
Java™

1. Introduction to Programming

Types of Language

Procedural

- specifies a series of well structured steps and procedures to compose a program
- contains a systematic order of statements, function and commands to complete a task.

Functional

- Writing a program only in pure functions, i.e. never modify variables, but create new ones as an input.
- Used mainly when we have to perform different operations on the same set of data.

Object Oriented

- revolves around objects.
- code + data = object
- developed to make it easier to develop, debug, reuse and maintain software.

What is a first class function?

A programming language is said to have First-Class Functions, when functions in that language can be treated as a variable.

Static Language	Dynamic Language
1. Performs type checking at compile time.	1. Performs type checking at runtime.
2. Errors will show at compile time.	2. Error might not show, till the program is run.
3. Declare datatypes before you use it.	3. No need to declare datatype of variables.
4. More control.	4. Saves time in writing code, but might give error.

Memory Management

A programming language has two types of memory, (i) stack memory, (ii) heap memory.

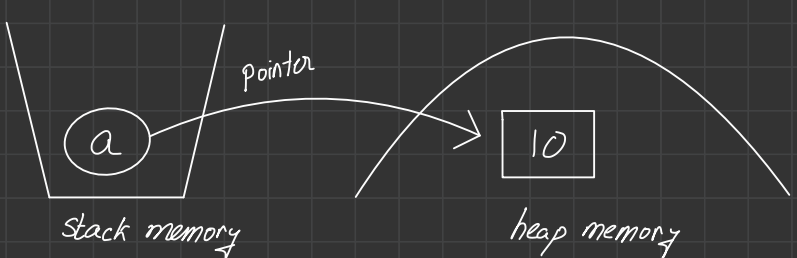
Let us consider a case, where we set a value to a variable.

```
a=10;
```

Reference variable → a

Object → 10

So in this case, the reference variable is stored in a stack, and the object is stored in a heap.



Now, if there is a scenario, where multiple reference variables point to the same object, and if there is a change in the original object, then the change will be seen by all the reference variables.

Ex:

```
int a[]= {1,2};
int b[] = a;
a[0] = 6;
```

→ In this scenario, the initial object of a is something. Then another another variable is also pointing to this same object. Now if the main object is modified, it will effect both the reference variables.

What is Garbage Collection in Java?

Java Application obtain objects in memory as needed. It is the task of the Garbage Collection (GC) in the Java Virtual Machine (JVM) to automatically determine what memory is no longer used by a Java Application and to recycle this memory for other users.

Ex:

```
int a = 10;
a = 20;
```

→ In this example, reference variable 'a' points to object later, this reference variable points to another object. And the previous object is left unused. So, when the Garbage Collection is called, this unreferenced object is basically destructed.

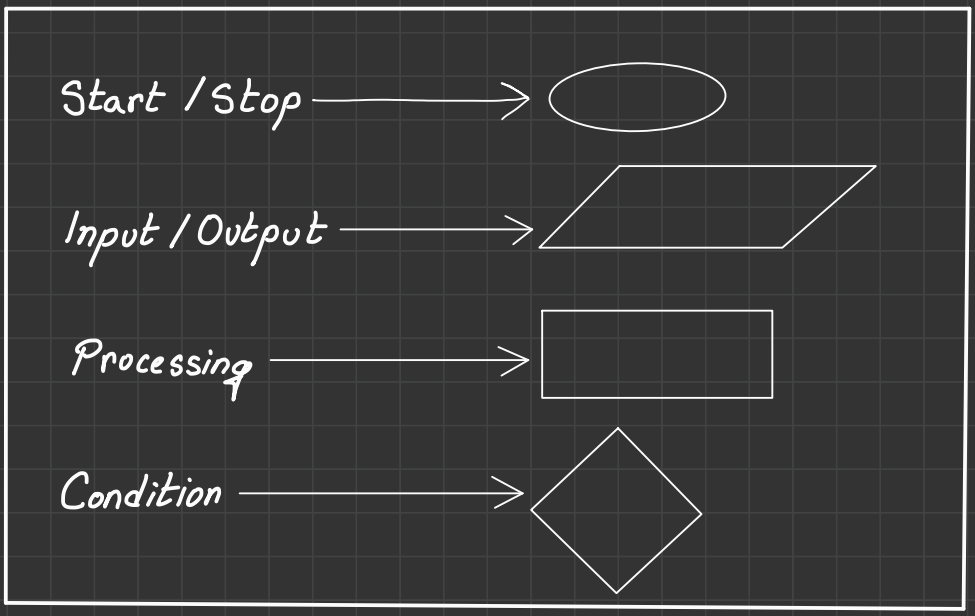
How to manually call the Garbage collection?

So, manually, the garbage collection can be invoked the following two ways:

- System.gc();
- Runtime.getRuntime().gc();

2. Flow of Program

Flowchart



A flowchart, is basically a diagramatic process, which details how a certain problem can be solved.

While, pseudocode is a rough logical written version.

3. Introduction to Java-Architecture

Java code execution

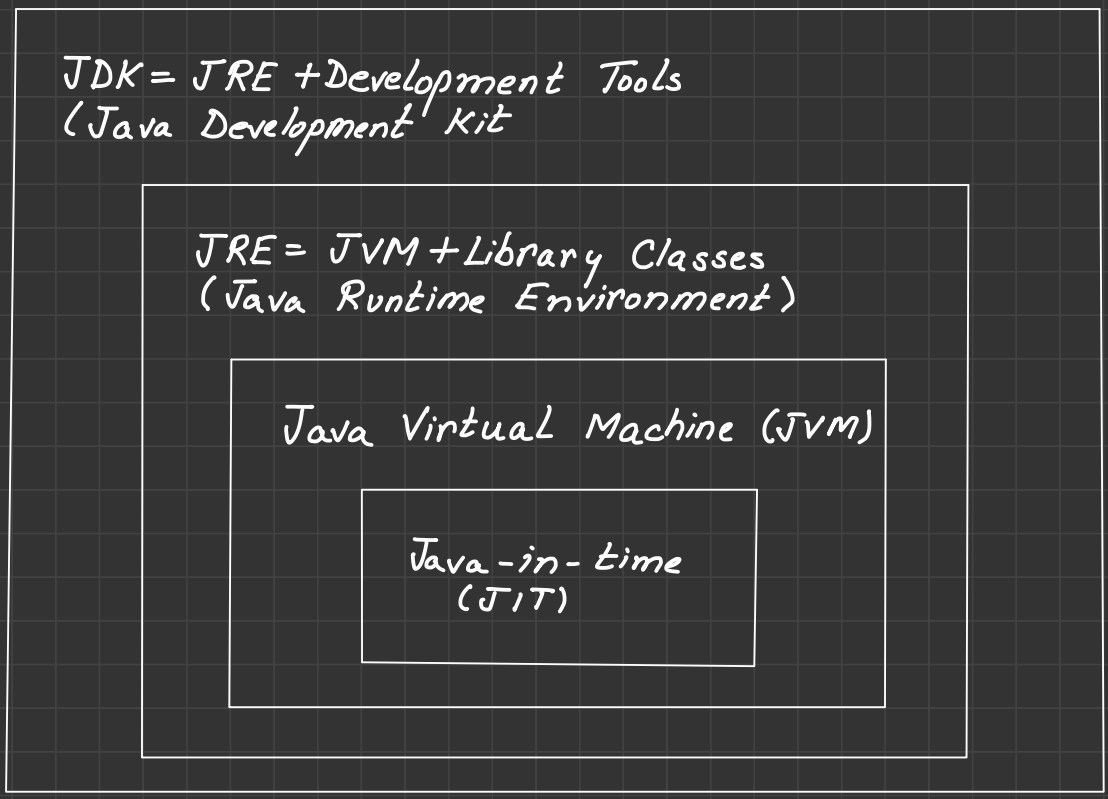
this is the source code

- this code will not directly run on the system
- JVM is needed to run this code
- This is reason, why Java is platform independent.

More about platform independence

- It means that byte code can run on all operating systems.
- We need to convert source code to machine code, so computer can understand
- Compiler helps in doing this by turning it into executable code
- This executable code is a set of instructions for the computer.
- After compiling C/C++ file, we get a .exe file, which is platform dependent.
- In Java we get bytecode, JVM converts this to machine code.
- Java is platform-independent, but JVM is platform dependent.

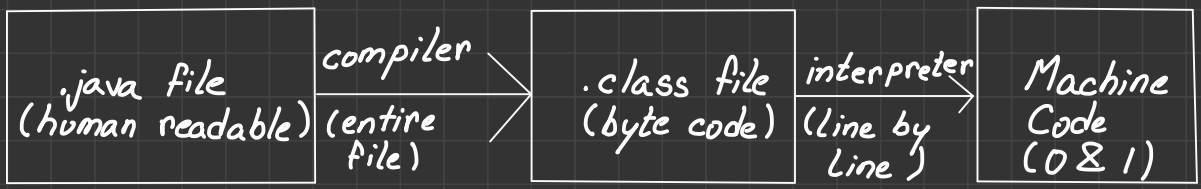
JDK vs JRE vs JVM vs JIT



JDK

- Provides environment to develop and run the Java program.
- It is a package that includes:
  1. Development Tools - To provide an environment to develop your program.
  2. JRE - To execute your program
  3. a compiler - javac
  4. archiver - jar
  5. docs generator - javadoc
  6. interpreter / loader

How Java code executes



This is the source code

- this code will not directly run on a system.
- we need JVM to run this
- Reason why Java is platform independent

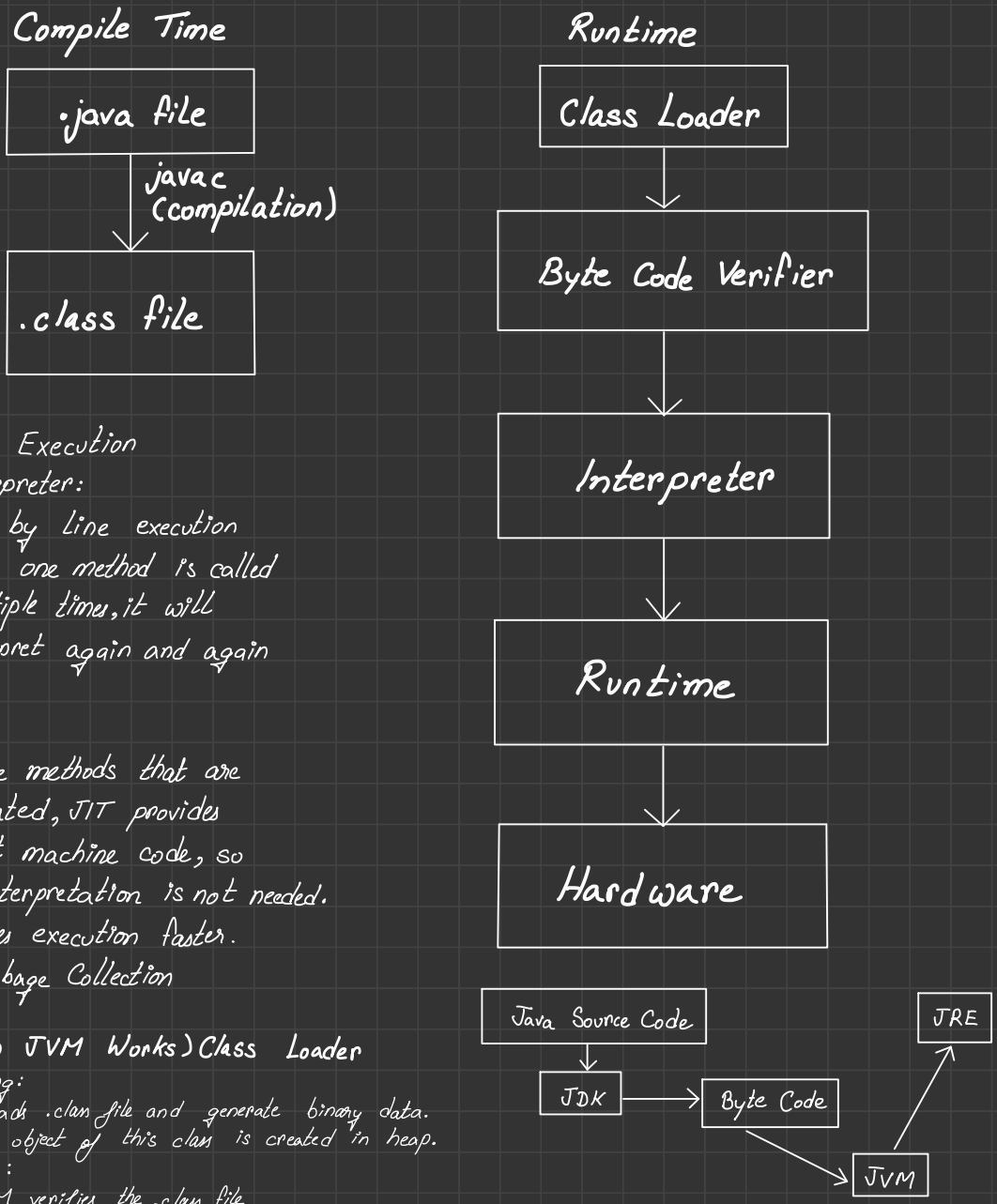
JDK

- Provides environment to develop and run the Java Program
- It is a package that includes:
  1. Development Tools - To provide an environment to develop your program
  2. JRE - to execute your program
  3. a compiler - javac
  4. archiver - jar
  5. docs generator - javadoc
  6. interpreter / loader

JRE

- It is an installation package that provides environment to only run the program
- It consists of:
  1. Deployment Technologies
  2. User interface toolkits
  3. Integration Libraries
  4. Base Libraries
  5. JVM

- After we get the .class file, the next things happen at runtime:
  1. Class loader loads all classes needed to execute the program.
  2. JVM sends code to Byte code verifier to check the format of code.



JVM Execution

Interpreter:

- Line by line execution
- When one method is called multiple times, it will interpret again and again

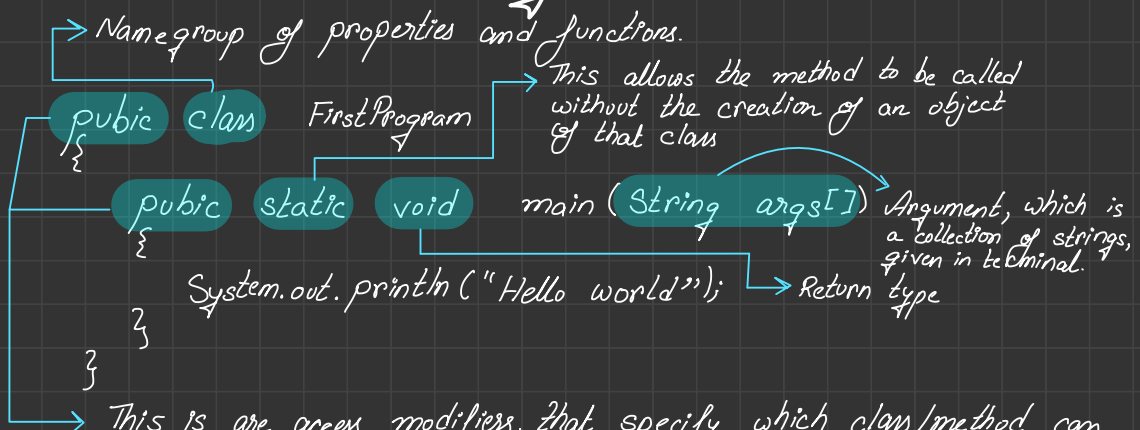
JIT:

- Those methods that are repeated, JIT provides direct machine code, so re-interpretation is not needed.
- Makes execution faster.
- Garbage Collection

(How JVM Works) Class Loader

- Loading:
  - Reads .class file and generate binary data.
  - An object of this class is created in heap.
- Linking:
  - JVM verifies the .class file
  - Allocates memory for class variables & default values.
  - Replace symbolic references from the type with direct references.
- Initilization
  - All static variables are assigned with their values defined in the code and static block.
- JVM contains the stack and heap memory allocations.

4. First Java Program



Converting Java File to Byte Code

If we consider the above code, the Java file name will be FirstProgram So we need to use the java compiler. So, go to the terminal and give the following command.

```
javac FirstProgram.java
```

So, the java compiler doesnot compile it to machine code. It compiles it to byte code. And that can be independently run from any platform. And now, to run this program, we need to execute the byte code. And to execute it, we use the following command.

```
java FirstProgram
```

OR

```
javac -d .. FirstProgram.java
```

→ This command specifies the directory in in which this bytecode will be generated.

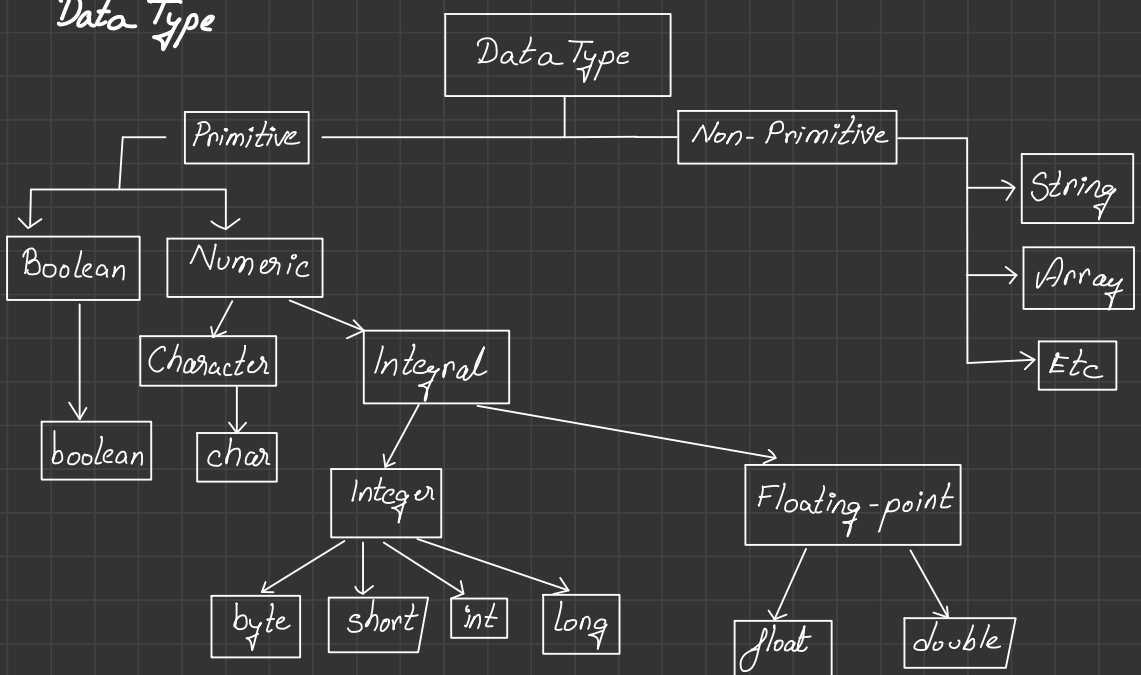
↳ -d is followed by the path of directory for bytecode.

Java Package

A java package is a group of similar types of classes, interfaces and sub-packages.
Packages in Java can be categorised in two forms, (i) built-in package, and (ii) User defined package.

- Java package is used to categorize the classes and interfaces, so that they can be easily maintained.
- Java package provides access protection.
- Java package removes name collision.

Data Type



Type Casting

If one type of data is assigned to another type of variable, an automatic conversion takes place, if the following conditions are met:
(i) The two types should be compatible.

The process of converting one type of object and variable into another type is called type casting. When the conversion is automatically performed by the compiler without the programmers interference, it is called implicit type casting, or widening casting.
In implicit typecasting, the conversion involves a larger datatype being converted to a smaller one. It is also called a Narrowing Cast.

Automatic Type Promotion

```
int a = 257;
byte b = (byte) a;
System.out.println(b);
```

Output: 1.

Here, in this example, when casting, if the larger element is cast to a lower element, that exceeds its limits, it stores the modulus wrt its limit.
Hence, b = 257 % 256

5. Conditionals and Loops

if.... else condition

-In conditionals like if, there is a statement, that returns either a true, or a false. If it is true, then the if block will be executed, and if it is false, then it will go to the else if block, or the else block.

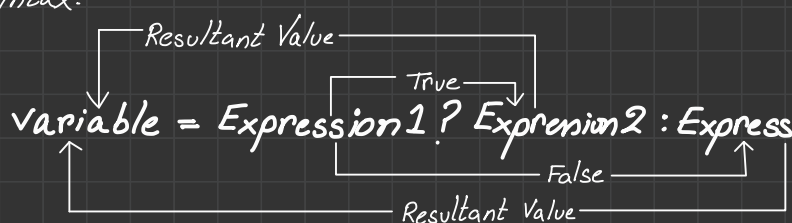
```
if(condition1)
{
   if block
}
else if(statement2)
{
   else if block
}
else
{
   else block
}
```

When all the conditions are false, the else block will get executed.

Ternary Operation

Java ternary operator is the only conditional operator that takes three operands. It is a one-linear replacement for the if-then-else statement, and used a lot in Java Programming.

Syntax:



Loops

The Java for loop is used to iterate a part of the program several times. If the number of iteration is fixed, it is recommended to use for loop.
Loops in Java are of the following types:

1. for Loop - The Java for loop is used to iterate a part of the program several times. If the number of iteration is fixed, it is recommended to use for loop.
Syntax:

```
for (int i=0; i<n; i++)
{ for loop block }
```

2. while Loop - The Java while loop is used to iterate a part of the program several times. If the number of iteration is not fixed, it is recommended to use while loop. It is an entry controlled loop.
Syntax:

```
while (condition)
{ while loop block }
```

When the condition is true, the loop block gets executed.

3. do-while Loop - The Java do-while loop is used to iterate a part of the program several times. Use it if the number of the iteration is not fixed and you must have to execute the loop at least once. It is an exit controlled loop.
Syntax:

```
do{
  do-while block
} while (condition)
```

When the condition is true, the loop block will get executed.

4. For-each Loop - The Java for-each loop or enhanced loop is introduced since J2SE 5.0. It provides an alternative approach to traverse the array or collection elements. The advantage for the for-each loop is that it eliminates the possibility of bugs and makes the code more readable. Syntax:

```
for (data_type variable : array | collection) {for-each block}
```

6. Switch Statement

In switch statements, you can jump to various cases based on your expression.

Syntax:

```
switch (expression) {
    //cases
    case one:
        //do something
        break;
    case two:
        //do something
        break;
    default:
        //do something
}
```

NOTE
- cases have to be the same type as expression, must be a constant or literal.
- duplicate case values are not allowed.
- break is used to terminate the sequence.
- if the break is not used, it will continue to the next case.
- default will execute, when none of the above does.
- if default is not at the end, put a break after it.

Enhanced Switch Condition

```
switch (day) {
    case 1,2,3,4,5 -> System.out.println("Weekday");
    case 6,7 -> System.out.println("Weekend");
}
```

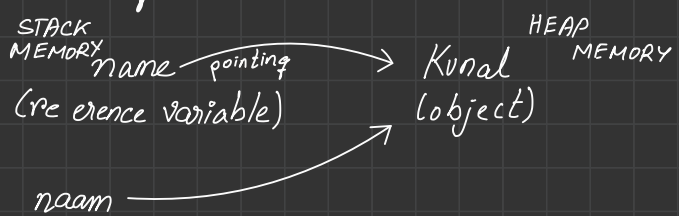
7. Functions or Methods

Any function in a Java that is in a class, is called a method.

Sample:

```
public class PassingExample
{
    public static void main(String[] args)
    {
        String name = "Kunal Kushwaha";
        greet(name);
    }
    static void greet(String naam)
    {
        System.out.println(naam);
    }
}
```

Internal Working



In Java, there is only pass by value. So in this case, a copy of the reference variable is passed over here.

Now, if we make changes in the naam variable, it doesn't change the original object, as string is immutable due to security reason. If string class is checked, you will find the final keyword in that class.

Scoping

Scoping basically means where we can access our variables.

Example

```
public class Scope
{

    public static void main(String[] args)
    {

        int a=10;
        int b=20;
    }

    static void random()
    {
        System.out.println(a);
    }
}
```

We will get an error here, as the variable 'a' is not in the scope of this function.

A simple thumb rule is, if a variable is declared, its scope is within the brackets enclosing it.

Shadowing

Shadowing is a practice of using two variables with the same name in the scope that overlaps.

Example

```
public class Shadowing
{

    static int x=90;
    public static void main(String[] args)
    {

        int x = 10;
        System.out.println(x);
        System.out.println(this.x);
    }
}
```

When there is overlap of scope, the variable in the higher scope is hidden. So that variable is shadowed.

But outside the lower scope, the value of the higher scoped variable remains unchanged.

OUTPUT: 10
90

Variable Arguments (Varargs)

Variable length arguments is basically, when we create a method, and the number of arguments, that are passed in it are variable.

Example

```
public class VarArgs
{
    public static void main(String[] args)
    {
        fun(1,2,3);
        fun(1,2,3,4,5,6,7);
    }
    static void fun(int ...v)
    {
        System.out.println(Arrays.toString(v));
    }
}
```

Number of arguments is three

Number of arguments is seven.

This is treated as an integer array.

In a function, when varargs is used with other arguments, then the varargs is used at the end.

> Example: static void multiple(int a, int b, int ... var)

Method Overloading in Java

If a class has multiple methods having same name, but different in parameters, it is known as Method Overloading.

If we have to perform only one operation, having same name of methods increases the readability of the program.

Different ways to overload the method:

1. By changing the number of arguments.
2. By changing the data type.