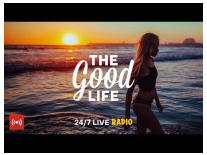
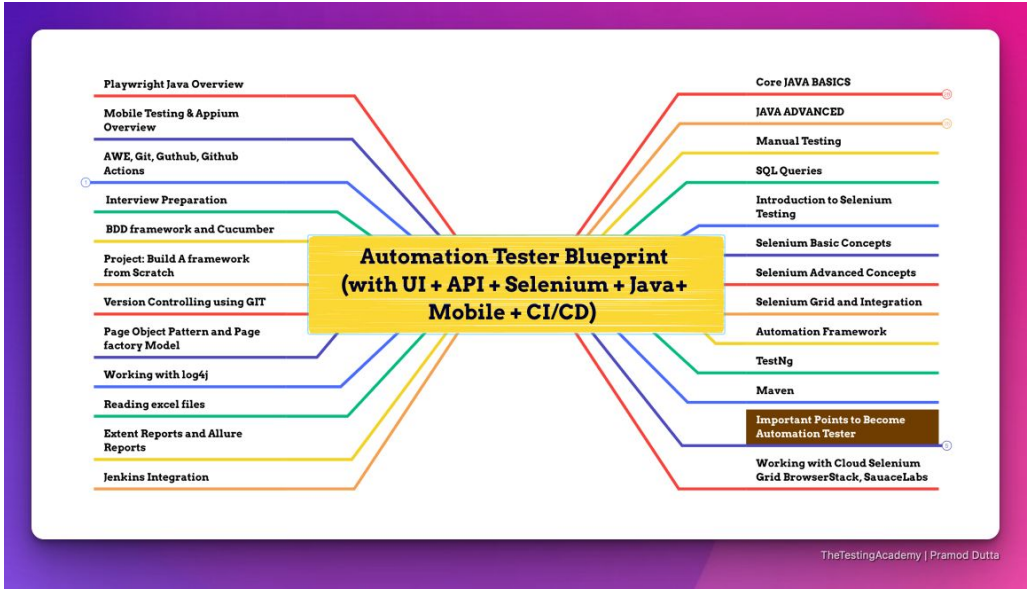
Meeting Etiquette

- Please be on Mute all the time.
- Turn off Video, So that we can save bandwidth.
- There is separate Q&A Section in End Please use that, Add questions in Google Form.
- Break at 5 min.
- Mute your microphone
- To help keep background noise to a minimum, make sure you mute your microphone when you are not speaking.
- Be mindful of background noise
- Avoid multitasking
- All links and Slides will be shared.



{ Automation Tester } BluePrint.

Pramod Dutta



Core Java + Interview Tips

Core Java
Intro to programming

Core Java
Intro to programming

Agenda

- Install Java JDK
- Variables
- Data Types
- String Class & Casting
- Operators
    a. Arithmetic Operators
    b. Assignment Operators
    c. Comparison/Relational Operators
    d. Logical Operators

Agenda

- Math class
- Taking Input
- Conditional Statements ‘if-else’
- Class
- Object
- Mini-Project Problem

Why Do We Need Programming Languages?

- Computers can't understand English
  - Too ambiguous (PB&J)
- Humans can't easily write machine code

Teach Yourself Programming in Ten Years

Peter Norvig

Why is everyone in such a rush?

Walk into any bookstore, and you'll see how to *Teach Yourself Java in 24 Hours* alongside endless variations offering to teach C, SQL, Ruby, Algorithms, and so on in a few days or hours. The Amazon advanced search for [title: teach, yourself, hours, since: 2000 and found 512 such books. Of the top ten, nine are programming books (the other is about bookkeeping). Similar results come from replacing "teach yourself" with "learn" or "hours" with "days."

The conclusion is that either people are in a big rush to learn about programming, or that programming is somehow fabulously easier to learn than anything else. Felleisen *et al*. give a nod to this trend in their book *How to Design Programs*, when they say "Bad programming is easy. *Idiots* can learn it in *21 days*, even if they are *dummies*." The Abtruse Goose comic also had their take.

Let's analyze what a title like *Teach Yourself C++ in 24 Hours* could mean:

Translations

Thanks to the following authors, translations of this page are available in:

Arabic
(Mohamed A. Yahya)
العربية

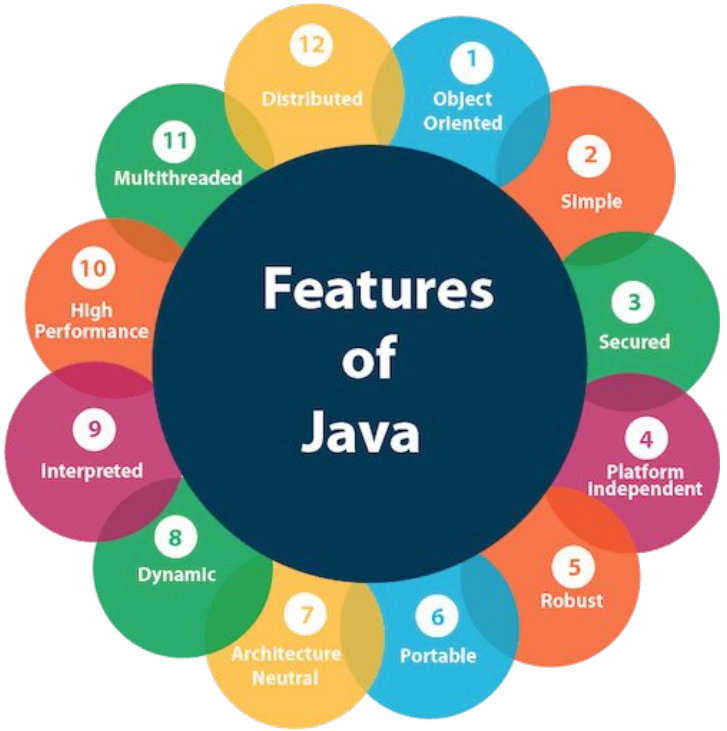
Bulgarian
(Boyko Bantchev)

https://norvig.com/21-days.html

What is Java?

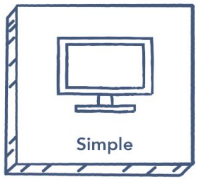
Java is an object oriented language.

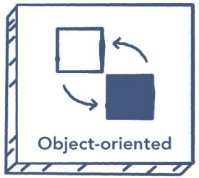
Java is a programming language and computing platform first released by Sun Microsystems in 1995.



Primary goals in the creation of the Java language

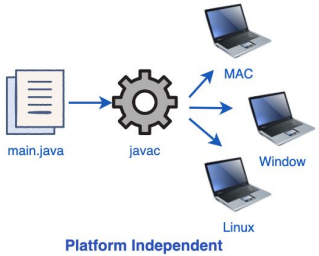
- It is **Simple & Portable**: Memory Management using Pointers is not allowed.



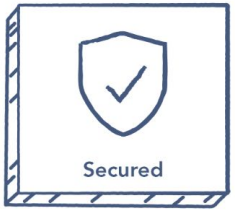


- It is **Object-Oriented**.

- It is **Independent** of the host platform.

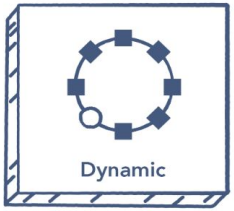


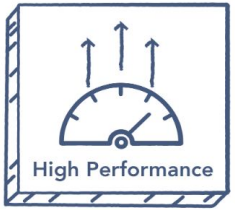
Primary goals in the creation of the Java language



- It is **Secured & Dynamic**: Designed to execute code from remote sources securely.

- It contains language facilities and libraries for **Networking.**

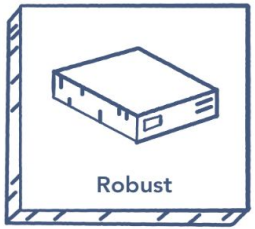


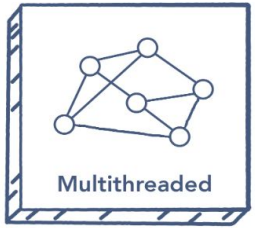


- **High Performance:** With the use of JIT (*Just-In-Time*) compilers, Java achieves high performance through the use of **byte-code** that can be easily translated into native machine code.

Primary goals in the creation of the Java language

- It is **Robust**: Java has its own strong **memory management** system. This helps to eliminate errors as it checks the code during compile and runtime.





- Java is **Multithreaded:** It supports multiple executions of threads (i.e., *lightweight processes*), including a set of synchronization primitives.

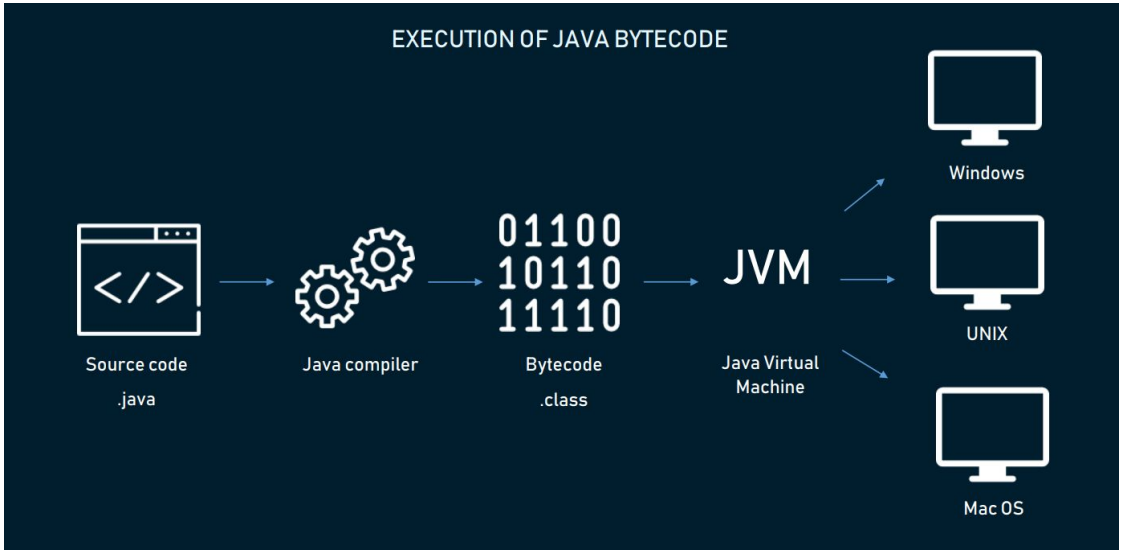
The Java language introduces some new features that did not exist in other languages like C and C++.

Bad practices#

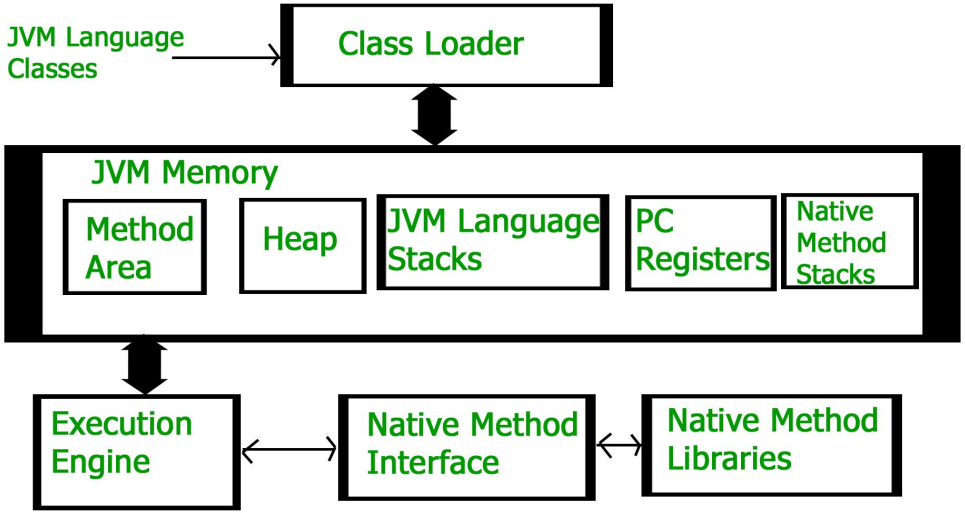
Over the years, some features in C/C++ programming became abused by the programmers. Although the language allows it, it was known as bad practices. So the creators of Java have disabled them:

- Use of Pointers
- Operator overloading
- Multiple inheritance
- Friend classes (access another object's private members)
- Restrictions of explicit type casting (related to memory management)

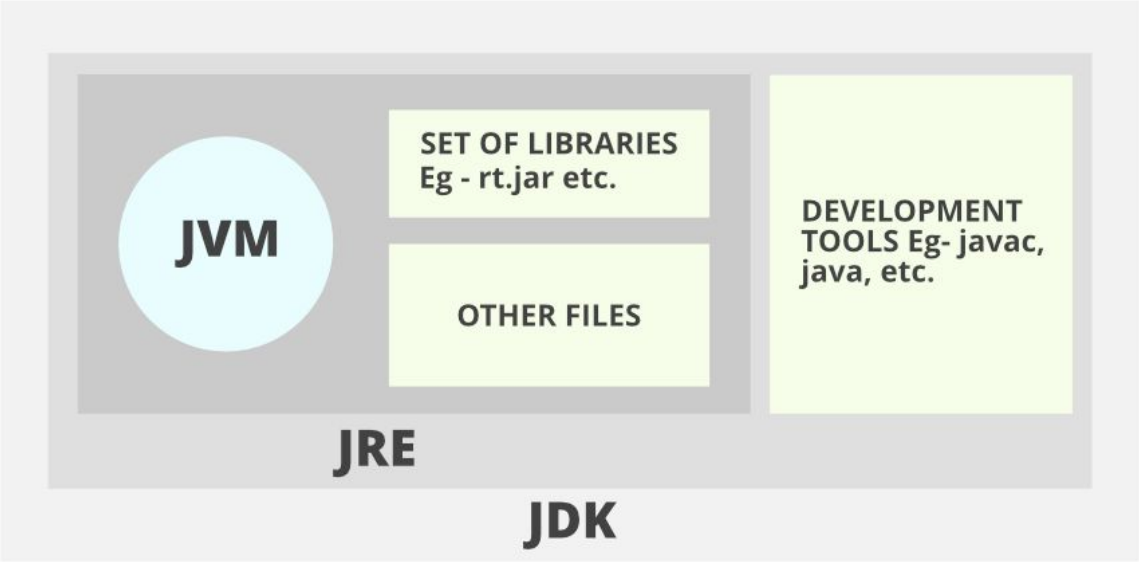
Core Funda of Java is Write once and Run anywhere!!



Run anywhere ??
JVM Architecture



Run anywhere ??
JVM Architecture



Java Development Kit (JDK)

JRE (Java Runtime Environment)

JVM (Java Virtual Machine)

JDK > JRE> JVM

Java Development Kit (JDK) — used for developing Java applications

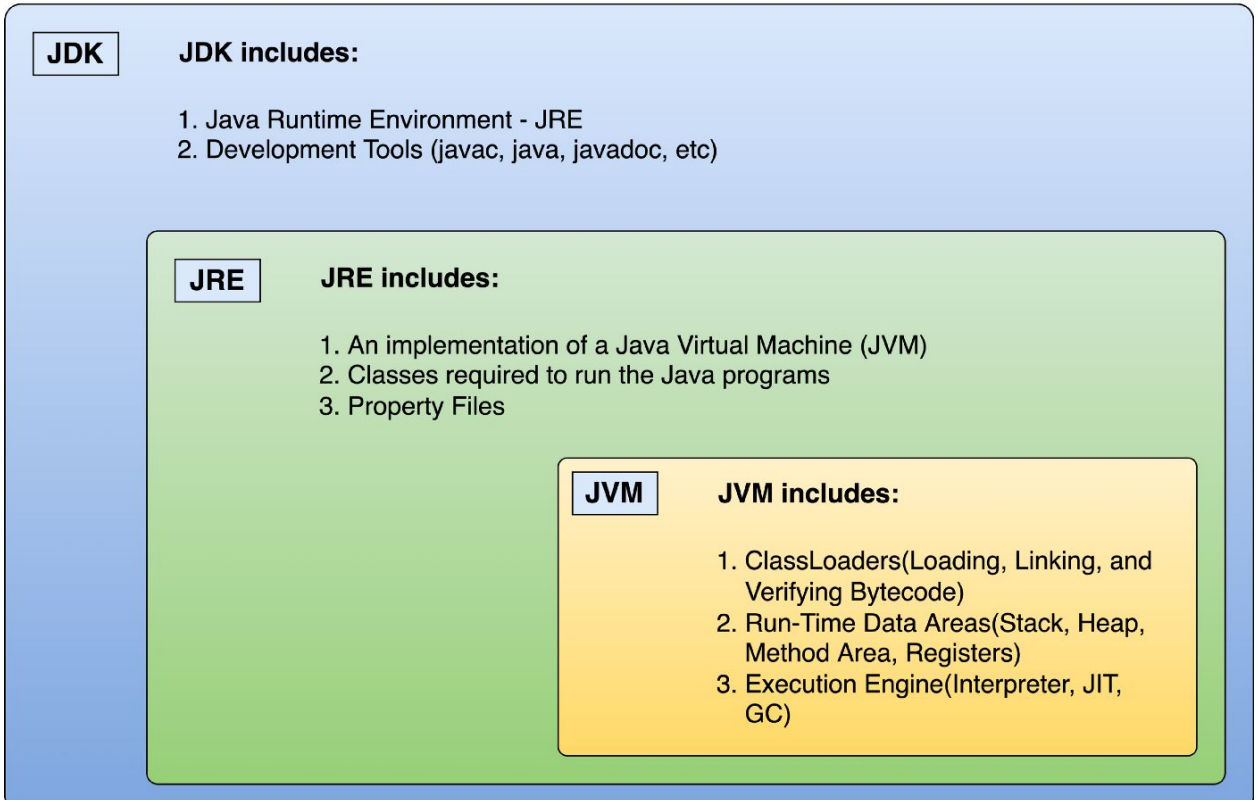
includes the Java Runtime Environment (JRE), an interpreter/loader (Java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc), and other tools needed in Java development.

JRE (Java Runtime Environment)

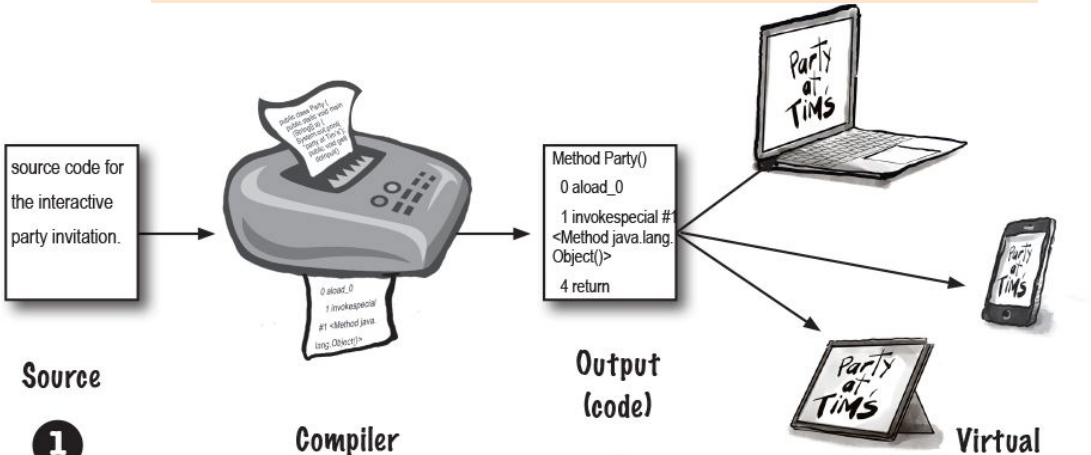
to execute your java program **not develop**

JVM (Java Virtual Machine) — JVM is responsible for executing the java program line by line, hence it is also known as an interpreter.

JDK > JRE> JVM


JDK
JDK includes:
1. Java Runtime Environment - JRE
2. Development Tools (javac, java, javadoc, etc)
JRE
JRE includes:
1. An implementation of a Java Virtual Machine (JVM)
2. Classes required to run the Java programs
3. Property Files
JVM
JVM includes:
1. ClassLoaders(Loading, Linking, and Verifying Bytecode)
2. Run-Time Data Areas(Stack, Heap, Method Area, Registers)
3. Execution Engine(Interpreter, JIT, GC)

How Java Works?



Source

1

Create a source document. Use an established protocol (in this case, the Java language).

Compiler

2

Run your document through a source code compiler. The compiler checks for errors and won't let you compile until it's satisfied that everything will run correctly.

Output (code)

3

The compiler creates a new document, coded into Java ***bytecode***. Any device capable of running Java will be able to interpret/translate this file into something it can run. The compiled bytecode is platform-independent.

Virtual Machines

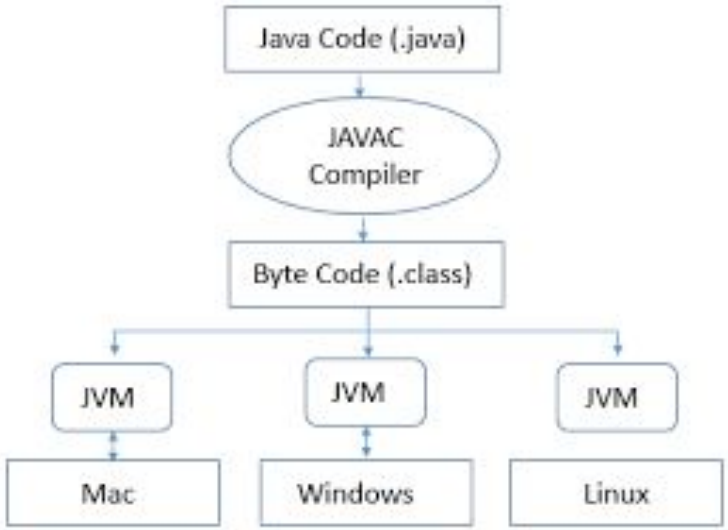
4

Your friends all have a Java ***virtual*** machine (JVM), implemented in software, running inside their electronic gadgets. When your friends run your program, the virtual machine reads and runs the bytecode.

401
Response

How Java Works?

Source Code -> Compile It (Javac) -> Run the Compiled Code -> JVM installed on Machines



401
Response

Running Java File Using CMD

filename.java

Move to src

java com.thetestingacademy.MyClass

MyClass.java

```java
package com.thetestingacademy;

public class MyClass {

    public static void main(String[] args) {
    // write your code here
        System.out.println("Pramod Hello");
    }
}
```

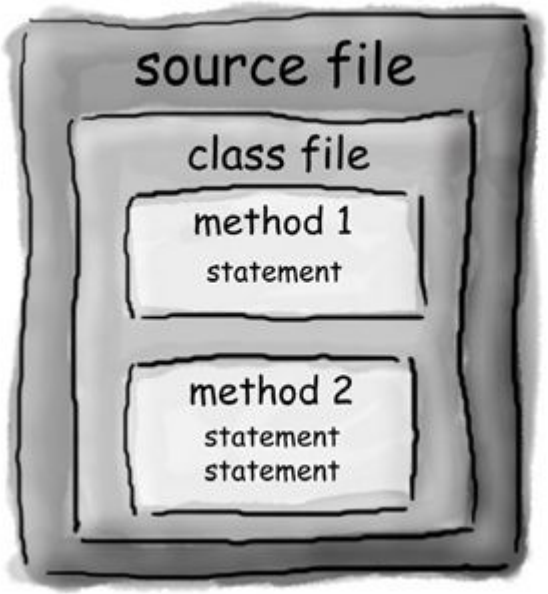
```
→  LearnJava cd src
→  src java com.thetestingacademy.MyClass
Pramod Hello
→  src pwd
/Users/pramod/Documents/MacBookBackup/Code/ATB/LearnJava/src
→  src
```

What does the compiler do?

- The compiler is a translator that acts as an intermediary between the programmer and the CPU on the computer.
- Javac.exe, Mac - javac(sh)
-

401
Response

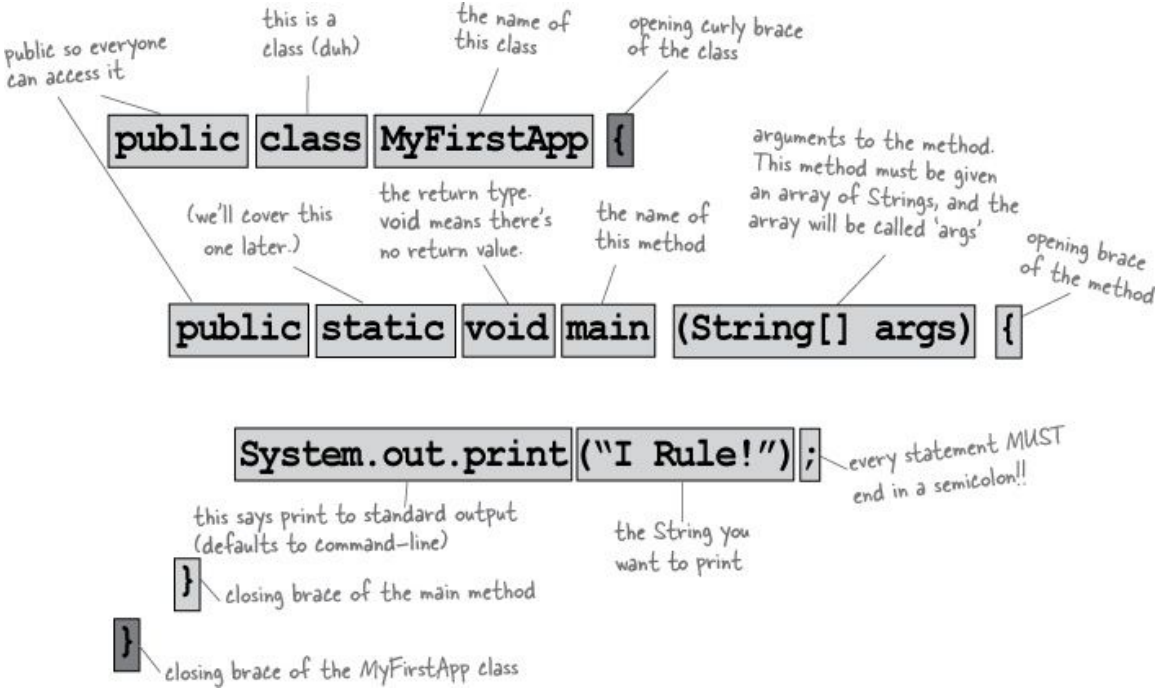
Code Structure in Java



Class Holds -> Methods -> Statements (what Methods can do)

401
Response

Code Structure in Java



401
Response

!Important

- Every line that needs to be executed should be inside a class.
- Declarations and definitions
  - `class HelloWorld` is the class declaration.
  - The main is the point from where all the Java programs start its execution.
- Just as sentences in English must be terminated with a period.
- Just as sentences in English can span several lines, so can the statements in Java.
-

401
Response

Doubts

Do I have to put Main() Method in every class to run?

No, We can have multiple classes in Java, to run we need only one main method

401
Response

!Important

Did you know? The `System.out.println()` displays text in the current line of the console and then move the cursor to the beginning of the next line.

System.out.print()	System.out.println()
Prints the argument only, without adding any space or new line after the printing is done	Prints the argument passed to it and then also prints a new line
The next print statement starts printing immediately adjacent to the end of the last printed statement	The next statement after println() is printed in a new line

401
Response

Quiz

Can we execute a java program without a main method?

401
Response

Quiz

Can we execute a java program without a main method?

https://stackoverflow.com/questions/15173474/can-we-execute-a-java-program-without-a-main-method

Yes, we can execute a java program without a main method by using a static block

```java
package com.thetestingacademy;

public class NoMainMethod3 {
    static{
        System.out.println("class without a main method");
        System.exit( status: 0);
    }
    //we can execute a java program without a main method (works until Java 1.6 version).
    //Java 7 and newer versions don't allow this because JVM checks the presence of the main method
}
```

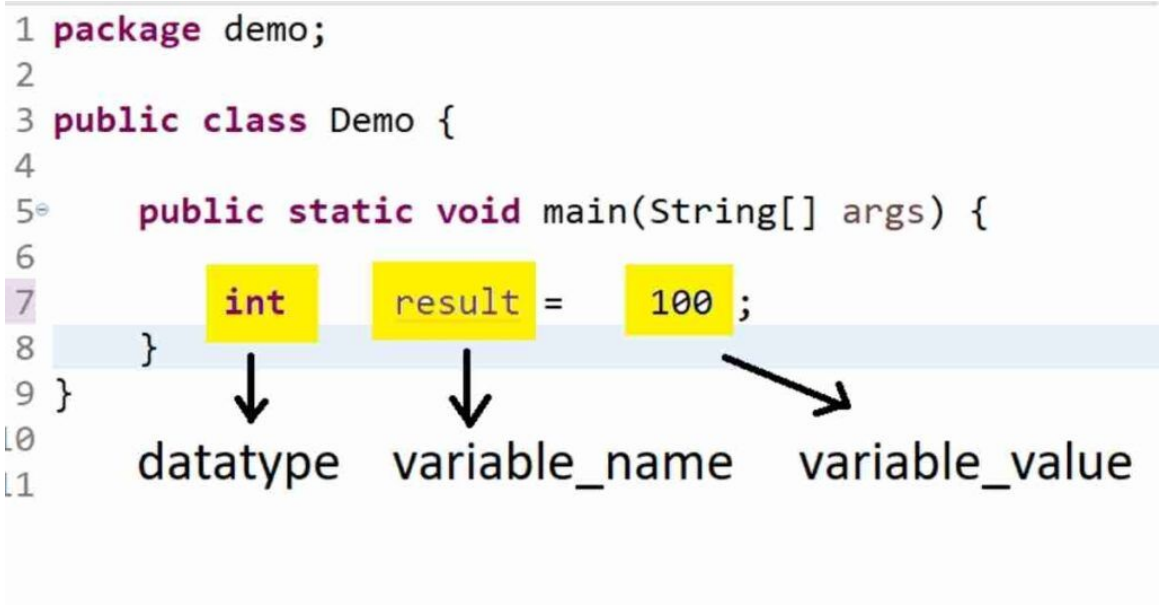
What you can Say in Main Method?

Do something
Do Some Things Again and Again
Do some under this condition
Loop and Loop Again

401
Response

Variables

A variable is a container (storage area) used to hold data.
Each variable should be given a unique name (identifier).



Variables

A variable is a storage location paired with an associated symbolic name (an identifier), which contains some known or unknown quantity of information referred to as a value.

Variables in Java are strongly typed; thus they all must have a data type declared with their identifier

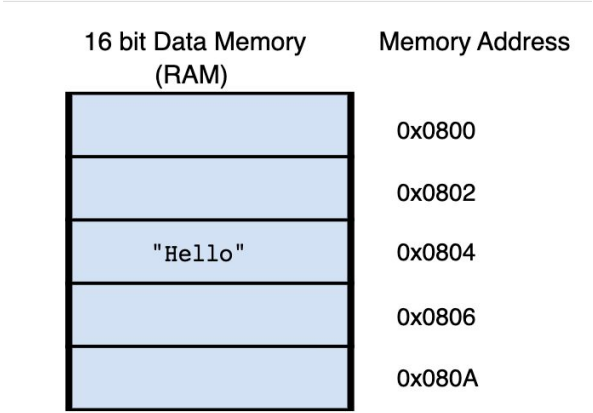
There are three rules to create an identifier:

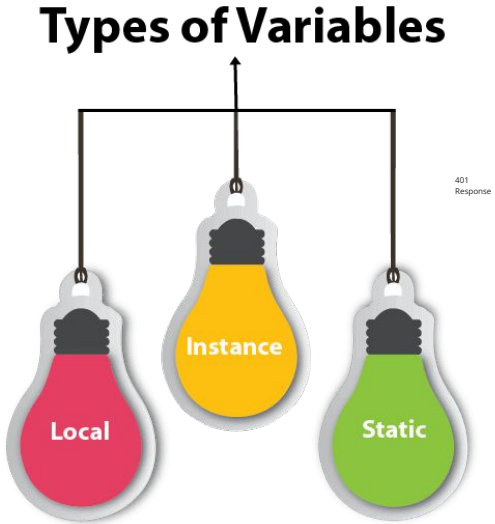
Characters from A to Z, as well as their lowercase counterparts, can be used.
Numbers from 0-9 can be used.
Special characters that can be used are $ (the dollar sign) and _ (underscore).

401
Response

Where are the variables stored?

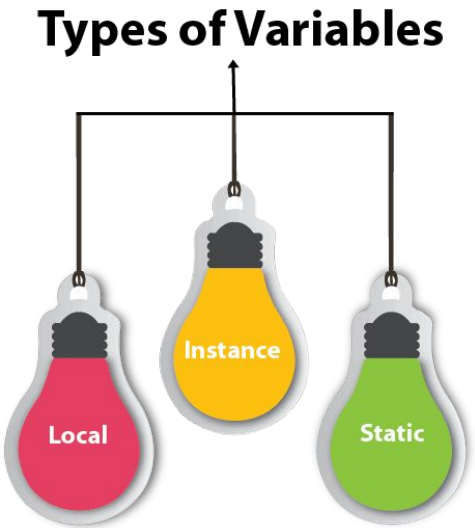
Variables, once declared in the program, are allocated space in memory.
Memory itself is organized as a long collection of one byte after another.





Variables

local variable
instance variable
static variable



401
Response

Sr. No.	Local variables	Instance variables	Static variables
1.	Variables declared within a method are local variables.	An instance variable is declared inside a class but outside of any method or block.	Static variables are declared inside a class but outside of a method starting with a keyword static.
2.	The scope of the local variable is limited to the method it is declared inside.	An instance variable is accessible throughout the class.	The static variable is accessible throughout the class.
3.	A local variable starts its lifetime when the method is invoked.	The object associated with the instance variable decides its lifetime.	The static variable has the same lifetime as the program.
4.	Local variable is accessible to all the objects of the class.	Instance variable has different copies for different objects.	Static variables only have one single copy of the entire class.
5.	Used to store values that are required for a particular method.	Used to store values that are needed to be accessed by different methods of the class.	Used for storing constants.

Data Types

Data types are declarations for variables

There are 2 types of Data Types :

Primitive Data types : to store simple values These are the data types of fixed size.

	DATA TYPE	RANGE OF VALID VALUES	MEMORY VOLUME
integer	byte	from -128 to 127	1 byte
	short	from -32768 to 32767	2 bytes
	int	from -2147483648 to 2147483647	4 bytes
	long	from -9223372036854775808 to 9223372036854775807	8 bytes
floating point	float	from -3.4E + 38 to 3.4E + 38	4 bytes
	double	from -1.7E + 308 to 1.7E + 308	8 bytes
symbols	char	from 0 to 65536	2 bytes
logical	boolean	true or false	For value of this type 1 bit is enough, but in reality memory isn't provided by such portions, so variables of this type may be packed by virtual machine in different ways

Data Types

Non-Primitive Data types : to store complex values

These are of variable size & are usually declared with a ‘new’ keyword.

Eg : String, Arrays

```
String name = new String("Aman");
int[] marks = new int[3];
marks[0] = 97;
marks[1] = 98;
marks[2] = 95;
```

401
Response

Data Types

```
// A short type is a 16 bit signed integer
short age; // Stores from +32,767 to -32,768

// An int type is a 32 bit signed integer
int age; // Stores from +2,147,483,647 to -2,147,483,648

// A long type is a 64 bit signed integer
long age; // Stores from +9,223,372,036,854,775,807 to
-9,223,372,036,854,775,808
```

These are the data types of fixed size.

401
Response

Data Types

```
float radius1 = 0.000000000000863872078f;
System.out.println(radius1);
double radius = 0.00000000000863872078;
System.out.println(radius);
```

401
Response

Data Types

```java
public class Type_Char {
    public static void main(String[] args) {
        char letter = 'A';
        System.out.println(letter);

        char number_as_text = '6';
        System.out.println(number_as_text);

        char number = 65;
        System.out.println(number);
    }
}
```

401
Response

Declaring multiple variables in a single line

```
// Group Declarations

int age, height;
char letter, alpha;
double radium, area;


// Group initializations
int age = 10, height = 5;
double radius = 2.34, area = 4.55;
```

401
Response

Operations in Java

Symbols	Arithmetic operators
+	add
-	subtract
/	divide
*	multiply
%	modulus division
++	post and pre-increment
--	post and pre-decrement

Operator precedence

Rule of BODMAS, which says; Brackets, Order, Division, Multiplication, Addition, and then Subtraction.

(10 - 4) + 3 *4

200 / 50 + (3 * 4)

10+3-4

401
Response

Java Math class

```
System.out.println("2 raised to the power 3 is " + Math.pow(2, 3));
        System.out.println("Exponent squared is " + Math.exp(2));
        System.out.println("The square root of 16 is " + Math.sqrt(16));
        System.out.println("The cube root of 27 is " + Math.cbrt(27));
```

401
Response

Comparative operators

Symbols	Comparative operators
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
!=	Not equal to

Boolean operators

Symbols	Boolean operators
!	Boolean NOT
&&	Boolean AND
||	Boolean OR
^	Boolean exclusive XOR

Problem Statement

$$\sqrt[3]{(x^2+y^2-|z|)}$$

401
Response

Problem Statement

- First, compute the respective powers of two floating-point variables x and y.
- Then Add them after taking powers.
- Then, compute the absolute value of floating-point z.
- Subtract this from the above-computed addition value.
- Now take Cube Root of the answer.
- Use inbuilt functions to calculate this expression

401
Response

Quiz

- (2+3)-10 * 5;

401
Response

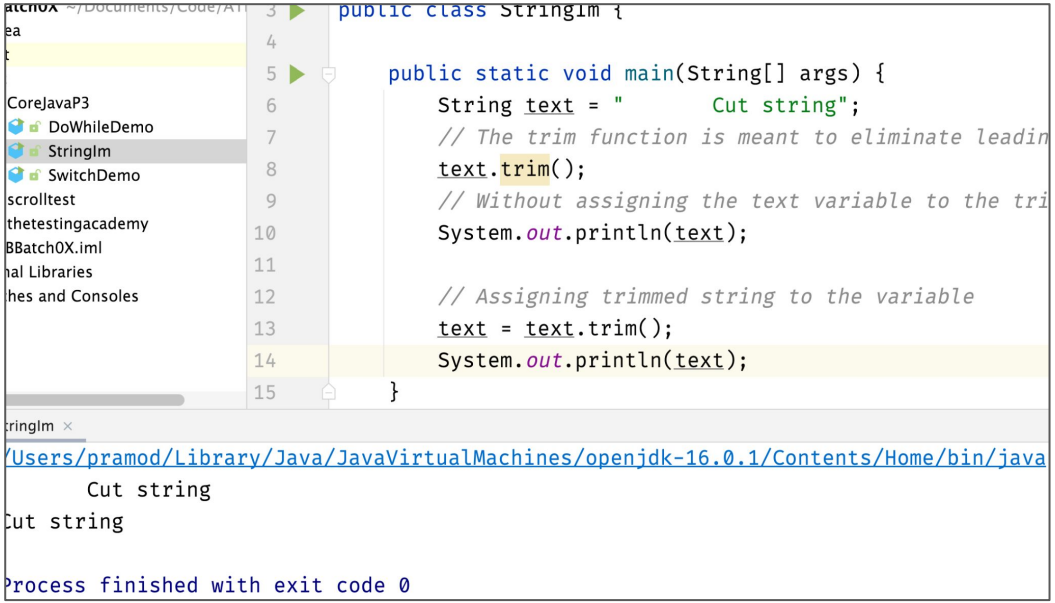
String Class

Strings are immutable non-primitive data types in Java.
Once a string is created it's value cannot be changed
if we wish to alter its value then a new string with a new value has to be created.

```
public class String5 {

    public static void main(String[] args) {

        System.out.println("String demo");
        String name = "Pramod";
        String new_name = "Amit";

        name = new_name;
        System.out.println(name);
```

String Immutability

Keep in mind that String objects are immutable, i.e, they cannot be modified once created



String Class

Concatenation

```
String name1 = new String("Aman");
String description = new String("is a good boy.");

String sentence = name1 + description;
System.out.println(sentence);
```

CharAt

```
String name = new String("Aman");
System.out.println(name.charAt(0));
```

401
Response

`equalsIgnoreCase`

Equals Demo

Split String

Substrings

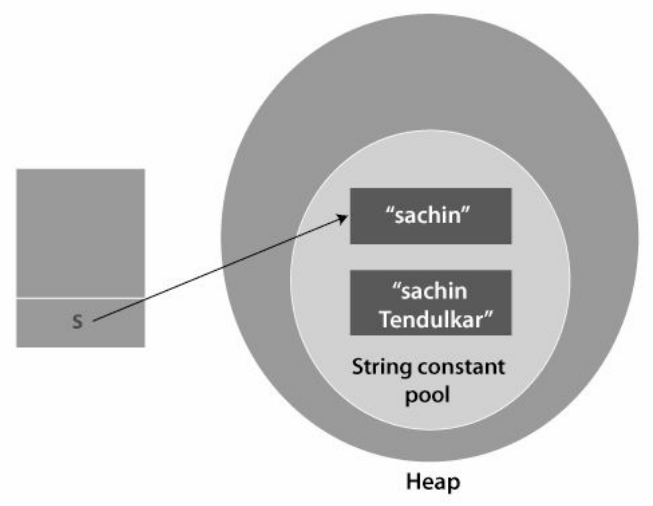
ToUpper

ToLower

String Class

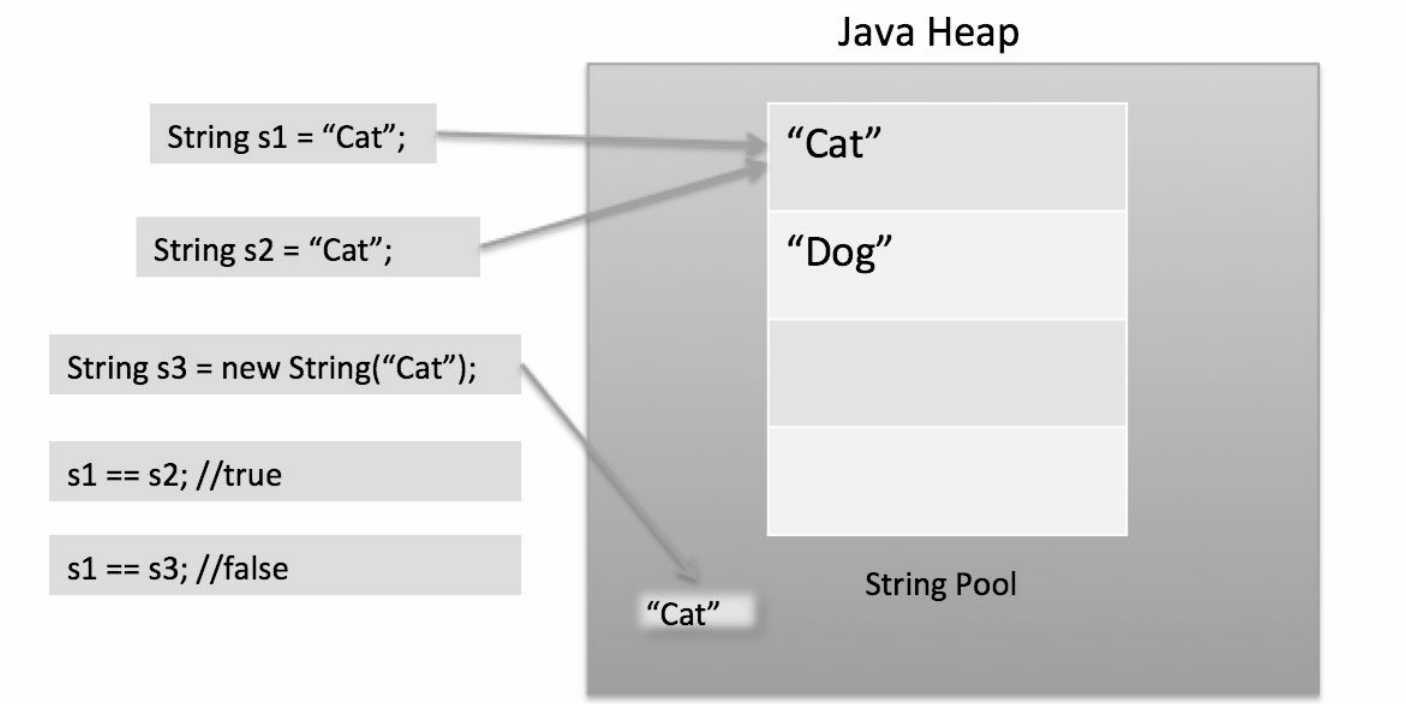
Strings are immutable non-primitive data types in Java.
New String is created

String Pool or String Constant Pool is a special area in **Java Heap memory** where string literals are stored



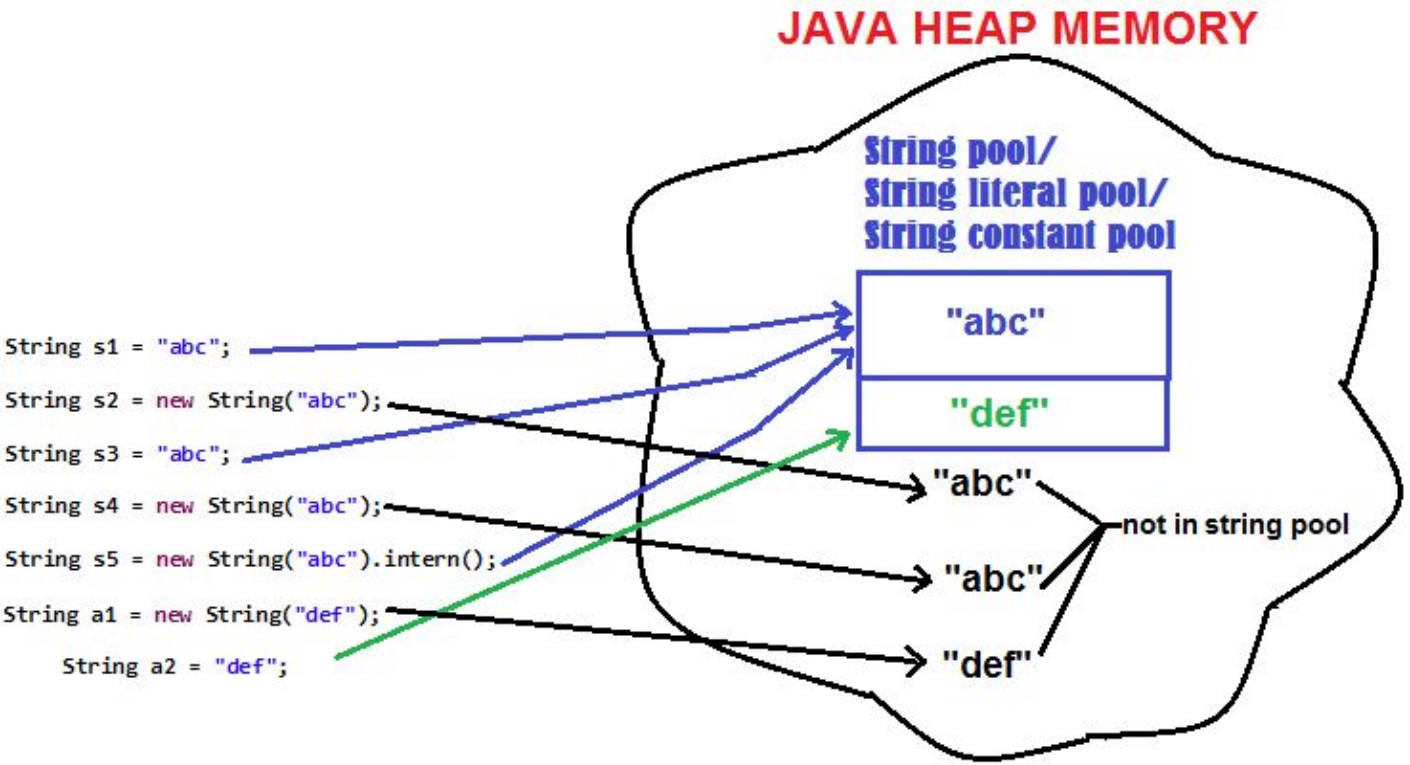
401
Response

String Class

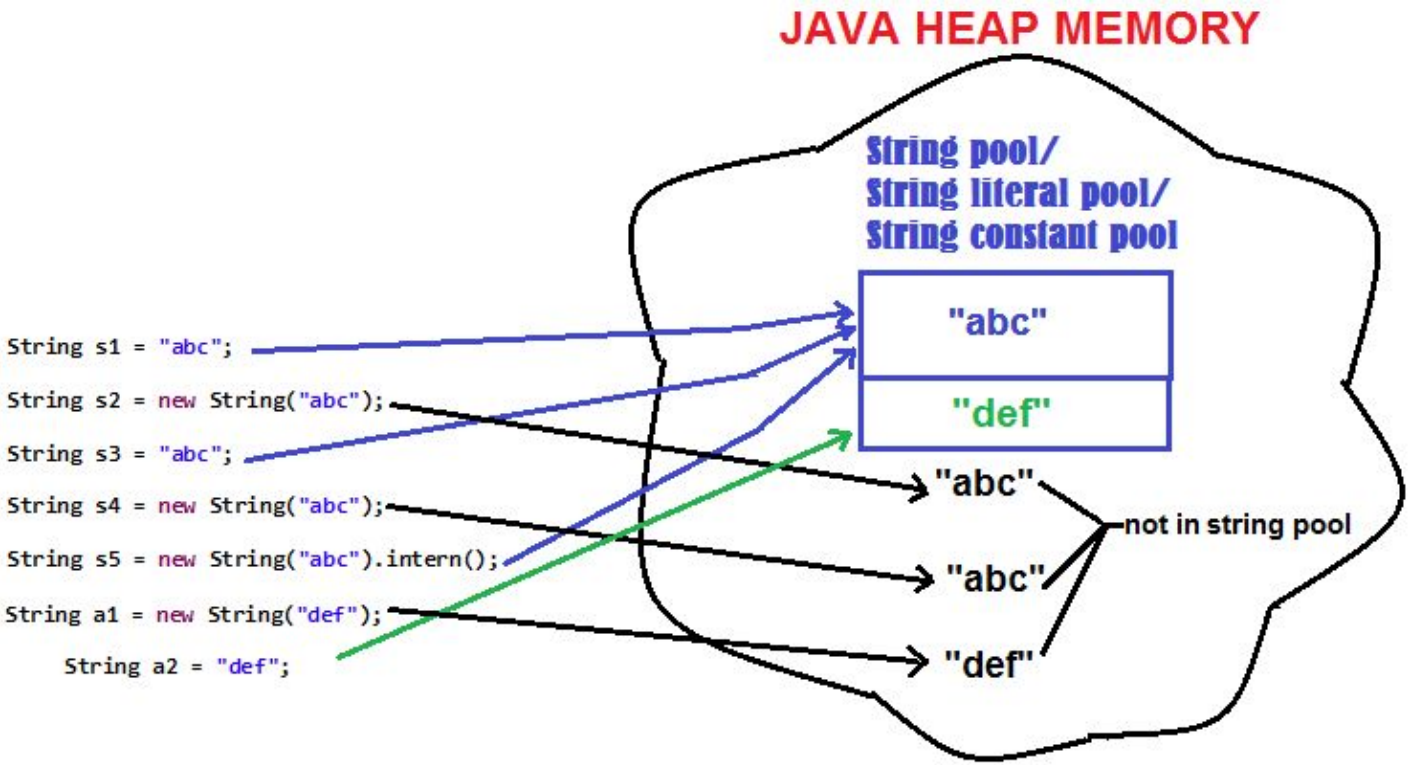


401
Response

String Class



String Interning is a method of storing only one copy of each distinct String Value, which must be immutable.



String Class

Substring

```
String name = new String("AmanAndAkku");
System.out.println(name.substring(0, 4));
```

Length

```
String name = new String("Aman");
System.out.println(name.length());
```

Replace

```
String name = new String("Aman");
System.out.println(name.replace('a', 'b'));
```

401
Response

Problem String

Sample Input

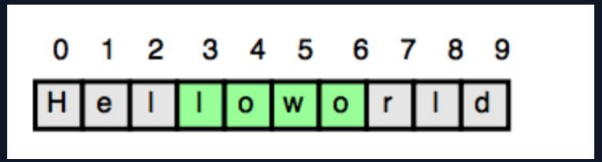
```
Helloworld
3 7
```

Sample Output

```
lowo
```

Explanation

In the diagram below, the substring is highlighted in green:



401
Response

Problem String

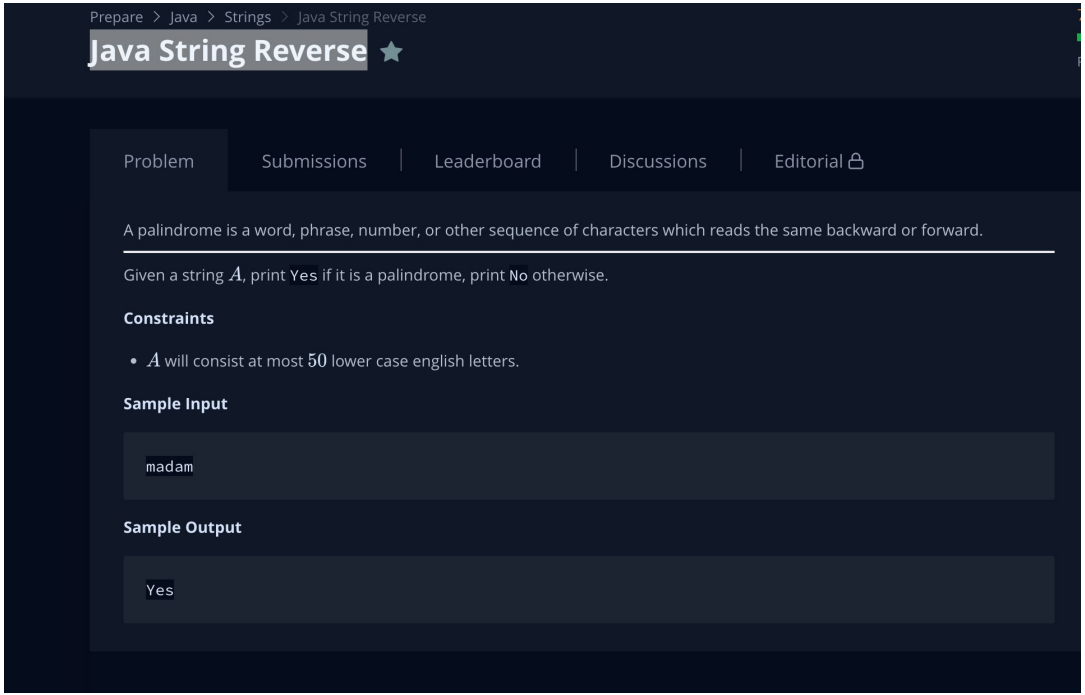
```java
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String S = in.next();
        int start = in.nextInt();
        int end = in.nextInt();
        String result = S.substring(start,end);
            System.out.println(result);

            }
}
```

```java
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner in =  new Scanner(System.in);
        String S = in.next();
         int start = in.nextInt();
         int end = in.nextInt();
         String result = S.substring(start,end);
             System.out.println(result);

             }
}
```

401
Response

https://www.hackerrank.com/challenges/java-substring/problem

Problem String



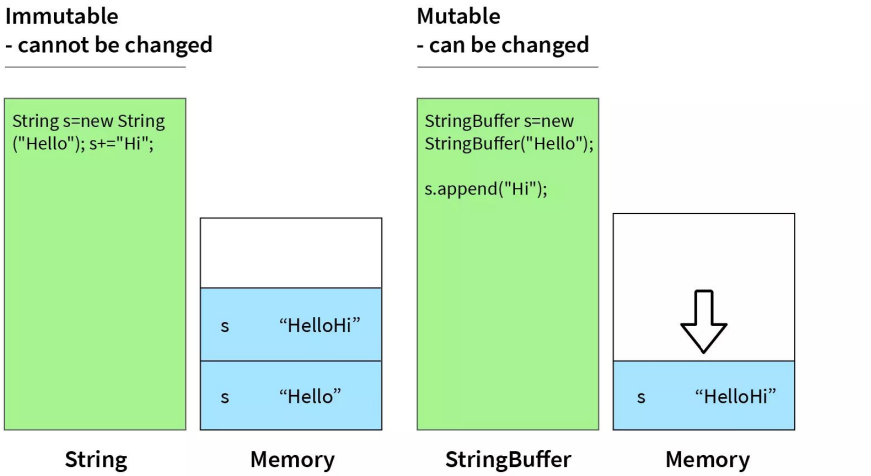
https://www.hackerrank.com/challenges/java-string-reverse/problem

StringBuilder vs StringBuffer

StringBuffer Class	StringBuilder Class
StringBuffer is present in Java.	StringBuilder was introduced in Java 5.
StringBuffer is synchronized. This means that multiple threads cannot call the methods of StringBuffer simultaneously.	StringBuilder is asynchronized. This means that multiple threads can call the methods of StringBuilder simultaneously.
Due to synchronization, StringBuffer is called a thread safe class.	Due to its asynchronous nature, StringBuilder is not a thread safe class.
Due to synchronization, StringBuffer is lot slower than StringBuilder.	Since there is no preliminary check for multiple threads, StringBuilder is a lot faster than StringBuffer.

401
Response

StringBuilder vs StringBuffer



Casting

Casting in java is the assigning values of one type to another

we can only assign values of a number type to another type storing numbers

Implicit casting
This casting is done by java implicitly i.e. on its own. It is assigning smaller values to larger data types.

```
float price = 100.00F;
int gst = 18;
float finalPrice = price + gst;
```

Explicit casting
This casting is done by the programmer. It is assigning larger values to smaller data types.

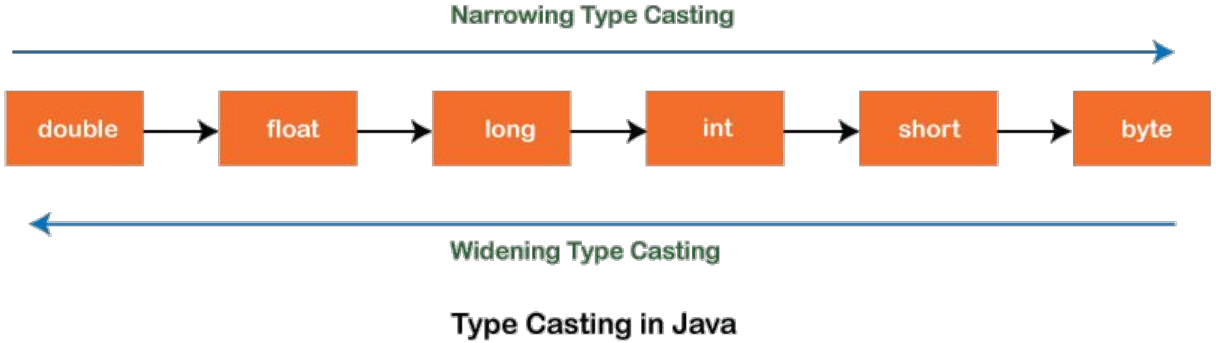
```
int price = 100;
float gst = 18.00F;
int finalPrice = price + (int)gst;
```

Casting : Auto widening and explicit narrowing

The data is implicitly casted from small sized primitive type to big sized primitive type. This is called **auto-widening**

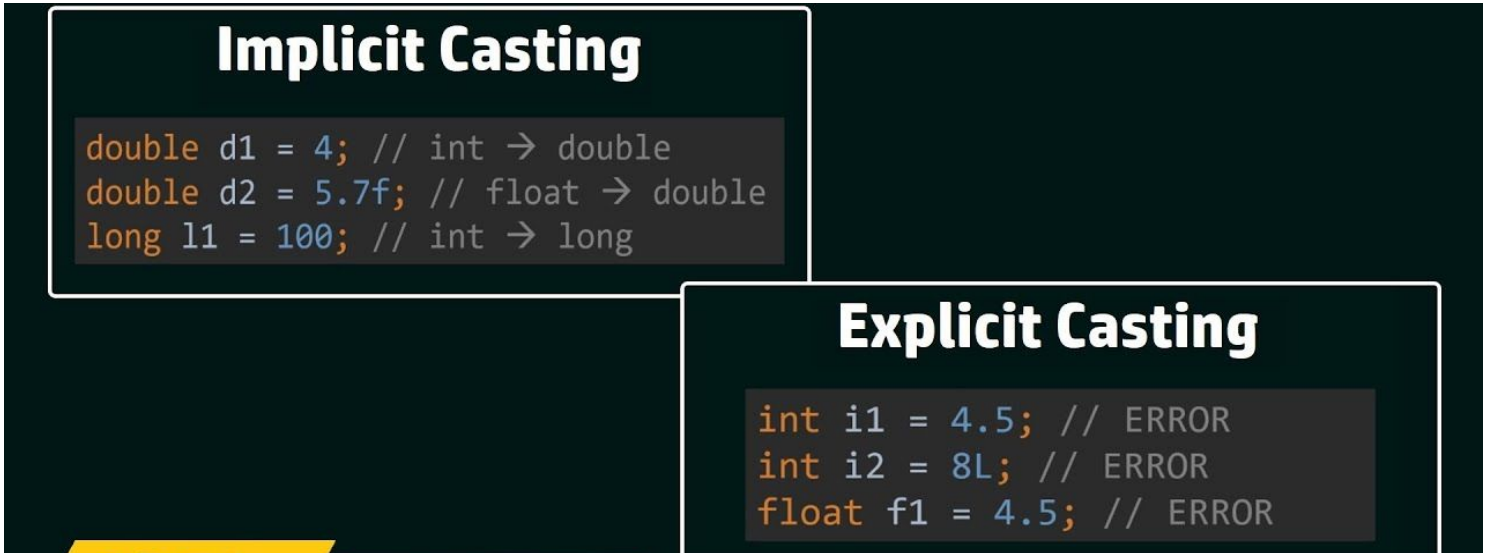
The data is automatically casted from byte to short, short to int, int to long, long to float and float to double.

explicitly convert the data from double to float, float to long, long to int, int to short and short to byte This is called **explicit narrowing.**



Type Casting in Java

Casting : Auto widening and explicit narrowing



401
Response

Constants

A constant is a variable whose value cannot change once it has been assigned

final float pi = 3.14f;

401
Response

Keywords

keywords are also known as reserved words.

abstarct	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

Operators in Java

- Unary Operator,
- Arithmetic Operator,
- Shift Operator,
- Relational Operator,
- Bitwise Operator,
- Logical Operator,
- Ternary Operator and
- Assignment Operator.

401
Response

Operator Type	Category	Precedence
Unary	postfix	*expr++ expr--*
	prefix	*++expr --expr +expr -expr* ~ !
Arithmetic	multiplicative	* / %
	additive	+ -
Shift	shift	<< >> >>>
Relational	comparison	< > <= >= instanceof
	equality	== !=
Bitwise	bitwise AND	&
	bitwise exclusive OR	^
	bitwise inclusive OR	|
Logical	logical AND	&&
	logical OR	||
Ternary	ternary	? :
Assignment	assignment	= += -= *= /= %= &= ^= |= <<= >>= >>>=

Operator Precedence

401 Response

Math Class

Math is an important class in Java that is extensively used and has a lot of interesting functions.
To import - import java.lang.Math;

Some functions include:
Max

```
int a = 10;
int b = 20;
Math.max(a, b);
```

Min

```
int a = 10;
int b = 20;
Math.min(a,b);
```

Random

```
int randomNumber = (int)(Math.random()*100);
```

401
Response

Taking Input

We take input using the **Scanner class** and input various types of data using it.
To import the Scanner class - import java.util.Scanner;

Example :

```
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
float a = sc.nextFloat();
String name = sc.next();
String line = sc.nextLine();
```

401
Response

Conditional Statements ifelse

```
if(boolean) {
   //Code
} else {
   //Code
}
```

401
Response

Conditional Statements ifelse

```
int age = 30;
if(age > 18) {
   System.out.println("This is an adult");
} else {
   System.out.println("This is not an adult");
}
```

Coding Exercise

Coding exercise#
Given a number x, you should check whether it is even or odd.

If it is even, then store "even" in answer.

If it is odd, then store "odd" in answer.

401
Response

Coding Exercise

```
String answer;
        if (x % 2 == 0) {
            answer = "even";
        } else {
            answer = "odd";
        }
        return answer;
```

401
Response

Conditional Statements

Switch

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

401
Response

While Loop

```
int i = 0;
while (i < 5) {
  System.out.println(i);
  i++;
}
```

401
Response

Do While Loop

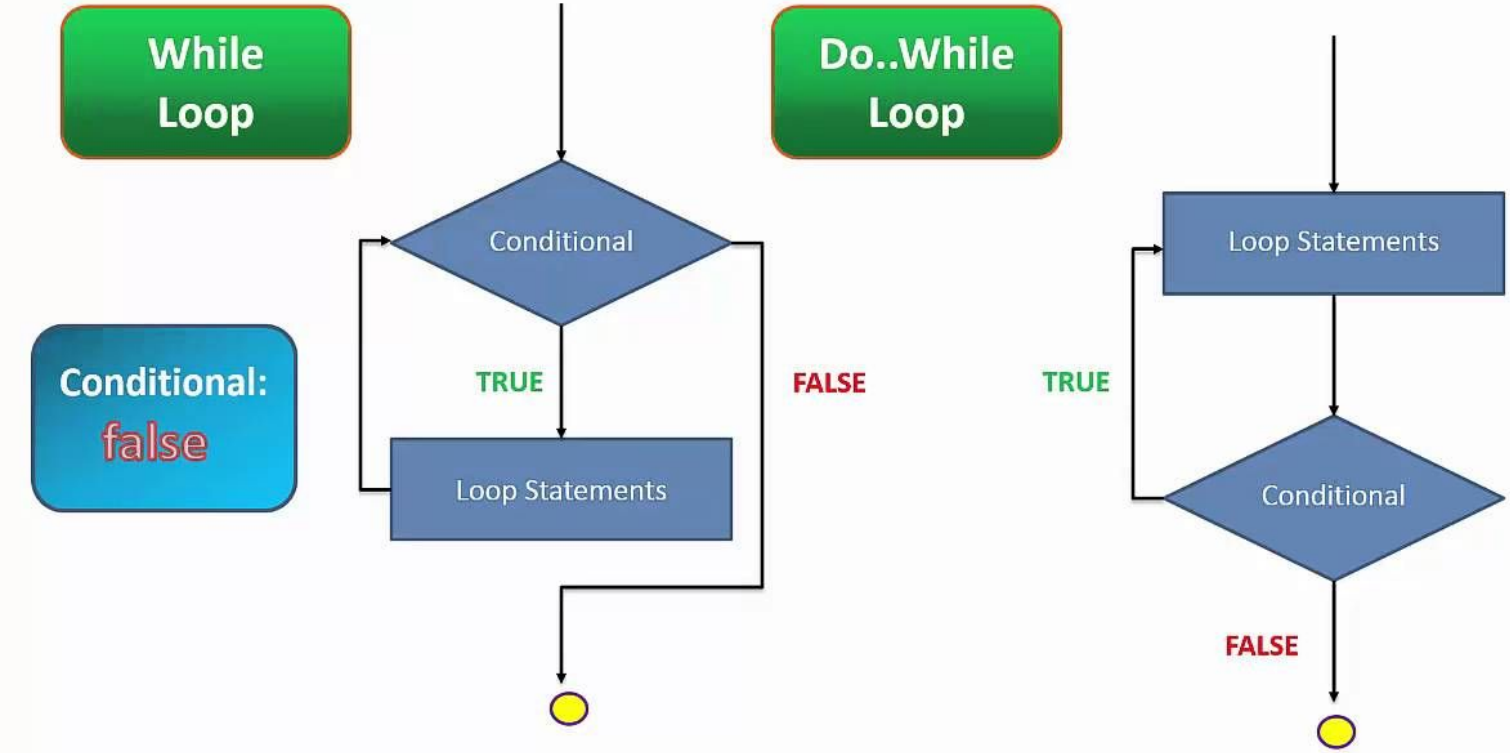
do-while loop is used to iterate a part of the program repeatedly, until the specified condition is true.

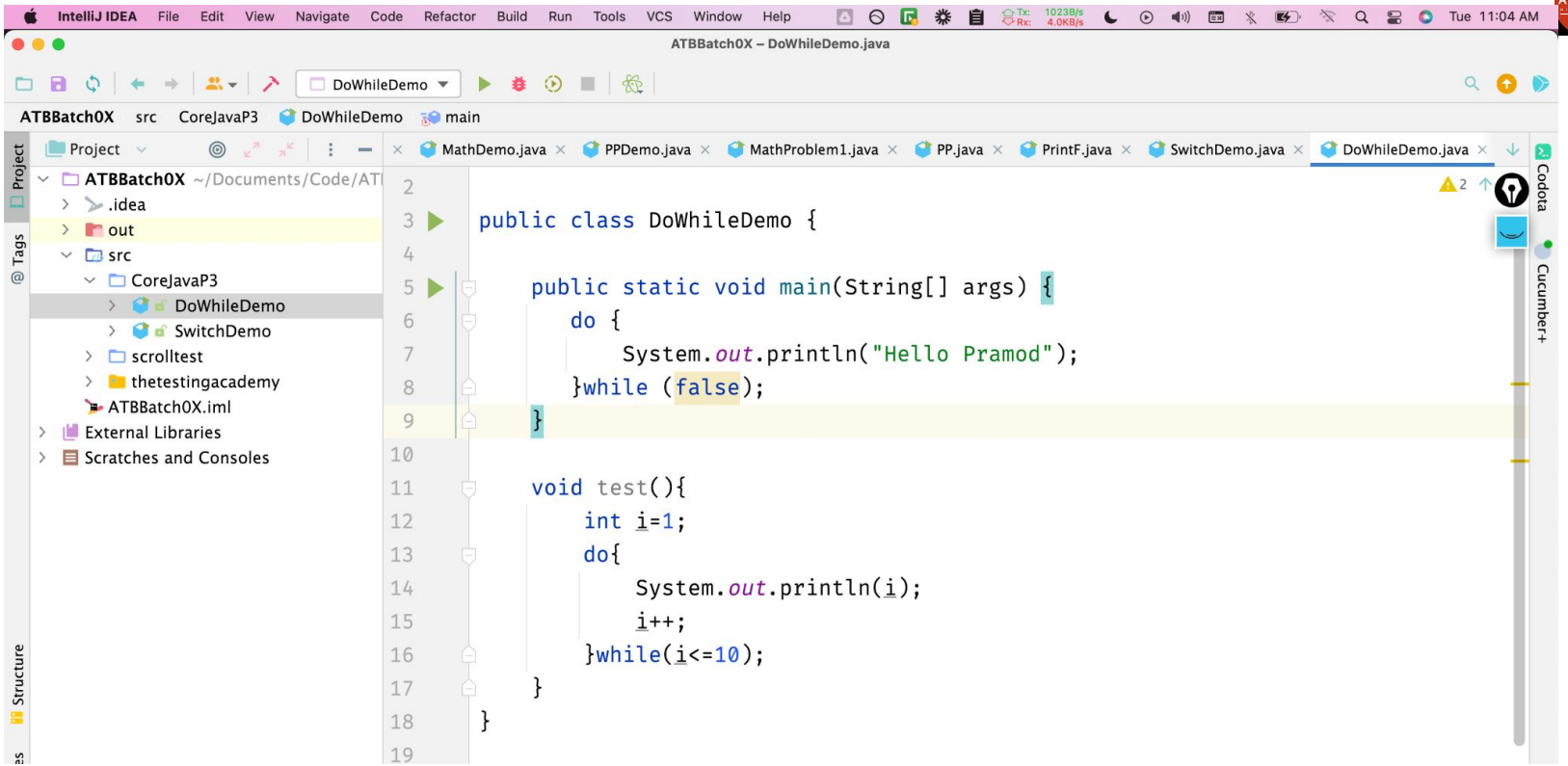
do-while loop is called an **exit control loop**.

401
Response

```
do{
//code to be executed / loop body
//update statement
}while (condition);
```

LOOPS: WHILE VS DO..WHILE




IntelliJ IDEA File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
Tue 11:04 AM
ATBBatch0X – DoWhileDemo.java
DoWhileDemo
ATBBatch0X src CoreJavaP3 DoWhileDemo main
Project
ATBBatch0X ~/Documents/Code/ATI
.idea
out
src
CoreJavaP3
DoWhileDemo
SwitchDemo
scrolltest
thetestingacademy
ATBBatch0X.iml
External Libraries
Scratches and Consoles
MathDemo.java
PPDemo.java
MathProblem1.java
PP.java
PrintF.java
SwitchDemo.java
DoWhileDemo.java
public class DoWhileDemo {
public static void main(String[] args) {
do {
System.out.println("Hello Pramod");
}while (false);
}
void test(){
int i=1;
do{
System.out.println(i);
i++;
}while(i<=10);
}
}
Codota
Cucumber+
@ Tags
Structure

For LOOP

```
for (int i = 0; i <= 10; i = i + 2) {
  System.out.println(i);
}
```

401
Response

For Each Loop

```
String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
for (String i : cars) {
  System.out.println(i);
}
```

401
Response

Break & Continue

```
for (int i = 0; i < 10; i++) {
  if (i == 4) {
    break;
  }
  System.out.println(i);
}
```

401
Response

Break & Continue

```
for (int i = 0; i < 10; i++) {
  if (i == 4) {
    continue;
  }
  System.out.println(i);
}
```

401
Response

Conditional Statements ifelse

Problem
https://www.hackerrank.com/challenges/java-if-else/problem

401
Response

Conditional Statements ifelse

Solution

```
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;

public class Solution {

    private static final Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        int N = scanner.nextInt();
        scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");
        if(N%2!=0){
            System.out.println("Weird");
        }else{
            if(N >= 2 && N <= 5){
                System.out.println("Not Weird");
            }
            if(N >= 6 && N <= 20){
                System.out.println("Weird");
            }
            if(N > 20){
                System.out.println("Not Weird");
            }
        }

        scanner.close();
    }
}
```

401
Response

Arrays

Like a list of elements of the same type i.e. a list of integers,

Creating an Array (method 1) - with new keyword

```
int[] marks = new int[3];
marks[0] = 97;
marks[1] = 98;
marks[2] = 95;
```

Creating an Array (method 2)

```
int[] marks = {98, 97, 95};
```

401
Response

Problem Max in Array

Max in ARRAY

Sample input & output #

Input: {4, 10, 12, 17, 11}

Output: 17

401
Response

Functions / Methods

A function is a block of code which takes some input,
performs some operations and returns some output.
The functions stored inside classes are called methods.
The function we have used is called main.

Problem statement

Write a static method `checkSum` that adds two integer arguments and returns 0, 1 or 2.

- Takes two integers `one` and `two` in its input. Arguments are *pass by value* to the method.
- Has a `check` variable whose value gets updated as explained below.
- **Adds** `num1` and `num2`, and *checks* if their **sum** is **less** than **100** in which case
  - Sets the value of the `check` variable to **0**.
- If **sum** is **greater** than **100**
  - Sets the value of the `check` variable to **1**.
- If **sum** is **equal** to **100**
  - Sets the value of the `check` variable to **2**.
- In the end, it will `return` the **check** variable.

```
public static int checkSum(int one, int two) {
        //Write your code here
        //Declare the necessary variable

        int check;
        int sum = one + two;
        if (sum < 100) {
            check = 0;
        } else if (sum > 100) {
            check = 1;
        } else {
            check = 2;
        }

        //Change the return variable as well

        return check;
    }
}
```

Problem

Sample input#

var = 3557

Sample output#

sum = 3 + 5 + 5 + 7 = 20

Problems

Todo

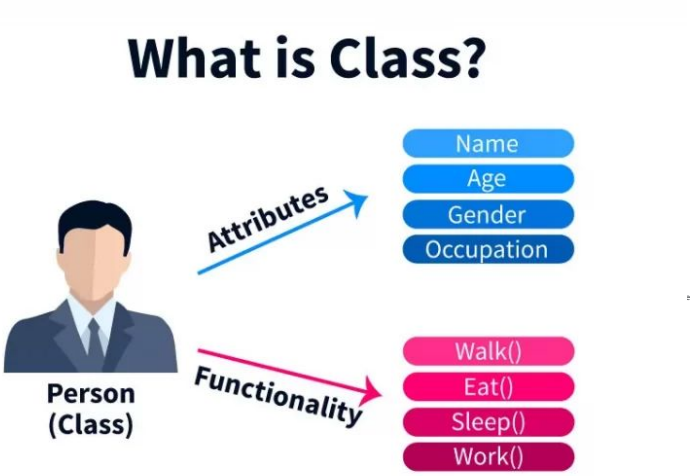
problem of sum
max array
Function with return check variable

401
Response

```
package CoreJavaP3.Loops;


public class SumOfDigit {

    public static void main(String[] args) {


        int input = 3453;

        int sum= 0;

        while(input > 0){

            sum = sum + input%10;

            input = input/10;

        }

        System.out.println(sum);

    }

}
```

Class

A class is a group of objects which have common properties. A class can have some properties and functions (called methods).
The class we have used is Main.

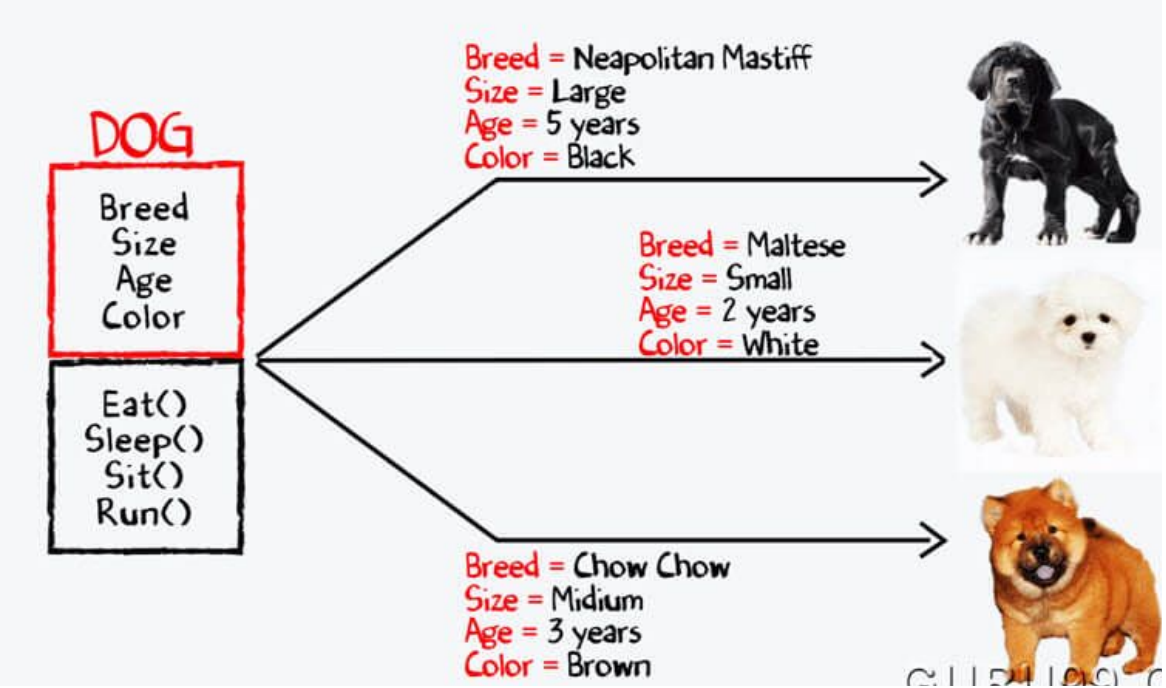


MyClass.java × MyClass2.java × NoMainMethod

```java
package com.thetestingacademy;

public class FunctionsDemo4 {

    public int doubleMe(int a){
        return a*2;
    }
}
```

When a class is created, no memory is allocated

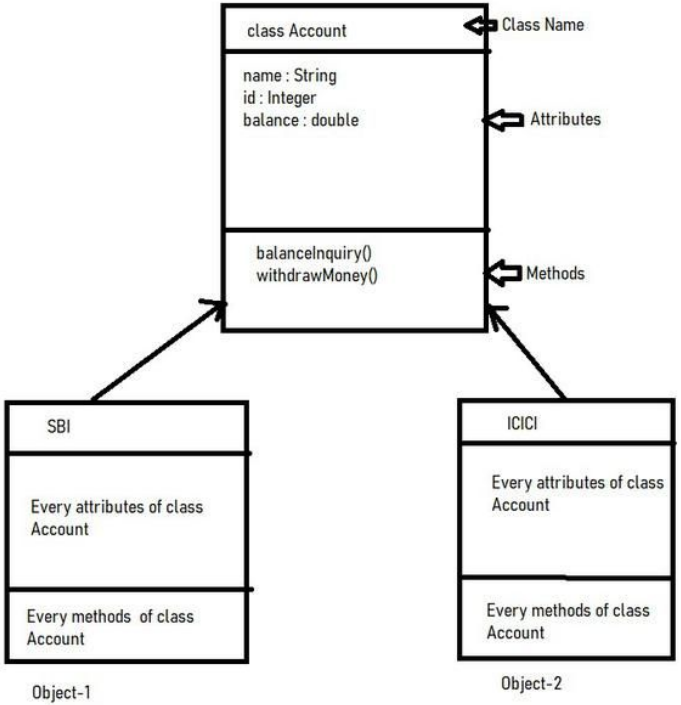
Objects



Object is an instance of a class. All data members and member functions of the class can be accessed with the help of objects

Objects are allocated memory space whenever they are created.

Class & Objects



401
Response

Class

Every class in Java can be composed of the following elements:

- **`fields, member variables or instance variables`** - These are variables that hold data specific to a particular object. For each object of a class, there is one field.
- **`member methods or instance methods`** - These perform operations on objects and are defined within the class.
- **`static or class fields`** - These fields are common to any object within the same class. They can only exist once in the class irrespective of the total number of objects in the class.
- **`static or class methods`** - These methods do not affect a specific object in the class.
- **`inner classes`** - At times a class will be defined within a class if it is a subset of another class. The class contained within another is the **inner** class.

Thanks, for attending Class

I hope you liked it. Say Thanks in Comment :)