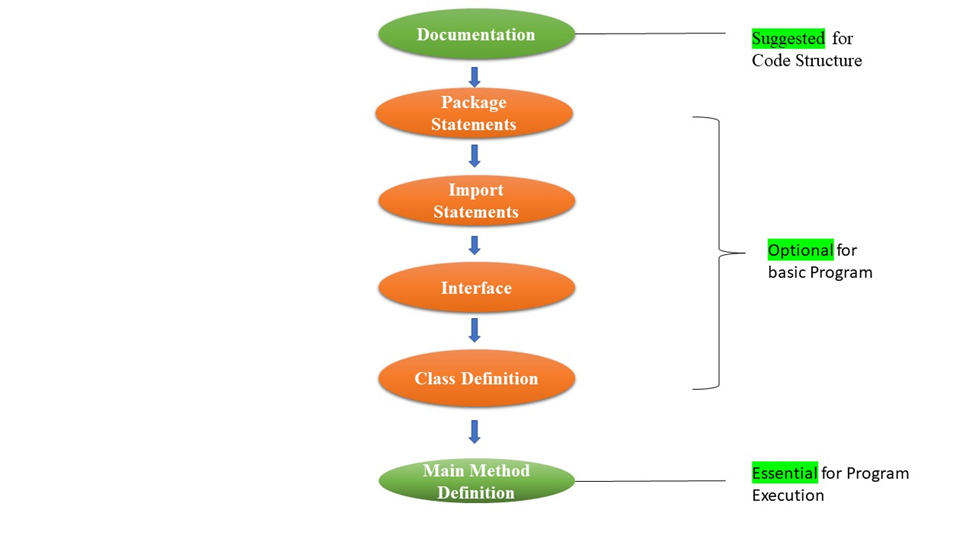
**Structure of Java Program**

Java is an object-oriented programming, platform-independent, and secure programming language that makes it popular. Using the Java programming language, we can develop a wide variety of applications. So, before diving in depth, it is necessary to understand the basic structure of Java program in detail.



**Documentation Section**

The documentation section is an important section but optional for a Java program. It includes basic information about a Java program. The information includes the author's name, date of creation, version, program name, company name, and description of the program. It improves the readability of the program. Whatever we write in the documentation section, the Java compiler ignores the statements during the execution of the program. To write the statements in the documentation section, we use comments. The comments may be single-line, multi-line, and documentation comments.

**Package Declaration**

The package declaration is optional. It is placed just after the documentation section. In this section, we declare the package name in which the class is placed. It must be defined before any class and interface declaration. It is necessary because a Java class can be placed in different packages and directories based on the module they are used. We use the keyword package to declare the package name. For example:

package javatpoint;

**Import Statements**

The package contains the many predefined classes and interfaces. If we want to use any class of a particular package, we need to import that class. We use the import keyword to import the class. It is written before the class declaration and after the package statement.Example:

import java.util.Scanner;

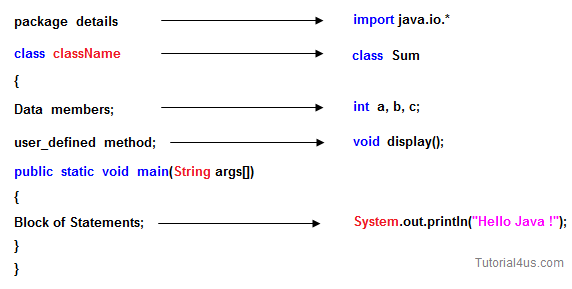
import java.util.\*;

**Interface Section**

It is an optional section.. We use the interface keyword to create an interface.. It contains only constants and method declarations. Another difference is that it cannot be instantiated. We can use interface in classes by using the implements keyword. An interface can also be used with other interfaces by using the extends keyword.

**Class Definition**

In this section, we define the class. It is vital part of a Java program. Without the class, we cannot create any Java program. A Java program may conation more than one class definition. We use the class keyword to define the class. The class is a blueprint of a Java program. It contains information about user-defined methods, variables, and constants. Every Java program has at least one class that contains the main() method. **For example:**



**Main Method Class**

In this section, we define the main() method. It is essential for all Java programs. Because the execution of all Java programs starts from the main() method. In other words, it is an entry point of the class. It must be inside the class. Inside the main method, we create objects and call the methods. We use the following statement to define the main() method.

**For example**: public class Student {

public static void main(String args[]) {

//statements

}

}

**Writing your first Java program**

import java.util.Scanner;

public class RectangleArea {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the length of the rectangle: ");

double length = scanner.nextDouble();

System.out.print("Enter the width of the rectangle: ");

double width = scanner.nextDouble();

double area = length \* width;

System.out.println("The area of the rectangle is: " + area);

}

}

**Constructor In Java**

A constructor in Java is a special method used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes. Every time an object is created using the new() keyword, at least one constructor is called. It calls a default constructor if there is no constructor available in the class. In such case, Java compiler provides a default constructor by default.

**Characteristics of a Constructor**

**Same Name as the Class:** A constructor must have the same name as the class it belongs to.

**No Return Type:** Constructors do not have a return type, not even void.

**Called Automatically:** A constructor is called automatically when an object is created.

**Types of Java constructors**

There are two types of constructors in Java:

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

**Default constructor**

A default constructor is a constructor without parameters. If a class has no constructors, the Java compiler automatically provides a default constructor. This is called an implicit default constructor. You can also define a default constructor explicitly.

**Example:**

public class Dog {

public Dog() {

System.out.println("A new dog is born!");

} }

public class Main {

public static void main(String[] args) {

Dog myDog = new Dog();

}

}

**Parameterized Constructor**

A parameterized constructor in Java is a constructor that takes parameters to initialize an object with specific values. Unlike the default constructor, which does not take any arguments, a parameterized constructor allows you to set initial values for the object's attributes when the object is created.

Examle:

class Student4{

int id;

String name;

Student4(int i,String n){

id = i;

name = n;

}

void display(){

System.out.println(id+" "+name); }

public static void main(String args[]){

Student4 s1 = new Student4(111,"Karan");

Student4 s2 = new Student4(222,"Aryan");

s1.display();

s2.display();

}

}