

COMPLETE JAVA

1
20+ HOURS

NOTES CODE CERTIFICATE

70+ PROGRAMMING
CHALLENGES



Ex- amazon Microsoft



[Video Link](#)

KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)

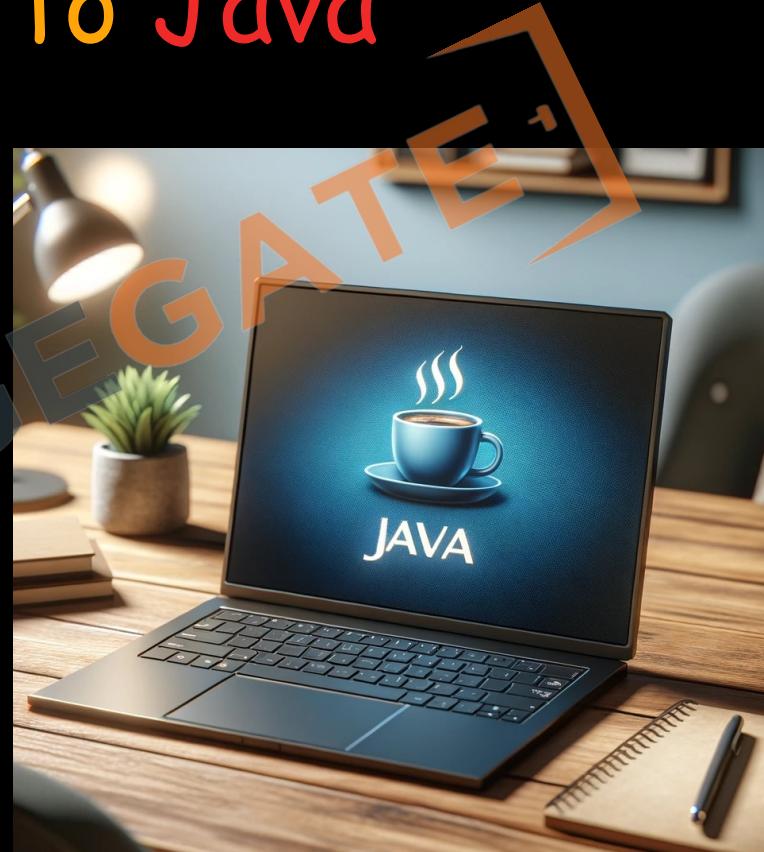


[Sanchit Socket](#)



1. Introduction to Java

1. Why you must learn Java
2. What is a Programming Language
3. What is an Algorithm
4. What is Syntax
5. History of Java
6. Magic of Byte Code
7. How Java Changed the Internet
8. Java Buzzwords
9. Object Oriented Programming





1.1 Why you must learn Java

Trending Technologies: Most Googled Programming Languages Across the World



1. One of the **most popular** language. Java currently runs on **60,00,00,00,000** devices.
2. **Wide Usage** (Web-apps, backend, **Mobile apps**, enterprise software).
3. **High paying** and a **lot of Jobs**
4. **Object Oriented**
5. **Rich APIs and Community Support**



1.2 What is a Programming Language



Humans use natural language (like Hindi/English) to communicate

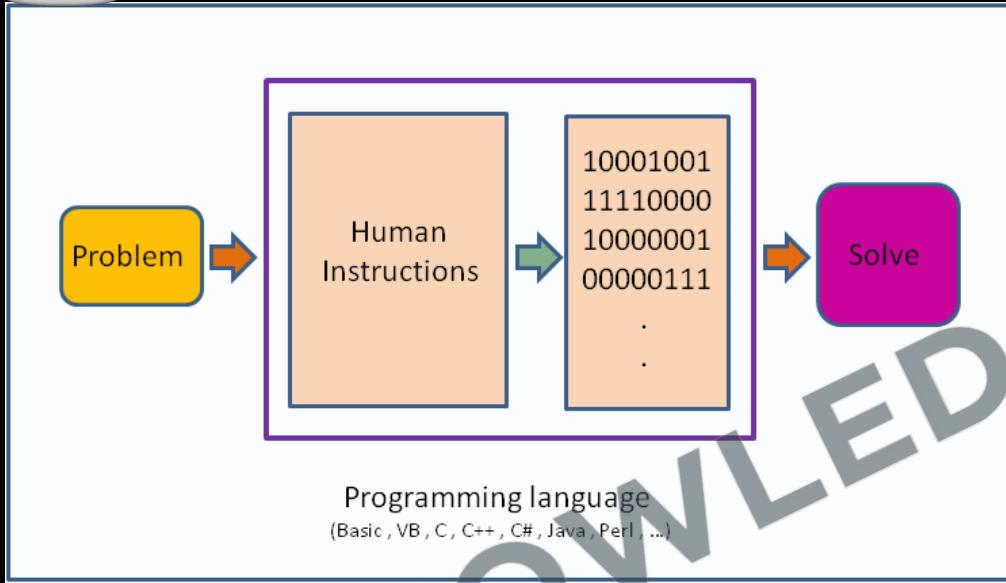


1.2 What is a Programming Language



Computers only understand 0/1 or on/off

1.2 What is a Programming Language



- Giving instructions to a computer
- **Instructions:** Tells computer what to do.
- These instructions are called **code**.
- Human instructions are given in High level languages.

Compiler converts **high level languages** to **low level languages** or **machine code**.



1.3 What is an Algorithm

-STEP BY STEP-

How To Make Tea



put the tea and pour
hot water into the cup



brew the tea for
5 minutes then drain



add sugar/honey/lemon
according to your taste



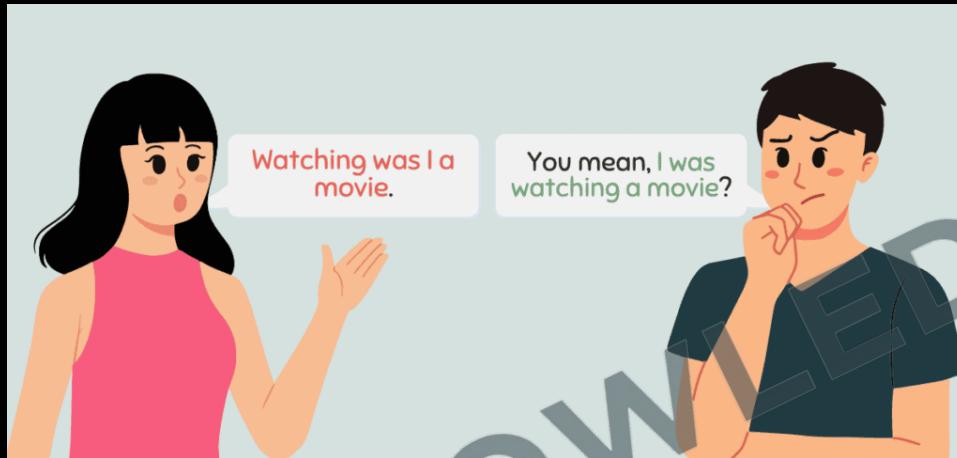
tea is ready
to be enjoyed



1.3 What is an Algorithm



An algorithm is a step-by-step procedure for solving a problem or performing a task.



1.4 What is Syntax

- Structure of words in a sentence.
- Rules of the language.
- For programming exact syntax must be followed.



1.5 History of Java

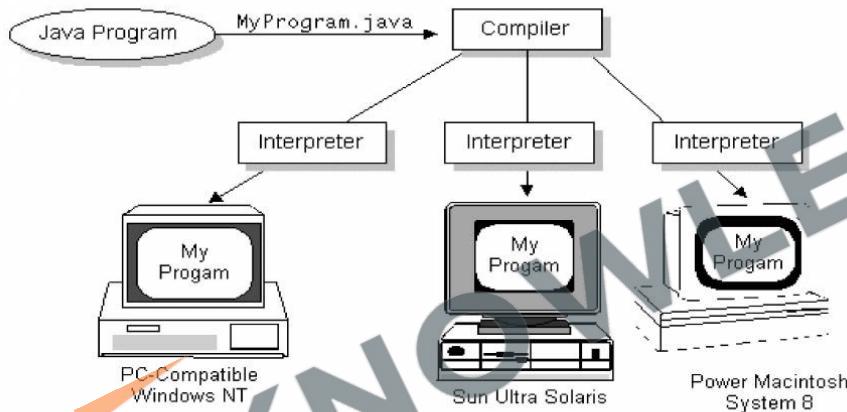


Developed by James Gosling at Sun Microsystems (Early 1990s):
Originally named 'Oak', later renamed Java in 1995.



1.5 History of Java

Write Once, Run Anywhere



First Release (1995): Introduced "Write Once, Run Anywhere" concept with cross-platform compatibility.



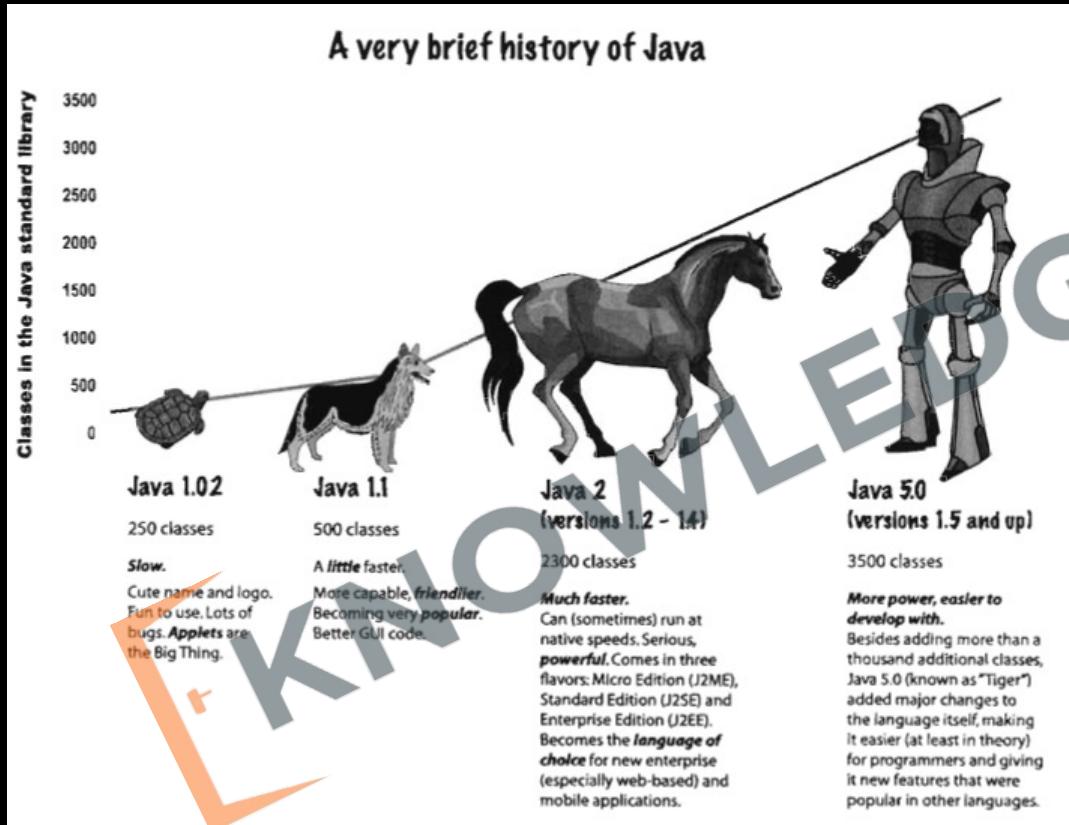
1.5 History of Java



Developed with vision of **backward compatibility**. Should not break with new version release.



1.5 History of Java

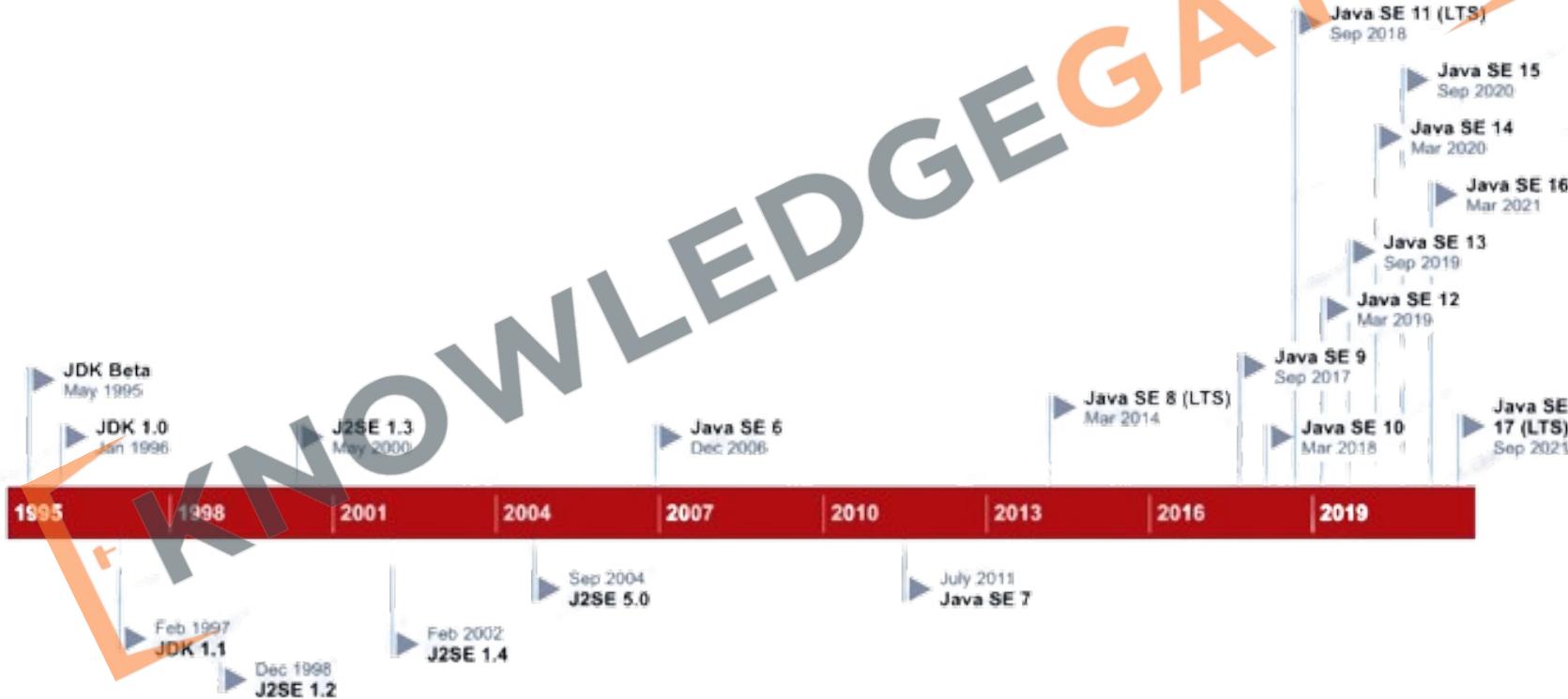


Rapid Growth and Diversification (Late 1990s - 2010s): Expanded from web applets to **server-side applications**; standardized into different editions for various computing platforms.



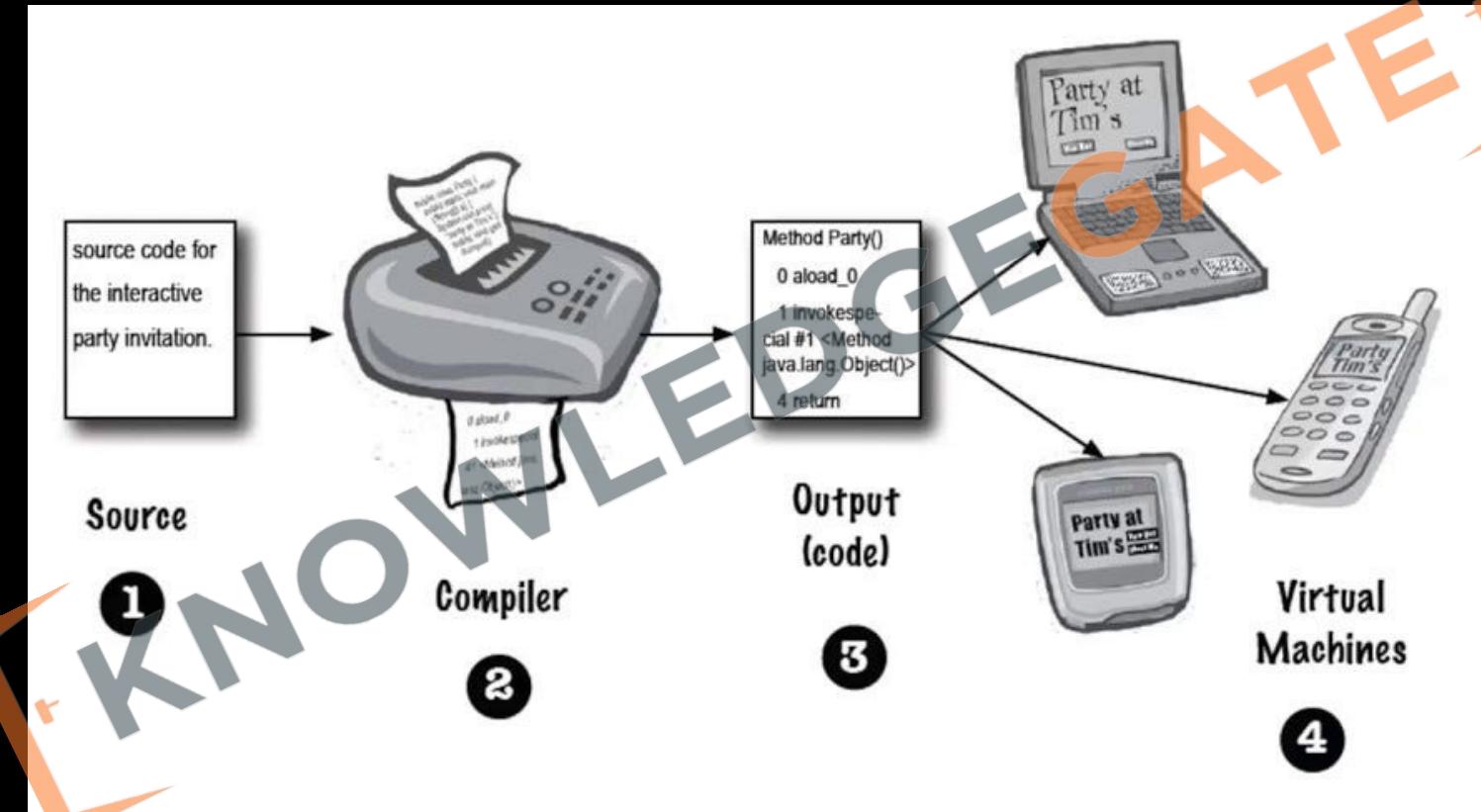
1.5 History of Java

Java Version History





1.6 Magic of Byte Code





1.7 How Java Changed the Internet



Portability with Write Once
Run Anywhere



Security because of Code
running on Virtual Machine



1.8 Java Buzzwords

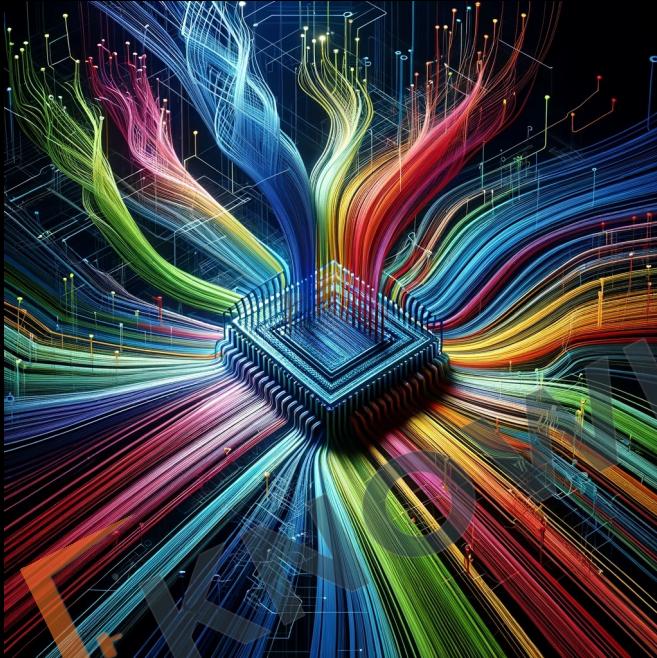


Robust

Java is robust due to its strong memory management, exception handling, and type-checking mechanisms, which help in preventing system crashes and ensuring reliable performance.



1.8 Java Buzzwords



Multithreaded

Multithreading in programming is the ability of a CPU to execute multiple threads concurrently, allowing for more efficient processing and task management.



1.8 Java Buzzwords



Architecture Neutral

Java is architecturally neutral because its compiled code (bytecode) can run on any device with a Java Virtual Machine (JVM), regardless of the underlying hardware architecture.



1.8 Java Buzzwords



Interpreted and High Performance

Java combines high performance with **interpretability**, as its bytecode is interpreted by the Java Virtual Machine (JVM), which employs **Just-In-Time (JIT) compilation** for efficient and fast execution.



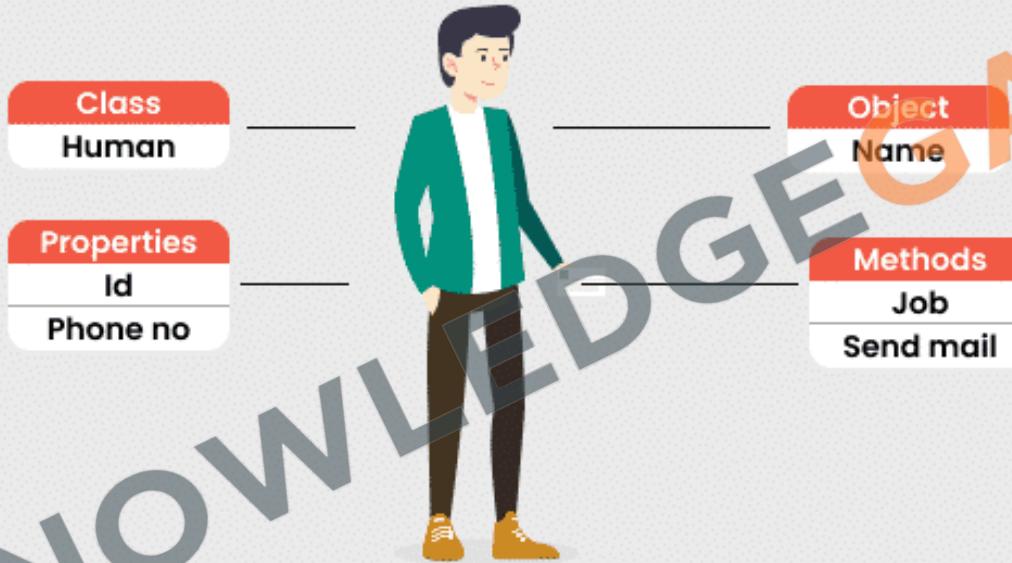
1.8 Java Buzzwords



Distributed

Java is inherently distributed, designed to facilitate network-based application development and interaction, seamlessly integrating with Internet protocols and remote method invocation.

1.9 Object Oriented Programming



Object Oriented Programming (oop)

1.9 Object Oriented Programming



KNOWLEDGE GATE

Revision

1. Why you must learn Java
2. What is a Programming Language
3. What is an Algorithm
4. What is Syntax
5. History of Java
6. Magic of Byte Code
7. How Java Changed the Internet
8. Java Buzzwords
9. Object Oriented Programming



KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)



2. Java Basics

1. Installing JDK
2. First Class using Text Editor
3. Compiling and Running
4. Anatomy of a Class
5. File Extensions
6. JDK vs JVM vs JRE
7. Showing Output
8. Importance of the main method
9. Installing IDE(IntelliJ Idea)
10. Creating first Project





2.1 Installing JDK



Search JDK Download



Make sure to download from the oracle website.



Download the latest version



JAVA DEVELOPMENT KIT

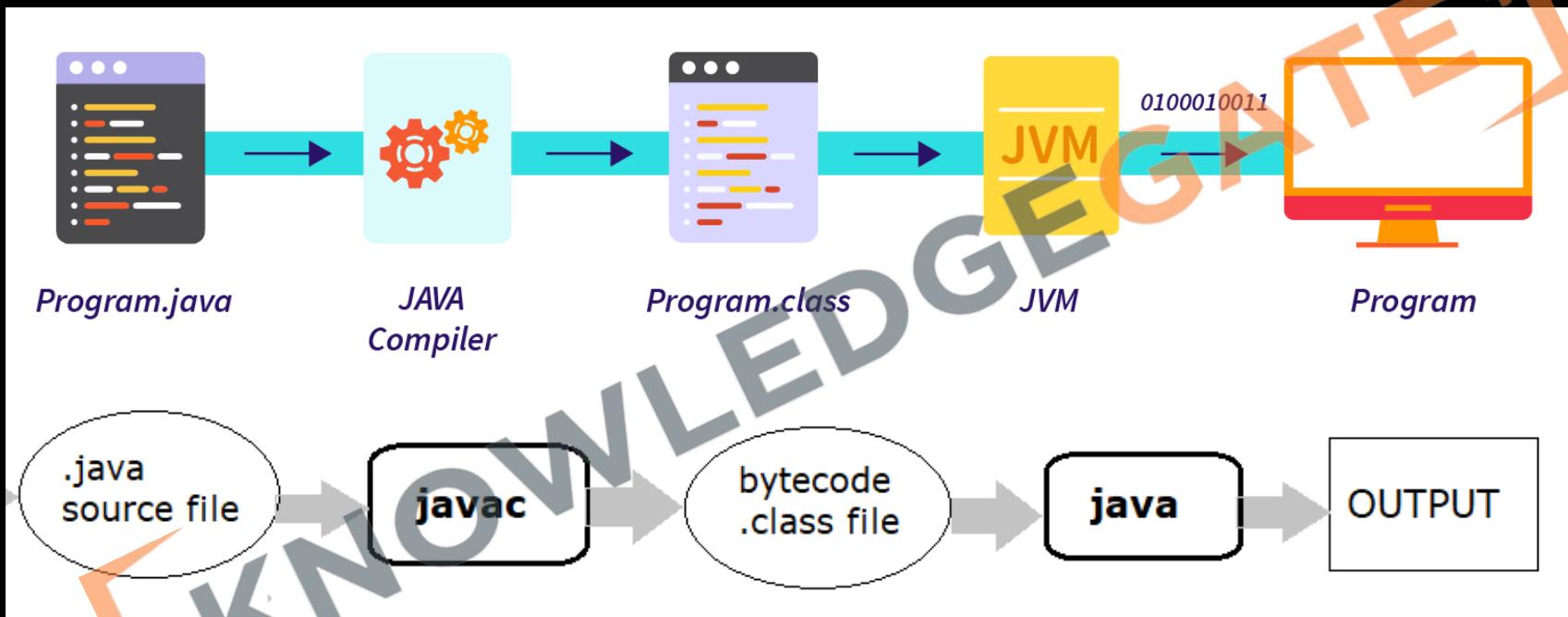


2.2 First Class using Text Editor

```
import java.lang.*;  
  
public class First {  
    public static void main(String[] args) {  
        System.out.println("Welcome to KGCoding");  
    }  
}
```

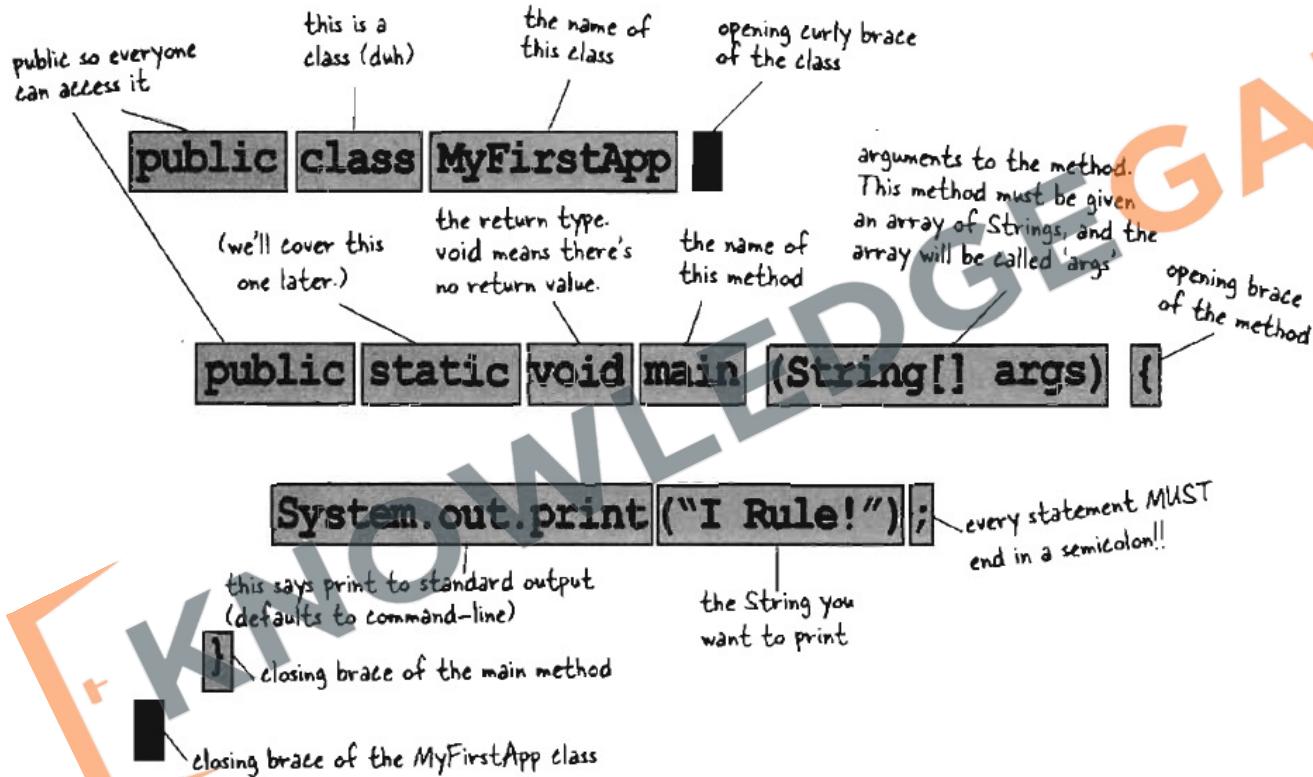


2.3 Compiling and Running





2.4 Anatomy of a Class



The diagram illustrates the anatomy of a Java class named `MyFirstApp`. It shows the code structure with various parts labeled:

- `public`: public so everyone can access it.
- `class`: this is a class (duh).
- `MyFirstApp`: the name of this class.
- `{`: opening curly brace of the class.
- `public static void main`: arguments to the method. This method must be given an array of Strings, and the array will be called 'args'.
- `(String[] args)`: the name of this method.
- `{`: opening brace of the method.
- `System.out.print("I Rule!");`: every statement MUST end in a semicolon!!
- `System.out.print`: this says print to standard output (defaults to command-line).
- `("I Rule!")`: the String you want to print.
- `}`: closing brace of the main method.
- `}`: closing brace of the MyFirstApp class.



2.5 File Extensions

.Java

- Contains Java Source Code
- High Level Human Readable
- Used for Development
- File is editable

.Class

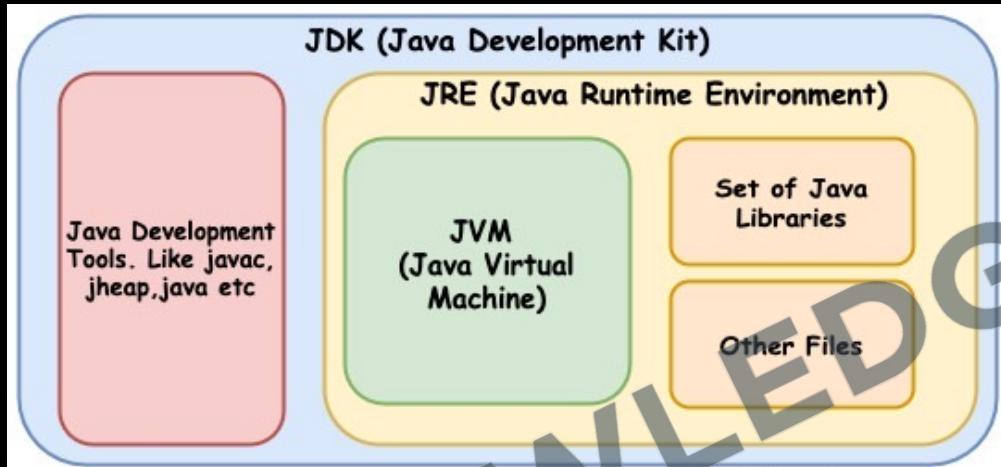
- Contains Java Bytecode
- For consumption of JVM
- Used for Execution
- Not meant to be edited

```
>Main.java x
1 public class Main {
2     public static void main(String[] args) {
3         System.out.println("Hello world!");
4     }
5 }
```

```
java Main
Hello world!
```



2.6 JDK vs JVM vs JRE

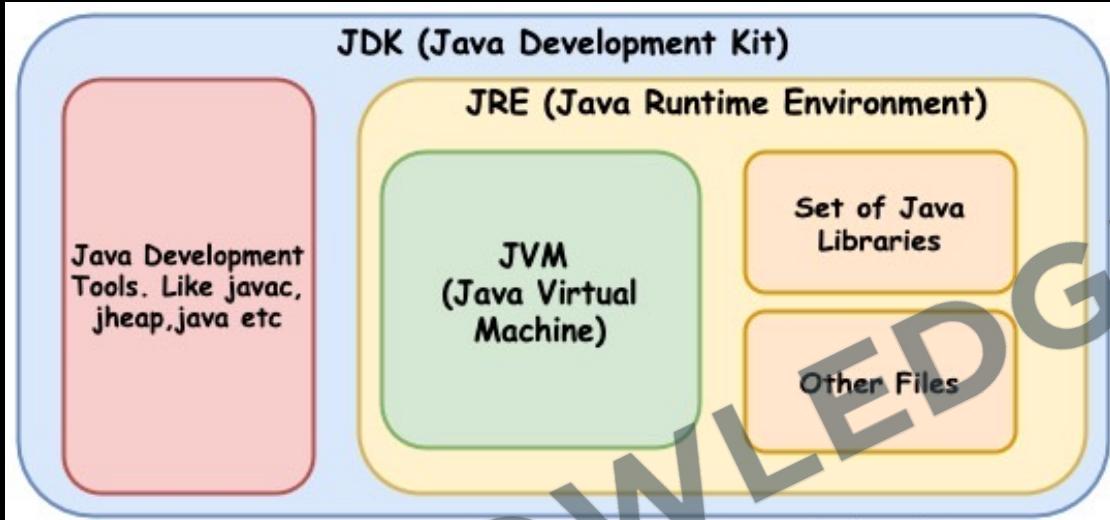


.JDK

- It's a software development kit required to develop Java applications.
- Includes the JRE, an interpreter/loader (Java), a compiler (javac), a doc generator (Javadoc), and other tools needed for Java development.
- Essentially, JDK is a superset of JRE.



2.6 JDK vs JVM vs JRE

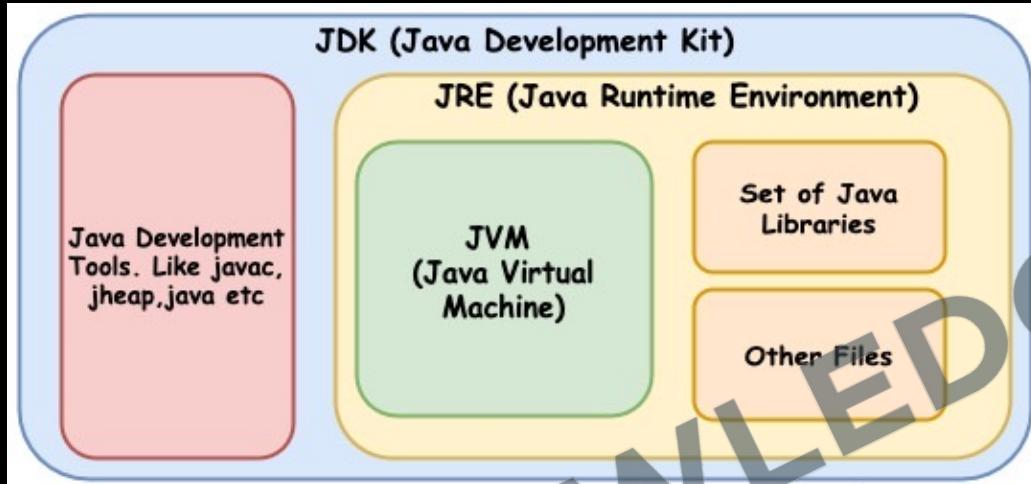


.JRE

- It's a part of the **JDK** but can be downloaded separately.
- Provides the libraries, the **JVM**, and other components to run applications
- Does not have tools and **utilities for developers** like compilers or debuggers.



2.6 JDK vs JVM vs JRE



.JVM

- It's a part of JRE and **responsible for executing** the bytecode.
- Ensures Java's **write-once-run-anywhere** capability.
- Not platform-independent: a different **JVM** is needed for **each type of OS**.

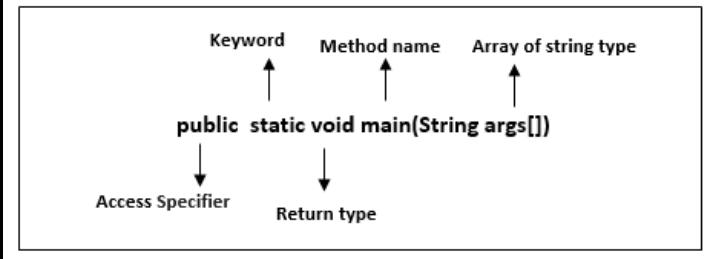


2.7 Showing Output

Example	Result
<pre>System.out.print("one"); System.out.print("two"); System.out.println("three");</pre>	onetwothree
<pre>System.out.println("one"); System.out.println("two"); System.out.println("three");</pre>	one two three
<pre>System.out.println();</pre>	[new line]



2.8 Importance of the main method



- **Entry Point:** It's the **entry point** of a Java program, where the **execution starts**. Without the main method, the **Java Virtual Machine (JVM)** does not know where to begin running the code.
- **Public and Static:** The **main method** must be **public** and **static**, ensuring it's **accessible to the JVM without needing to instantiate the class**.
- **Fixed Signature:** The main method has a **fixed signature**: **public static void main(String[] args)**. Deviating from this signature means the **JVM won't recognize it as the starting point**.

KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)



2.9 What is IDE

1. IDE stands for Integrated Development Environment.
2. Software suite that consolidates basic tools required for software development.
3. Central hub for coding, finding problems, and testing.
4. Designed to improve developer efficiency.





2.9 Need of IDE

1. Streamlines development.
2. Increases productivity.
3. Simplifies complex tasks.
4. Offers a unified workspace.
5. IDE Features
 1. Code Autocomplete
 2. Syntax Highlighting
 3. Version Control
 4. Error Checking

```
@Composable
fun MessageCard(msg: Message) {
    Row(modifier = Modifier.padding(all = 8.dp)) {
        Image(
            painter = painterResource(R.drawable.android_studio_logo),
            contentDescription = "Profile Picture",
            modifier = Modifier
                .size(45.dp)
        )
        Spacer(modifier = Modifier.width(8.dp))
        Column (Modifier
            .background(color = Color.White)) {
            Text(text = msg.author, color = Color.Black)
            Spacer(modifier = Modifier.height(1.dp))
            Text(text = msg.body, color = Color.Black)
        }
    }
}
```



2.9 Installing IDE(IntelliJ Idea)



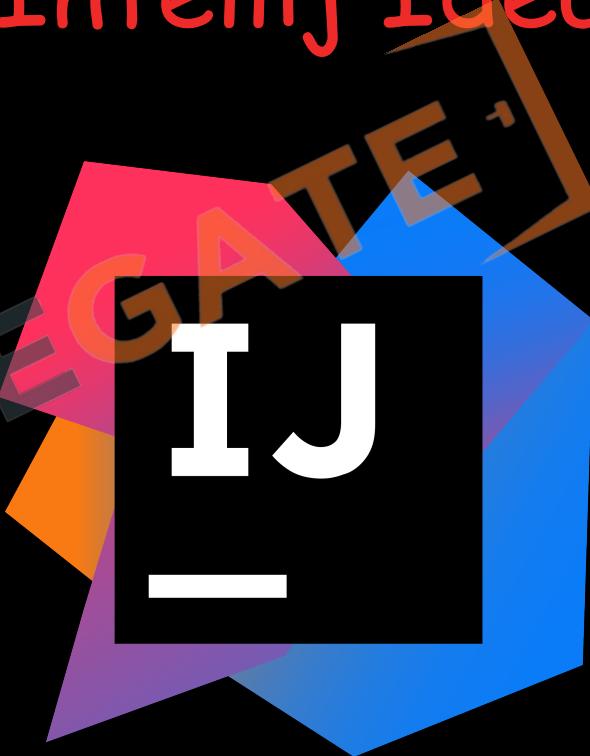
Search IntelliJ IDEA



Make sure to download the
community Version.



Keep Your Software up to
date.





2.9 Installing IDE(IntelliJ Idea)

We're committed to giving back to our wonderful community, which is why IntelliJ IDEA Community Edition is completely free to use



IntelliJ IDEA Community Edition

The IDE for Java and Kotlin enthusiasts

Download

.dmg

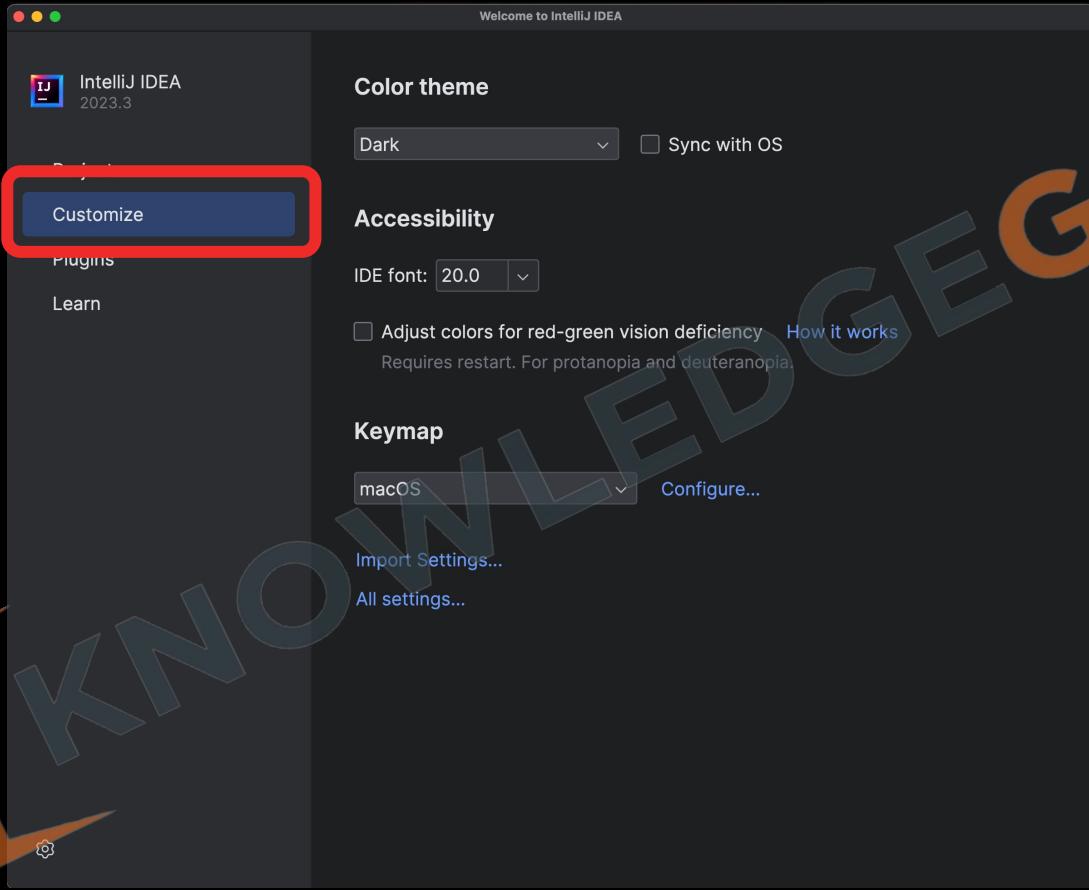
Free, built on open source



Select an installer for Intel or Apple Silicon

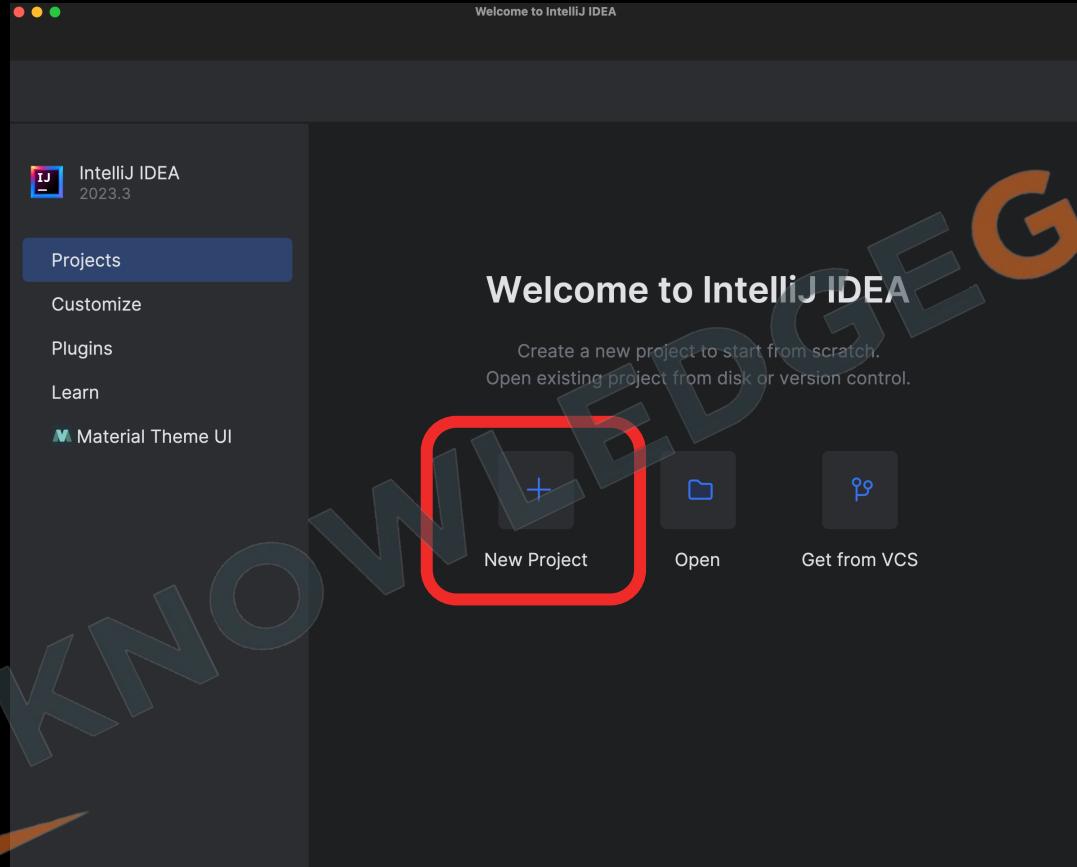


2.9 Installing IDE(IntelliJ Idea)



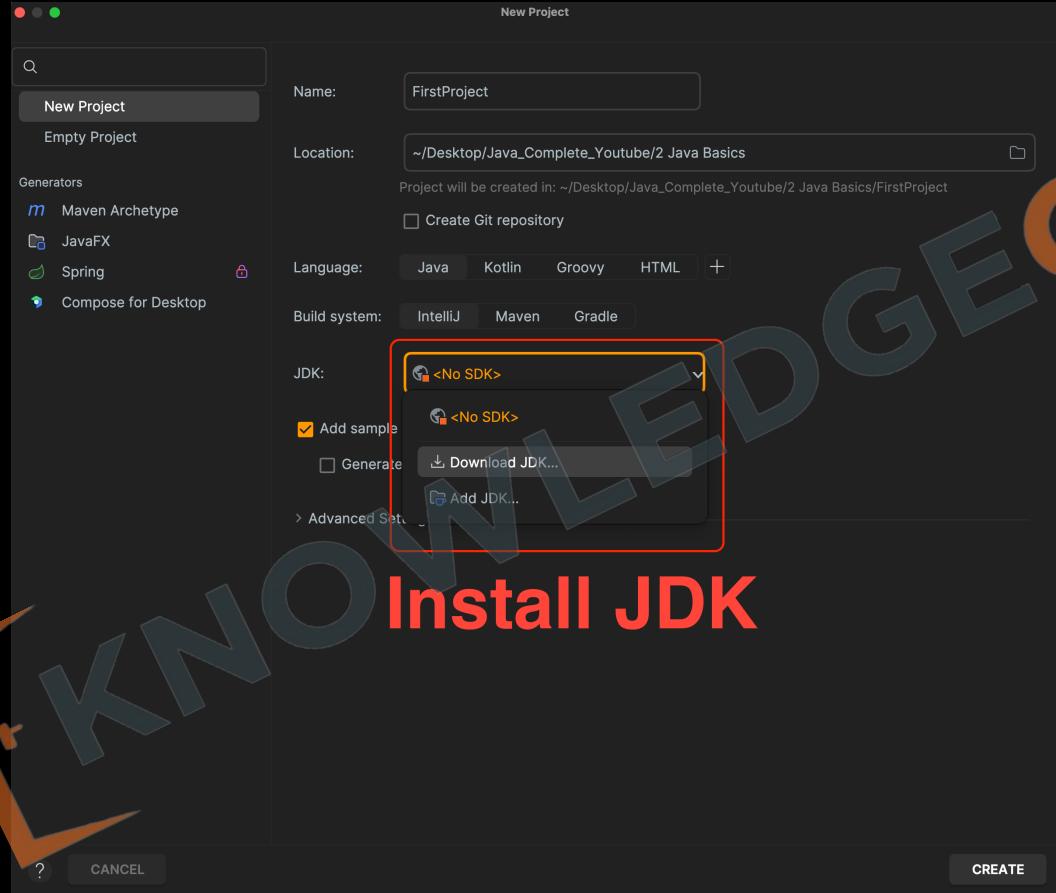


2.9 Installing IDE(IntelliJ Idea)



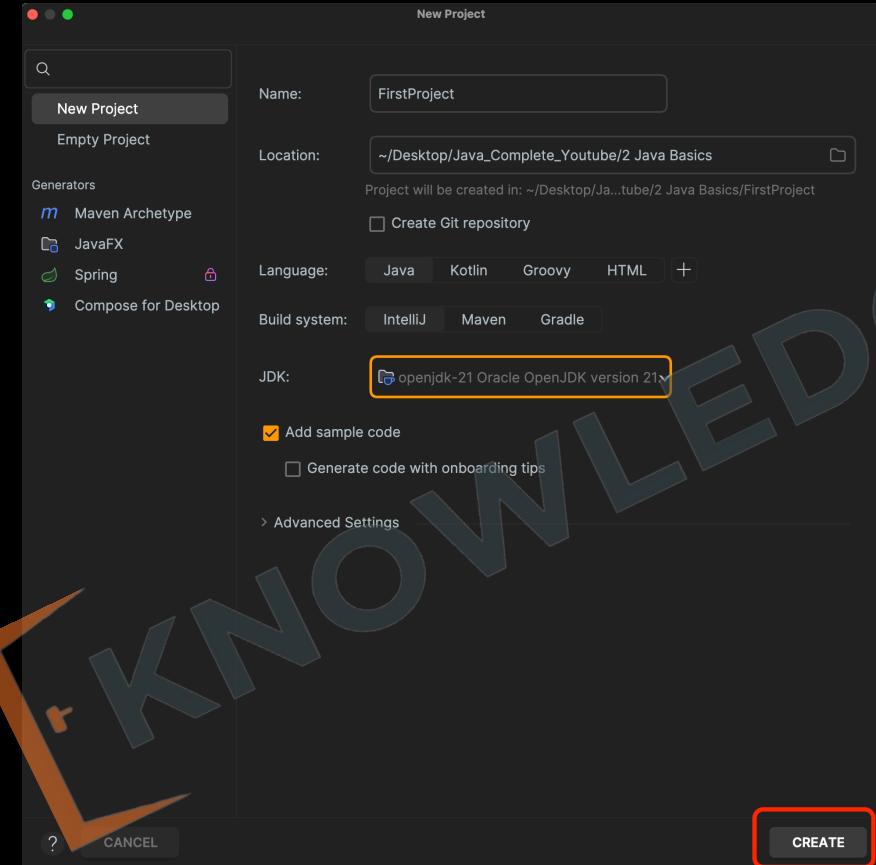


2.9 Installing IDE(IntelliJ Idea)





2.10 Creating first Project





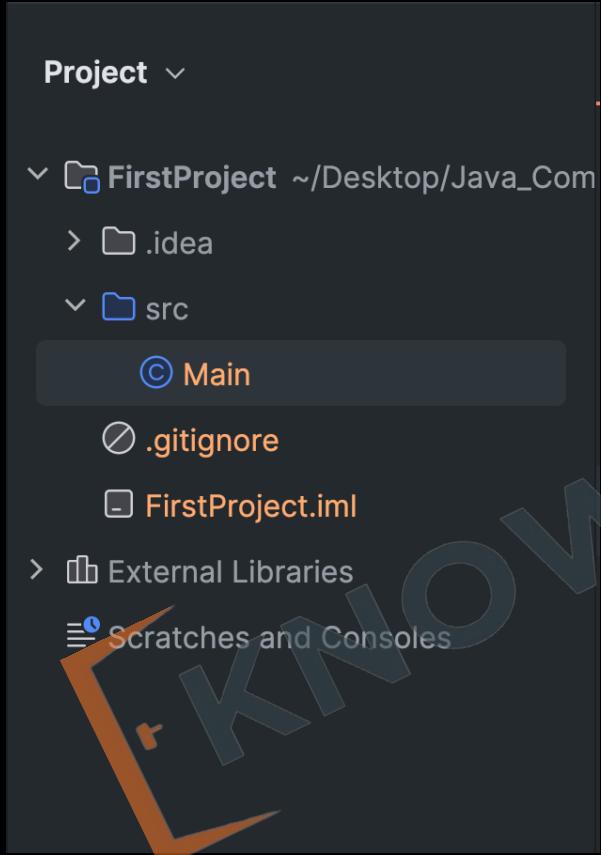
2.10 First Java Class

© Main.java ×

```
1▶ public class Main {  
2▶     public static void main(String[] args) {  
3▶         System.out.println("Hello world!");  
4▶     }  
5▶ }
```



2.10 Project Structure





Revision

1. Installing JDK
2. First Class using Text Editor
3. Compiling and Running
4. Anatomy of a Class
5. File Extensions
6. JDK vs JVM vs JRE
7. Showing Output
8. Importance of the main method
9. Installing IDE(IntelliJ Idea)
10. Creating first Project





CHALLENGE

Tasks:

1. Create a class to output “good morning” using a **text editor** and check output.
2. Create a **new Project** in Intelij Idea and output “subscribe” on the console.
3. Show the following patterns:





Practice Exercise

Java Basics

Answer in True/False:

1. Computers understand **high level languages** like Java, C.
2. An **Algorithm** is a set of **instructions** to accomplish a task.
3. Computer is smart enough to **ignore incorrect syntax**.
4. Java was first released in **1992**.
5. Java was named over a person who made good **coffee**.
6. **ByteCode** is platform independent.
7. **JDK** is a part of **JRE**.
8. It's optional to declare **main** method as **public**.
9. **.class** file contains machine code.
10. **println** adds a new line at the end of the line.





Practice Exercise

Java Basics

Answer in True/False:

1. Computers understand **high level languages** like Java, C. False
2. An Algorithm is a set of **instructions** to accomplish a task. True
3. Computer is smart enough to **ignore incorrect syntax**. False
4. **Java** was first released in **1992**. False
5. **Java** was named over a person who made good **coffee**. False
6. **ByteCode** is platform independent. True
7. **JDK** is a part of **JRE**. False
8. It's optional to declare **main** method as **public**. False
9. **.class** file contains machine code. False
10. **println** adds a new line at the end of the line. True



KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)



3 Data Types, Variables & Input

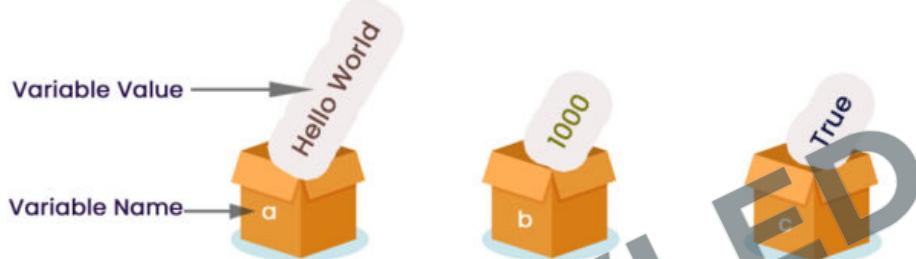
1. Variables
2. Data Types
3. Naming Conventions
4. Literals
5. Keywords
6. Escape Sequences
7. User Input
8. Type Conversion and Casting





3.1. What are Variables?

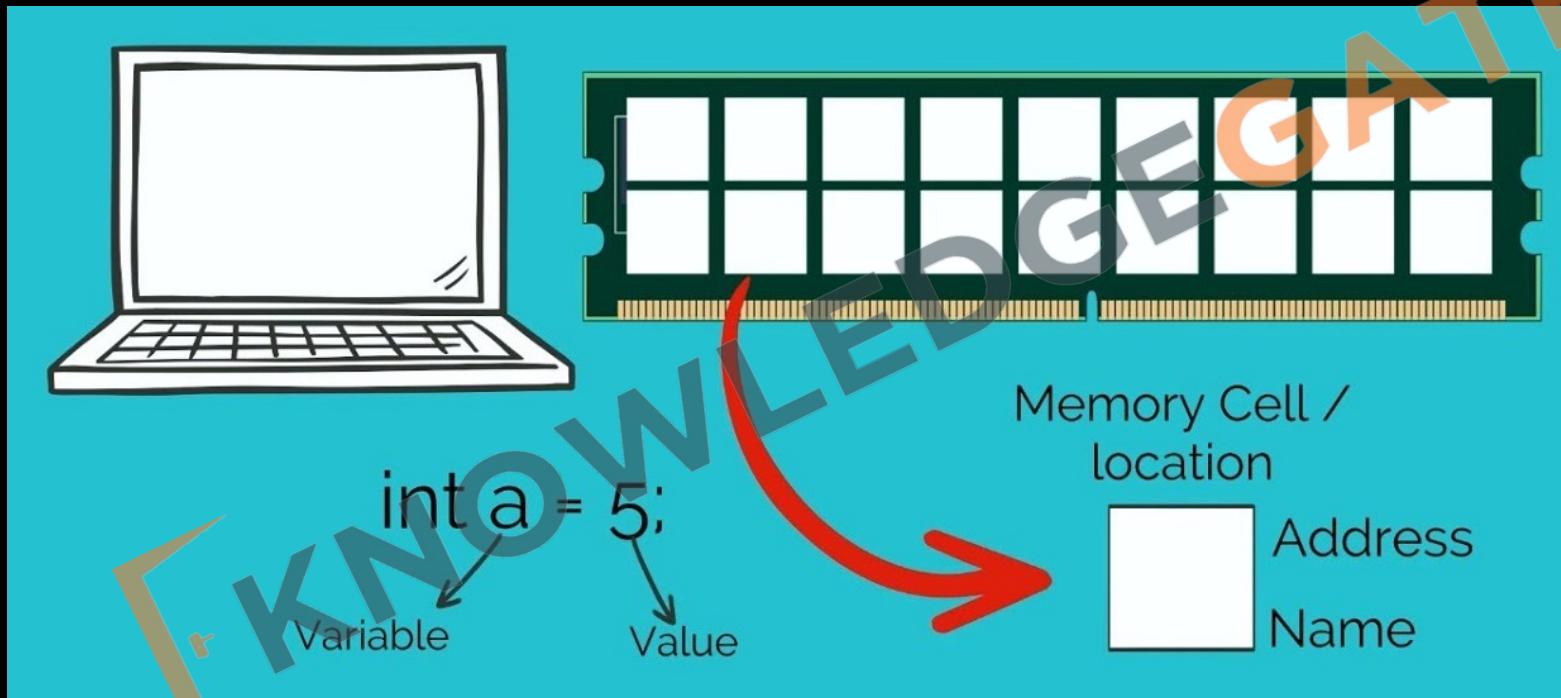
Variable is used to Store Data



Variables are like containers used for storing data values.



3.1. Memory Allocation

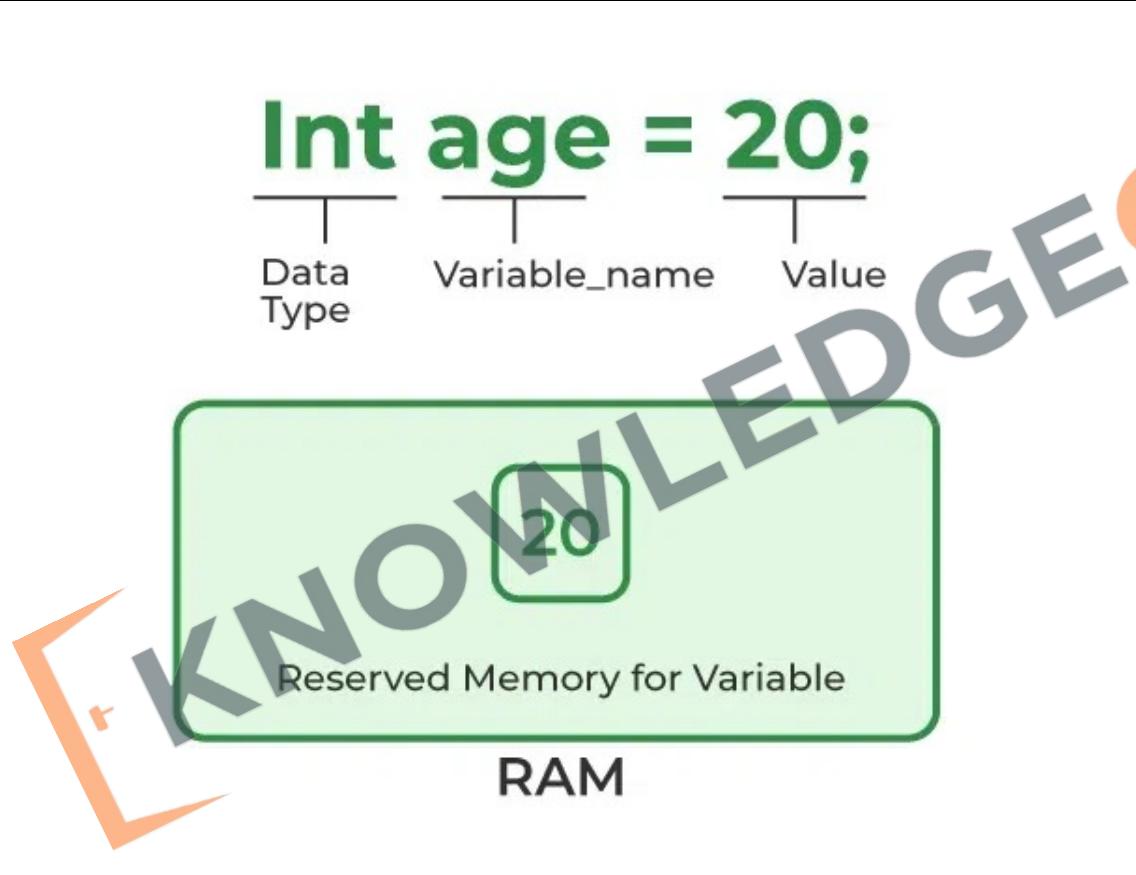
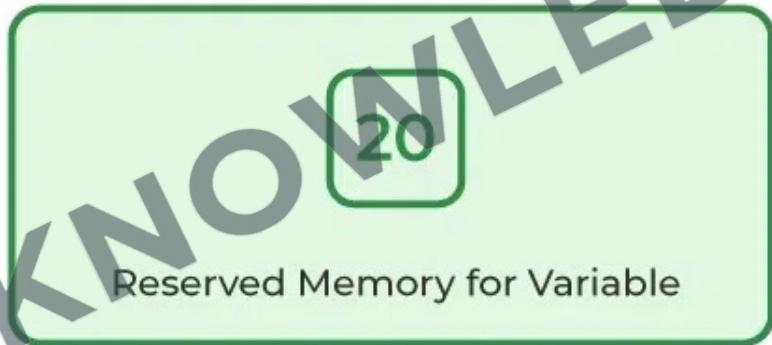




3.1. Variable Declaration

Int age = 20;

— — —
Data Type Variable_name Value



KNOWLEDGE GATE'



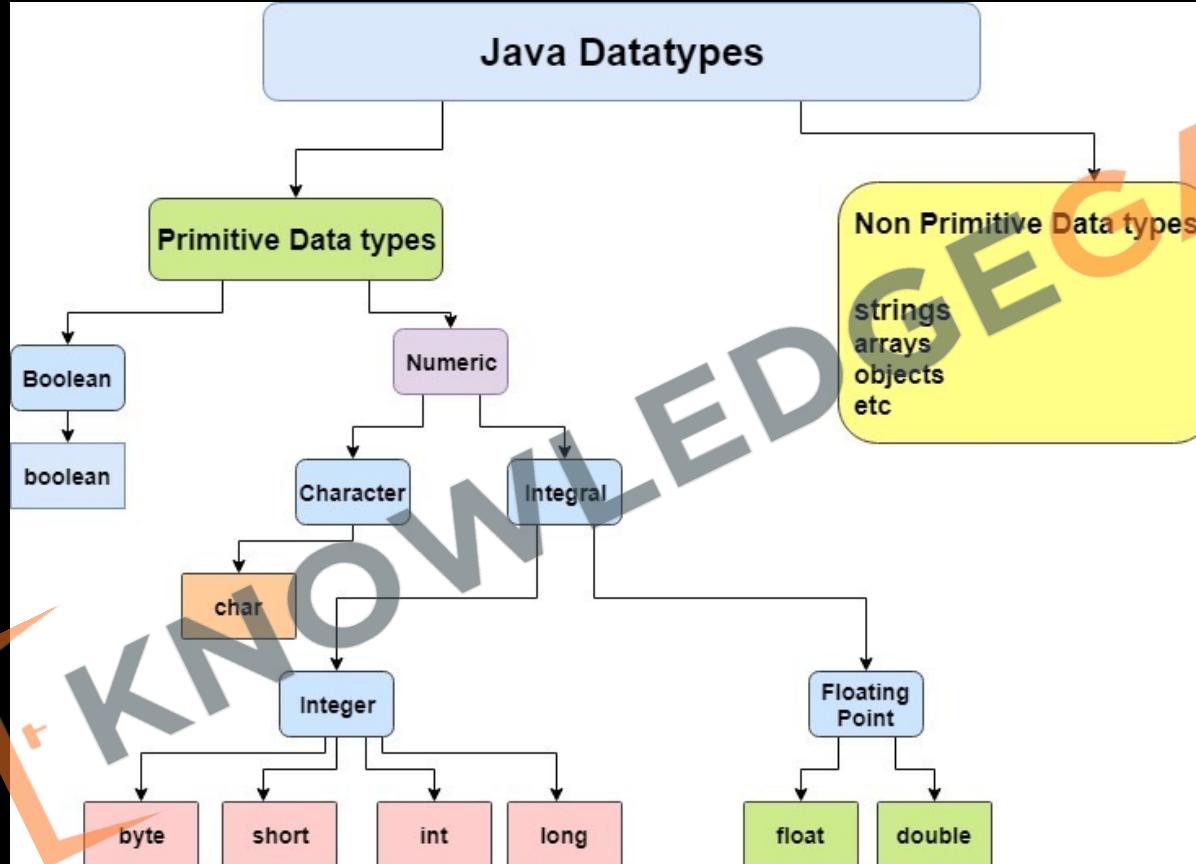
Java

3.2 Data Types

	Size	Default Value	Type of Value Stored
byte	1 byte	0	Integral
short	2 byte	0	Integral
int	4 byte	0	Integral
long	8 byte	0L	Integral
char	2 byte	'\u0000'	Character
float	4 byte	0.0f	Decimal
double	8 byte	0.0d	Decimal
boolean	1 bit (till JDK 1.3 it uses 1 byte)	false	True or False



3.2 Data Types





3.3. Naming Conventions

camelCase

- Start with a lowercase letter. Capitalize the first letter of each subsequent word.
- Example: `myVariableName`

snake_case

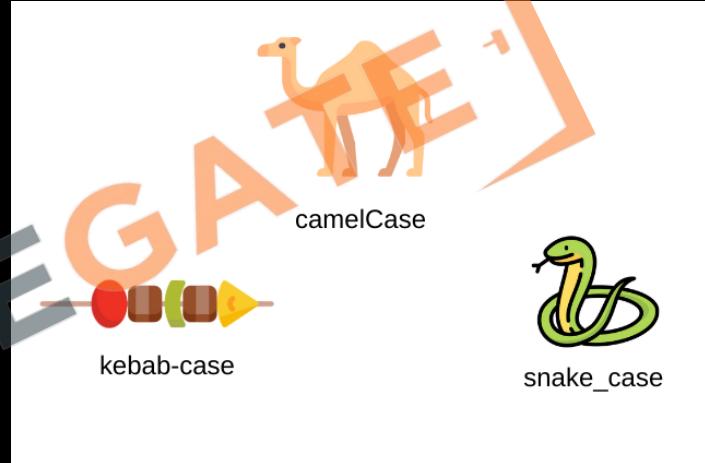
- Start with a lowercase letter. Separate words with `underscores`.
- Example: `my_variable_name`

Kebab-case

- All lowercase letters. Separate words with `hyphens`. Example:
`my-variable-name`

Keep a Good and Short Name

- Choose names that are descriptive but not too long. It should make it easy to understand the variable's purpose.
- Example: `age`, `firstName`, `isMarried`





3.3. Java Identifier Rules

1. The only allowed characters for identifiers are all alphanumeric characters([A-Z],[a-z],[0-9]), '\$' (dollar sign) and '_' (underscore).
2. Can't use keywords or reserved words
3. Identifiers should not start with digits([0-9]).
4. Java identifiers are case-sensitive.
5. There is no limit on the length of the identifier but it is advisable to use an optimum length of 4 – 15 letters only.

- | Identifier | Status |
|---------------|-----------|
| 1) \$ _ | (valid) |
| 2) Ca\$ h | (valid) |
| 3) Java2share | (valid) |
| 4) all@hands | (invalid) |
| 5) 123abc | (invalid) |
| 6) Total# | (invalid) |
| 7) Int | (valid) |
| 8) Integer | (valid) |
| 9) int | (invalid) |
| 10) tot123 | |



Java

3.4 Literals

Integer literals -> 10, 5, -8 etc...

Floating-point literals -> 1.2, 0.25, -1.999, etc...

Boolean literals -> true, false

Character literals -> 'a', 'A', 'N', 'q', etc...

String literals -> "hi", "hello", "What's up?"



Java

3.5 Keywords

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

SYNTHETIC
KEYWORDS
'AGGREGATE'



3.6 Escape Sequences

Escape Sequence	Description
\t	Insert a tab in the text at this point.
\b	Insert a backspace in the text at this point.
\n	Insert a newline in the text at this point.
\'	Insert a single quote character in the text at this point.
\"	Insert a double quote character in the text at this point.
\\\	Insert a backslash character in the text at this point.



CHALLENGE

Task:

4. Show the following patterns using single print statement:





3.7 User Input

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter your name:");
        String name = scanner.nextLine();
        System.out.println("Welcome " + name);
    }
}
```

NAME
nextInt();
nextDouble();
nextFloat();
nextLong();
nextShort();
next();
nextLine();



Java

CHALLENGE

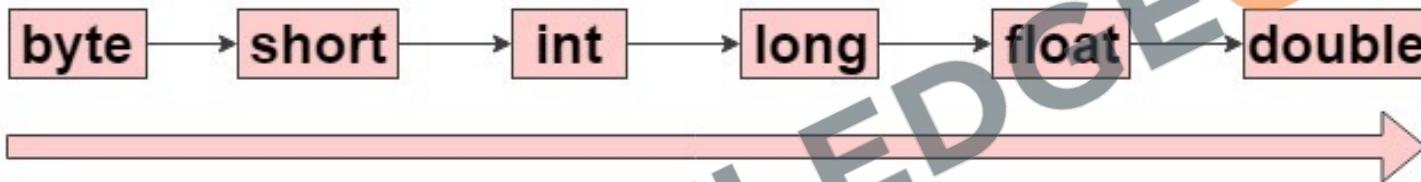
5. Create a program to input name of the person and respond with "Welcome NAME to KG Coding"
6. Create a program to add two numbers.



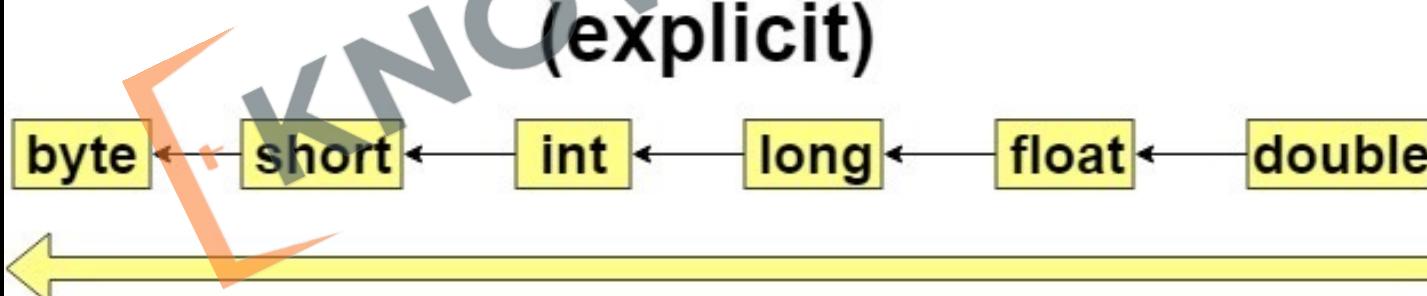


3.8 Type Conversion and Casting

Automatic Type Conversion (Widening - implicit)



Narrowing
(explicit)

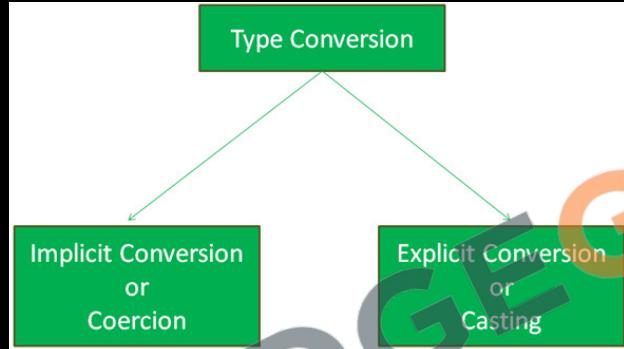




3.8 Type Conversion and Casting

```
// implicit  
long big = 45;  
float dec = 3;  
double d = 3.4f;
```

```
// explicit  
float eDec = (float) 3.4;  
long eBig = (long) 3.4;  
int eInt = (int) 3.4;
```





Revision

1. Variables
2. Data Types
3. Naming Conventions
4. Literals
5. Keywords
6. Escape Sequences
7. User Input
8. Type Conversion and Casting





Practice Exercise

Java Basics

Answer in True/False:

1. In Java, a variable's name can start with a number.
2. `char` in Java can store a single character.
3. Class names in Java typically start with a lowercase letter.
4. `100L` is a valid long literal in Java.
5. `\d` is an escape sequence in Java for a digit character.
6. `Scanner` class is used for reading console input.
7. In Java, an `int` can be automatically converted to a `byte`.
8. Java variable names are case-sensitive.
9. `Scanner` class can be used to read both primitive data types and strings.
10. Explicit casting is required to convert a `double` to an `int`.





Practice Exercise

Java Basics

Answer in True/False:

- | | |
|--|-------|
| 1. In Java, a variable's name can start with a number. | False |
| 2. <code>char</code> in Java can store a single character. | True |
| 3. Class names in Java typically start with a lowercase letter. | False |
| 4. <code>100L</code> is a valid long literal in Java. | True |
| 5. <code>\d</code> is an escape sequence in Java for a digit character. | False |
| 6. <code>Scanner</code> class is used for reading console input. | True |
| 7. In Java, an <code>int</code> can be automatically converted to a <code>byte</code> . | False |
| 8. Java variable names are case-sensitive. | True |
| 9. <code>Scanner</code> class can be used to read both primitive data types and strings. | True |
| 10. Explicit casting is required to convert a <code>double</code> to an <code>int</code> . | True |



KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)



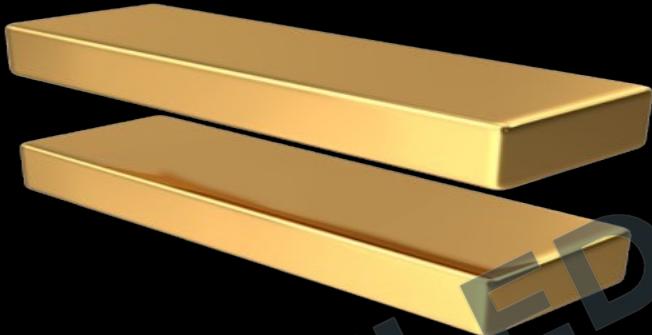
4. Operators, If-else & Number System

1. Assignment Operator
2. Arithmetic Operators
3. Order of Operation
4. Shorthand Operators
5. Unary Operators
6. If-else
7. Relational Operators
8. Logical Operators
9. Operator Precedence
10. Intro to Number System
11. Intro to Bitwise Operators





4.1 Assignment Operator



Assigns the **right-hand** operand's value to the **left-hand** operand.

Example: int a = 5;



CHALLENGE

7. Create a program to swap two numbers.





4.2 Arithmetic Operators

Operators	Meaning	Example	Result
+	Addition	4+2	6
-	Subtraction	4-2	2
*	Multiplication	4*2	8
/	Division	4/2	2
%	Modulus operator to get remainder in integer division	5%2	1



4.3 Order of Operation

B	O	D	M	A	S
Bracket	Order	Divide	Multiply	Add	Subtract
()	\sqrt{x} x^2	\div or \times		$+$ or $-$	
Parentheses	Exponents	Multiply	Divide	Add	Subtract
P	E	M	D	A	S

$$9 \div 3 \times 2 \div 6$$

$$8 - 5 + 7 - 1$$



KNOWLEDGE GATE'



4.4 Shorthand Operators

Operator symbol	Name of the operator	Example	Equivalent construct
<code>+=</code>	Addition assignment	<code>x += 4;</code>	<code>x = x + 4;</code>
<code>-=</code>	Subtraction assignment	<code>x -= 4;</code>	<code>x = x - 4;</code>
<code>*=</code>	Multiplication assignment	<code>x *= 4;</code>	<code>x = x * 4;</code>
<code>/=</code>	Division assignment	<code>x /= 4;</code>	<code>x = x / 4;</code>
<code>%=</code>	Remainder assignment	<code>x %= 4;</code>	<code>x = x % 4;</code>

KNOWLEDGE GATE



4.5 Unary Operators

Operator	Description	Example
-	Converts a positive value to a negative	<code>x = -y</code>
Pre Increment	Increment the value by 1 and then use it in our statement	<code>x = ++y</code>
Pre Decrement	Decrement the value by 1 and then use it in our statement	<code>x = --y</code>
Post Increment	Use current value in the statement and then increment it by 1	<code>x = y++</code>
Post Decrement	Use current value in the statement and then decrement it by 1	<code>x = y--</code>



CHALLENGE

8. Create a program that takes two numbers and shows result of all arithmetic operators (+,-,*,/,%).

9. Create a program to calculate product of two floating points numbers.

10. Create a program to calculate Perimeter of a rectangle.

$$\text{Perimeter of rectangle ABCD} = A+B+C+D$$

11. Create a program to calculate the Area of a Triangle.

$$\text{Area of triangle} = \frac{1}{2} * B * H$$

12. Create a program to calculate simple interest.

$$\text{Simple Interest} = (P \times T \times R)/100$$

13. Create a program to calculate Compound interest.

$$\text{Compound Interest} = P(1 + R/100)t$$

14. Create a program to convert Fahrenheit to Celsius

$$^{\circ}\text{C} = (^{\circ}\text{F} - 32) \times 5/9$$





Java

4.6 if-else

1. Syntax: Uses `if () {}` to check a condition.
2. What is `if`: Executes block if condition is `true`, skips if `false`.
3. What is `else`: Executes a block when the if condition is `false`.
4. Curly Braces can be omitted for single statements, but not recommended.
5. If-else Ladder: Multiple if and else if blocks; `only one` executes.
6. Use Variables: Can store `conditions` in variables for use in if statements.

```
if thirsty {  
      
} else {  
      
}
```



4.7 Relational Operators

< , > , <= , >= , == , !=

Equality

- `==` Checks value equality.

Inequality

- `!=` Checks value inequality.

Relational

- `>` Greater than.
- `<` Less than.
- `>=` Greater than or equal to.
- `<=` Less than or equal to.

Order of Relational operators is less than arithmetic operators



4.8 Logical Operators



AND

Or

not

1. Types: **&&** (AND), **||** (OR), **!** (NOT)
2. AND (**&&**): All conditions **must be true** for the result to be true.
3. OR (**||**): Only **one condition** must be true for the result to be true.
4. NOT (**!**): **Inverts** the Boolean value of a condition.
5. Lower Priority than **Math** and **Comparison** operators



CHALLENGE

15. Create a program that determines if a number is positive, negative, or zero.

16. Create a program that determines if a number is odd or even.

17. Create a program that determines the greatest of the three numbers.

18. Create a program that determines if a given year is a leap year (considering conditions like divisible by 4 but not 100, unless also divisible by 400).

19. Create a program that calculates grades based on marks

A -> above 90%

B -> above 75%

C -> above 60%

D -> above 30%

F -> below 30%

20. Create a program that categorize a person into different age groups

Child -> below 13

Teen -> below 20

Adult -> below 60

Senior-> above 60





4.9 Operator Precedence

Operator Type	Category	Precedence	Associativity
Unary	postfix	a++, a--	Right to left
	prefix	++a, --a, +a, -a, ~, !	Right to left
Arithmetic	Multiplication	*, /, %	Left to Right
	Addition	+, -	Left to Right
Shift	Shift	<<, >>, >>>	Left to Right
Relational	Comparison	<, >, <=, >=, instanceOf	Left to Right
	equality	==, !=	Left to Right
Bitwise	Bitwise AND	&	Left to Right
	Bitwise exclusive OR	^	Left to Right
	Bitwise inclusive OR		Left to Right
Logical	Logical AND	&&	Left to Right
	Logical OR		Left to Right
Ternary	Ternary	? :	Right to Left
Assignment	assignment	=, +=, -=, *=, /=, %-=, &=, ^=, =, <<=, >>=, >>>=	Right to Left

Operator Precedence: Determines the **evaluation order of operators** in an expression based on their priority levels.

Associativity: Defines the **order of operation for operators with the same precedence**, usually left-to-right or right-to-left.



4.10 Intro to Number System

Number System	Base or Radix	Digits or symbols
Binary	2	0,1
Octal	8	0,1,2,3,4,5,6,7
Hexadecimal	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Number System	Base or Radix	Digits or symbols
Unary	1	0/1
Decimal	10	0,1,2,3,4,5,6,7,8,9



4.10 Intro to Number System

Decimal Number System

- Decimal Number System is a base-10 system that has ten digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.
- The decimal number system is said to be of base, or radix, 10 because it uses 10 digits and the coefficients are multiplied by powers of 10.
- This is the base that we often use in our day to day life.
- Example: $(7,392)_{10}$, where 7,392 is a shorthand notation for what should be written as

$$7 * 10^3 + 3 * 10^2 + 9 * 10^1 + 2 * 10^0$$



4.10 Intro to Number System

Binary Number System

- The coefficients of the binary number system have only two possible values: 0 and 1.
- Each coefficient a_j is multiplied by a power of the radix, e.g., 2^j , and the results are added to obtain the decimal equivalent of the number.

Example: $(11010.11)_2$ value in Decimal?

$$1 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0 + 1 * 2^{-1} + 1 * 2^{-2} = 26.75$$



4.10 Intro to Number System

Decimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

KNOWLEDGE GATE



Java

4.11 Intro to Bitwise Operators

1. AND Operator (&): Performs on two integers. Each bit of the output is 1 if the corresponding bits of both operands are 1, otherwise 0.

2. OR Operator (|): Performs on two integers. Each bit of the output is 0 if the corresponding bits of both operands are 0, otherwise 1.

3. XOR Operator (^): Performs on two integers. Each bit of the output is 1 if the corresponding bits of the operands are different.

4. NOT Operator (~): Performs a bitwise complement. It inverts the bits of its operand (0 becomes 1, and 1 becomes 0).

5. Left Shift Operator (<<): Shifts the left operand's bits to the left by the number of positions specified by the right operand, filling the new rightmost bits with zeros.

6. Right Shift Operator (>>): Shifts the left operand's bits to the right. If the left operand is positive, zeros are filled into the new leftmost bits; if negative, ones are filled in.



CHALLENGE

21. Create a program that shows **bitwise AND** of two numbers.
22. Create a program that shows **bitwise OR** of two numbers.
23. Create a program that shows **bitwise XOR** of two numbers.
24. Create a program that shows **bitwise compliment** of a number.
25. Create a program that shows use of **left shift** operator.
26. Create a program that shows use of **right shift** operator.
27. Write a program to check if a given number is even or odd using **bitwise operators**.





Revision

1. Assignment Operator
2. Arithmetic Operators
3. Order of Operation
4. Shorthand Operators
5. Unary Operators
6. If-else
7. Relational Operators
8. Logical Operators
9. Operator Precedence
10. Intro to Number System
11. Intro to Bitwise Operators



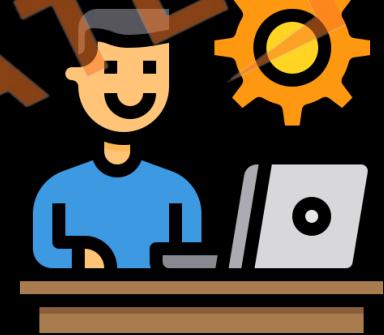


Practice Exercise

Java Basics

Answer in True/False:

1. In Java, `&&` and `||` operators perform short-circuit evaluation.
2. In an if-else statement, the else block executes only when the if condition is false.
3. Java allows an if statement without the else part.
4. The `^` operator in Java is used for exponentiation.
5. Unary minus operator can be used to negate the value of a variable in Java.
6. `a += b` is equivalent to `a = a + b` in Java.
7. In Java, the binary number system uses base 10.
8. The number `1010` in binary is equivalent to 10 in decimal.
9. `&` and `|` are logical operators in Java.
10. In Java, `a >> 2` shifts the binary bits of a to the left by 2 positions.





Practice Exercise

Java Basics

Answer in True/False:

1. In Java, `&&` and `||` operators perform short-circuit evaluation. True
2. In an if-else statement, the else block executes only when the if condition is false. True
3. Java allows an if statement without the else part. True
4. The `^` operator in Java is used for exponentiation. False
5. Unary minus operator can be used to negate the value of a variable in Java. True
6. `a += b` is equivalent to `a = a + b` in Java. True
7. In Java, the binary number system uses base 10. False
8. The number `1010` in binary is equivalent to `10` in decimal. True
9. `&` and `|` are logical operators in Java. False
10. In Java, `a >> 2` shifts the binary bits of `a` to the left by 2 positions. False



KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)

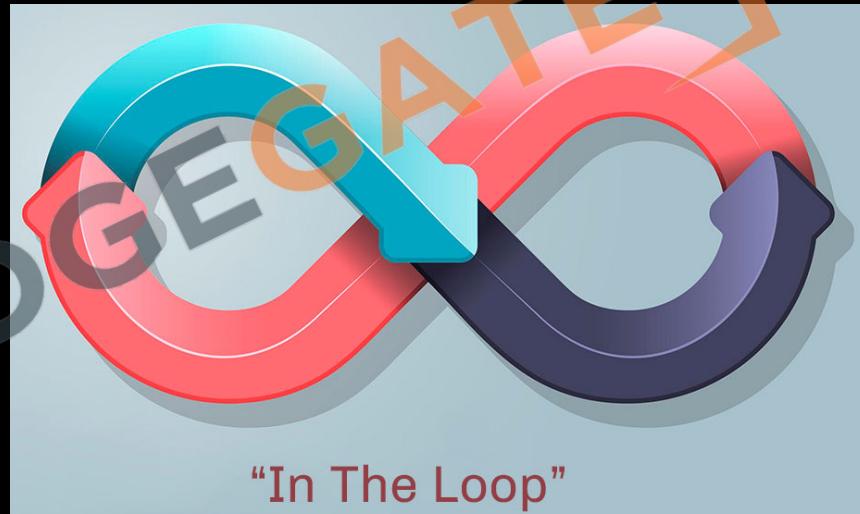


[Sanchit Socket](#)



5 While Loop, Methods & Arrays

1. Comments
2. While Loop
3. Methods
4. Return statement
5. Arguments
6. Arrays
7. 2D Arrays



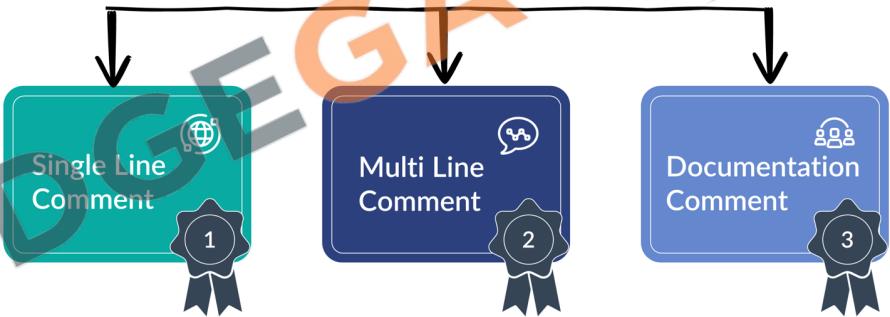
"In The Loop"



5.1 Java Comments

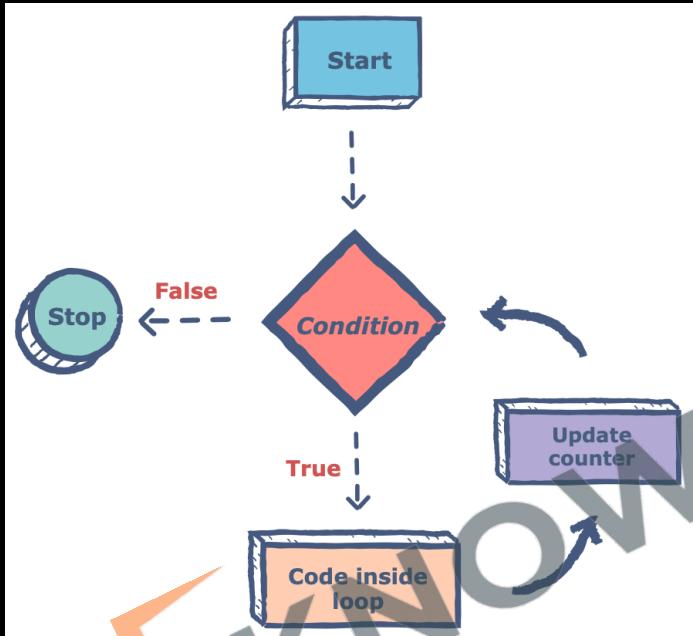
1. Used to add **notes** in **Java** code
2. **Not displayed** on the web page
3. Helpful for **code organization**
4. **Syntax:**
 1. Single Line: `//`
 2. Multi Line: `/* */`
 3. Java Docs: `/***/`

Type Of Comments in Java





5.2 What is a Loop?



1. Code that runs **multiple times** based on a condition.
2. Repeated execution of code.
3. Loops automate **repetitive tasks**.
4. Types of Loops: **while, for, while, do-while**.
5. Iterations: Number of times **the loop runs**.



5.2 While Loop

Execution

Initialization

condition

false

inside loop

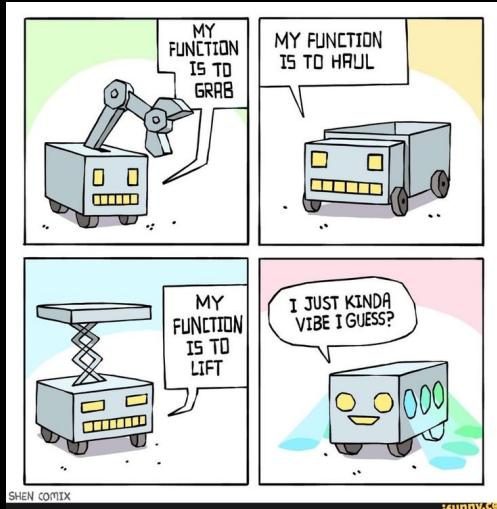
true

```
while (condition) {  
    // Body of the loop  
}
```

1. Iterations: Number of times the loop runs.
2. Used for non-standard conditions.
3. Repeating a block of code while a condition is true.
4. Remember: Always include an update to avoid infinite loops.



5.3 What are Functions / Methods?



1. Definition: Blocks of **reusable code**.
2. DRY Principle: "Don't Repeat Yourself" it Encourages code **reusability**.
3. Usage: Organizes **code** and performs specific tasks.
4. Naming Rules: Same as **variable** names: **camelCase**
5. Example: "Beta Gas band kar de"



5.3 Method Syntax

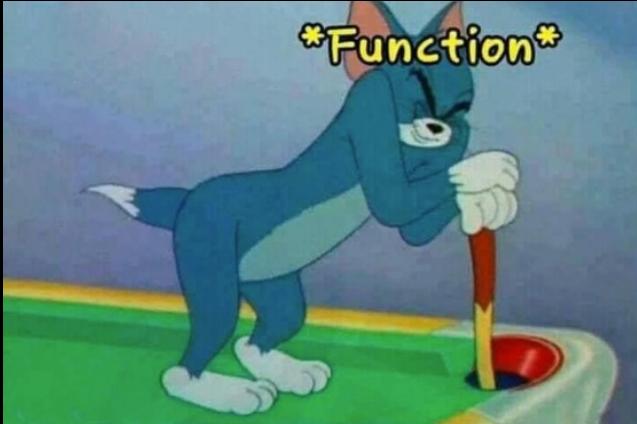
```
return-type <--> method name <--> parameter-list  
modifier  
public int addition(int operand1, int operand2)  
{  
    int sum;  
    sum = operand1 + operand2;  
    return sum;  
}
```

body
→ of the
method

1. Follows same rules as variable names.
2. Use () to contain parameters.
3. Invoke by using the function name followed by () .
4. Fundamental for code organization and reusability.



5.4 Return statement



1. Sends a value back from a function.
2. Example: "Ek glass paani laao"
3. What Can Be Returned: Value, variable, calculation, etc.
4. Return ends the function immediately.
5. Function calls make code jump around.
6. Prefer returning values over using global variables.



5.5 Arguments vs Parameters



Parameter



Function



Return

1. Input values that a **function** takes.
2. Parameters put value into function, while return gets value out.
3. Example: "Ek packet dahi laao"
4. Naming Convention: Same as **variable** names.
5. **Parameter** vs **Argument**
6. Examples: `System.out.print`, `Scanner.nextInt()`, are functions we have already used
7. **Multiple Parameters**: Functions can take **more than one**.
8. **Default Value**: Can set a **default** value for a parameter.



CHALLENGE

28. Develop a program that prints the **multiplication table** for a given number.
29. Create a program to **sum all odd numbers** from 1 to a specified number N.
30. Write a function that **calculates the factorial** of a given number.
31. Create a program that computes the **sum of the digits** of an integer.
32. Create a program to find the **Least Common Multiple (LCM)** of two numbers.
33. Create a program to find the **Greatest Common Divisor (GCD)** of two integers.
34. Create a program to check whether a given **number is prime**.
35. Create a program to **reverse the digits** of a number.
36. Create a program to print the **Fibonacci series** up to a certain number.
37. Create a program to check if a number is an **Armstrong number**.
38. Create a program to verify if a **number is a palindrome**.
39. Create a program that print patterns:

* * * * *
* * * * *
* * * * *
* * * * *
* * * * *

Right Half Pyramid

* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
*

Reverse Right Half Pyramid

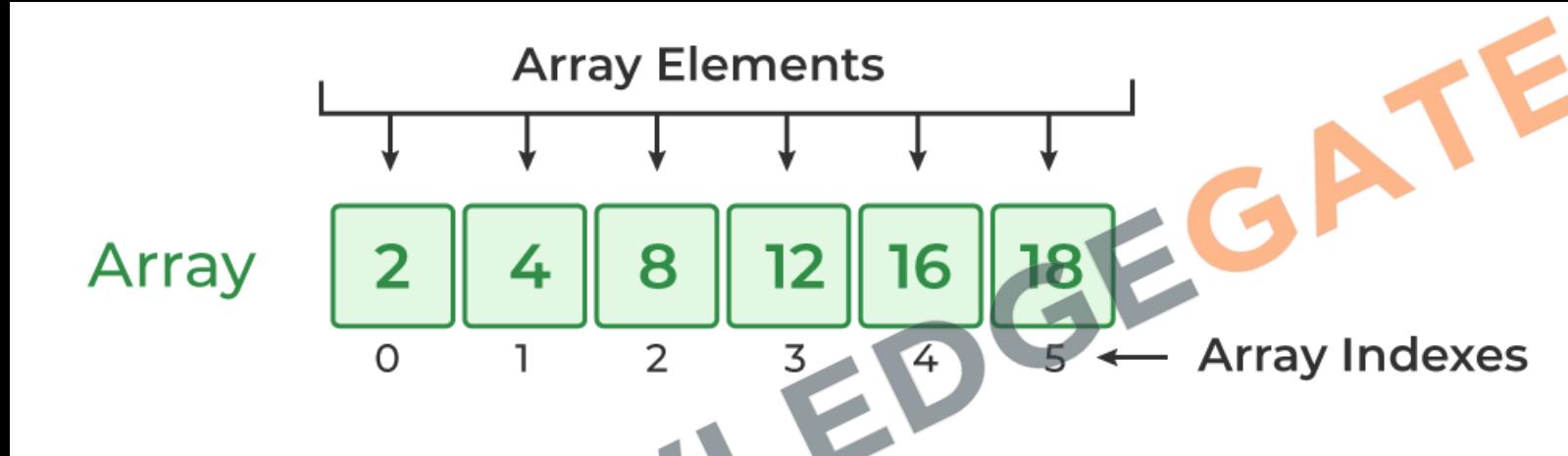
*
* *
* * *
* * * *
* * * * *
* * * * *

Left Half Pyramid





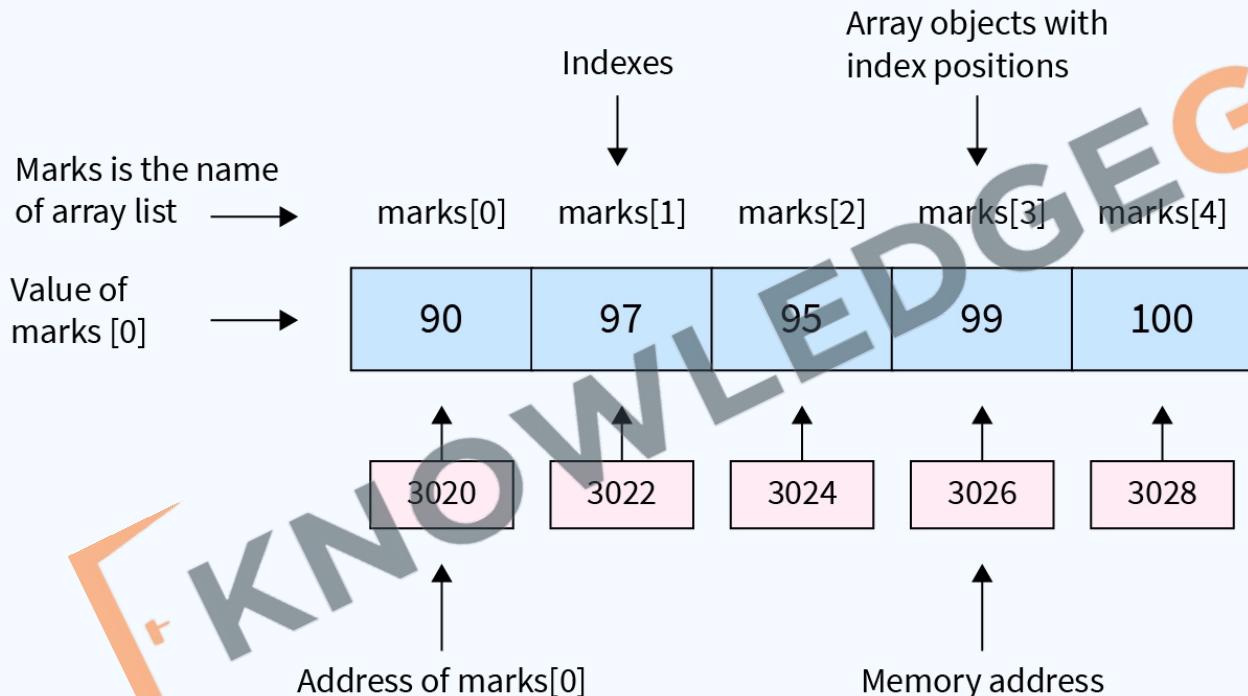
5.6 What is an Array?



1. An Array is just a list of values.
2. Index: Starts with 0.
3. Arrays are used for storing multiple values in a single variable.



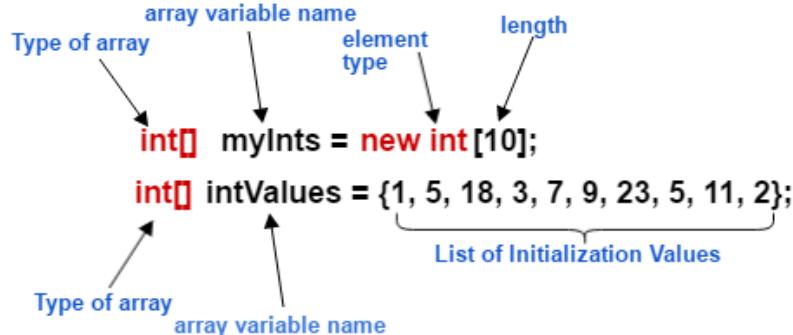
5.6 Array Memory





Java

5.6 Array Syntax



1. Use {} to create a new array, {} brackets enclose list of values
2. Arrays can be saved to a variable.
3. Accessing Values: Use [] with index.
4. Syntax Rules:
 - Brackets start and end the array.
 - Values separated by commas.
 - Can span multiple lines.



Java

5.6 Array Length

Array length = Array's Last Index + 1

1	2	3	4	5	6	7	8
0	1	2	3	4	5	6	7

Array's last index = 7

$$\text{Arraylength} = 7 + 1 = 8$$



KNOWLEDGE GATE



5.7 2D Arrays

Types of Multidimensional Array in Java

	Column 0	Column 1	Column 2
Row 0	X[0][0]	X[0][1]	X[0][2]
Row 1	X[1][0]	X[1][1]	X[1][2]
Row 2	X[2][0]	X[2][1]	X[2][2]

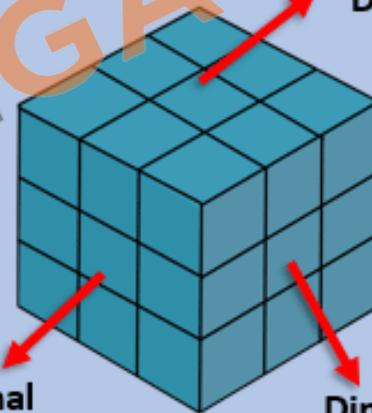
2D-Array

Second
Dimensional

First
Dimensional

Third
Dimensional

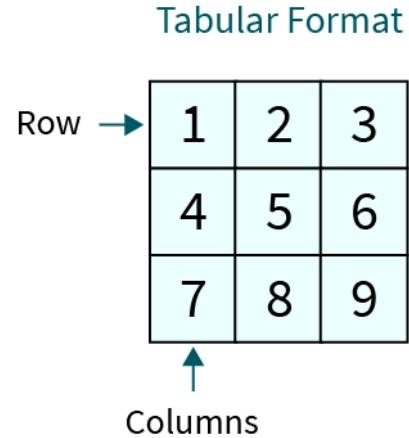
3D-Array





Java

5.7 2D Arrays



Java
Object Reference

Array of Pointers

1	2	3
4	5	6
7	8	9

Actual Data
In Different
1D Arrays



Java

5.7 2D Arrays

```
int[][] numArr = new int[2][3];
int[][] inArray = {{1, 2, 5}, {8, 9, 4}};

numArr[0][0] = 5;
```



CHALLENGE

40. Create a program to find the **sum and average** of all elements in an array.
41. Create a program to find **number of occurrences** of an element in an array.
42. Create a program to find the **maximum and minimum element** in an array.
43. Create a program to **check** if the given array is **sorted**.
44. Create a program to return a new array **deleting** a specific element.
45. Create a program to **reverse** an array.
46. Create a program to check if the array is **palindrome** or not.
47. Create a program to **merge two sorted arrays**.
48. Create a program to **search** an element in a **2-D array**.
49. Create a program to do **sum and average** of all elements in a **2-D array**.
50. Create a program to find the sum of two **diagonal elements**.





Java

Revision

1. Comments
2. While Loop
3. Methods
4. Return statement
5. Arguments
6. Arrays
7. 2D Arrays





Practice Exercise

Java Basics

Answer in True/False:

1. A 'while' loop in Java continues to execute as long as its condition is true.
2. The body of a 'while' loop will execute at least once, regardless of the condition.
3. A 'while' loop cannot be used for iterating over an array.
4. Infinite loops are not possible with 'while' loops in Java.
5. A method in Java can return more than one value at a time.
6. It's mandatory for every Java method to have a return type.
7. The size of an array in Java can be modified after it is created.
8. Arrays in Java can contain elements of different data types.
9. An array is a reference type in Java.
10. Java arrays are zero-indexed, meaning the first element has an index of 0.



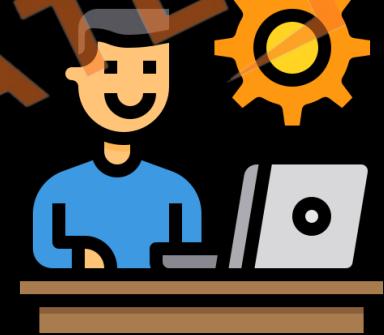


Practice Exercise

Java Basics

Answer in True/False:

1. A 'while' loop in Java continues to execute as long as its condition is true. True
2. The body of a 'while' loop will execute at least once, regardless of the condition. False
3. A 'while' loop cannot be used for iterating over an array. False
4. Infinite loops are not possible with 'while' loops in Java. False
5. A method in Java can return more than one value at a time. False
6. It's mandatory for every Java method to have a return type. True
7. The size of an array in Java can be modified after it is created. False
8. Arrays in Java can contain elements of different data types. False
9. An array is a reference type in Java. True
10. Java arrays are zero-indexed, meaning the first element has an index of 0. True



KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)



<http://www.kgcoding.in/>

Our YouTube Channels

KG Coding Android App



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)