones. A class that inherits from another class can reuse the methods and fields of that class. In addition, you can add new fields and methods to your current class as well.

How to Use Inheritance in Java?

The **extends keyword** is used for inheritance in Java. Using the extends keyword indicates you are derived from an existing class. In other words, “extends” refers to increased functionality.

Syntax :

```
class DerivedClass extends BaseClass
{
  //methods and fields
}
```

Java Inheritance Types

Below are the different types of inheritance which are supported by Java.

1. Single Inheritance
2. Multilevel Inheritance
3. Hierarchical Inheritance
4. Multiple Inheritance
5. Hybrid Inheritance

Method overriding-

If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding in Java**.

Exp no-11

Java Package

A **java package** is a group of similar types of classes, interfaces and sub-packages.

Package in java can be categorized in two form, built-in package and user-defined package.

There are many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc.

Here, we will have the detailed learning of creating and using user-defined packages.

How to access package from another package?

There are three ways to access the package from outside the package.

1. import package.*;
2. import package.classname;
3. fully qualified name.