**Link github:** [**tinisang (Phạm Tiến Sang)**](https://github.com/tinisang)

**Ref:**

1. [greekforgreek java](https://www.google.com/search?q=greeksforgreek+java%C2%A0&client=safari&sca_esv=a29bcbd63c283a80&hl=vi-jp&sxsrf=ADLYWILF63iW0mDYaeHlGhMk0KCstM40DA%3A1735546074713&ei=2lRyZ9-aK5na2roP2JTq2Q8&ved=0ahUKEwjf75j6hM-KAxUZrVYBHViKOvsQ4dUDCBA&uact=5&oq=greeksforgreek+java%C2%A0&gs_lp=Egxnd3Mtd2l6LXNlcnAiFWdyZWVrc2ZvcmdyZWVrIGphdmHCoDIIEAAYgAQYogQyBRAAGO8FMggQABiABBiiBDIIEAAYgAQYogRI-AhQgwNY4AdwAngBkAEBmAGBAqABqAeqAQUwLjQuMbgBA8gBAPgBAZgCBKAChAPCAgoQABiwAxjWBBhHwgIIEAAYFhgKGB6YAwCIBgGQBgiSBwMyLjKgB50N&sclient=gws-wiz-serp) // bao gồm cả cơ bản và nâng cao
2. <https://vietjack.com/java/> // bản tiếng việt
3. <https://docs.oracle.com/javase/specs/jls/se23/html/index.html> / tài liệu chính thống nhất
4. Và video anh Kính gửi trong discord

<https://drive.google.com/drive/folders/11EiLGu96yL2SYeDEKeJKfRR_llFt-EQs>

**Content:**

1. History of Java (30 minutes)
   1. Java History:

* In 1991, James Gosling, Mike Sheridan and Patrick Naughton, a team of engineers from Sun MicroSystem, set out to create a revolutionary programming language which could be embedded into electronics.
* Originally introduced as 'Oak,' it was later renamed to 'Java,' inspired by the name of the Indonesian island where the first coffee was produced.

1. What is Java? (1 hours)
   1. Introduction to Java:

* Applications:
* Enterprise Applications
* Mobile apps
* Webs-based application
* Server-side Programming
* Big Data Processing
* Cloud-based Application
* Features:
* *A class-based,* Object Oriented Programming Language: this approach to programming makes the code more modular, reusable and makes it easier to handle complex applications.
* *Platform Independent Language*, WORA (Write Once and Run All): Does Not rely on any specific hardware dependency. => so its portable across various platforms supporting JAVA Runtime Environment:

-> In essence, your program runs on JVM (Java Virtual Machine), not directly on the hardware. Each operating system has a different JVM, but after execution of the bytecode, the output produced is the same.

-> How WORA works:

**JAVA Code(.java) -> JAVAC Compiler -> Bytecode (.class) -> JVM -> Machine code (run on the host system)**

* *High performance*: Highly optimized
* *Security*
  1. Comparing Java with Other Languages:
* JAVA is a compiler/interpreted language
* Compiled Language: Code can be executed directly by your computer’s CPU:

**Code -> Compiler -> Machine language**

* Interpreted Language: Another program executes the code

Code -> **Interpreter (read each statement of codes -> convert and execute them directly)**

* 1. Java Versions:
* Enterprises normally use **older but still maintained version** since a huge amount of code could be altered when upgrading the newer version

1. Languages processor(48 hours): (<https://www.geeksforgeeks.org/language-processors-assembler-compiler-and-interpreter/>)

* Language processor / language translator is a computer software which converts the source code from a language to another language or to a machine code that a computer can understand
* Most computer programs are written in high-level languages such as Java, C++, Python,... They are called the **source code.** Source codes can not directly be executed by the computer unless **being converted into Machine codes** which a computer can understand
* 3 types:
* Compilers
* Interpreter
* Assembler
* Compilers, Interpreter translate programs which use **high level** programming languages
* Assemblers translate programs which use **low level** programming language **(assembly language)**
  1. Assembler
* Assembly Language: low level, hardware specific
* Assembler is basically the very first interface which is able to communicate human with machine
* Convert mnemonics *(dependent on the architecture of the machine)* into binary code

**Source Code (Assembly Language) -> Assembler -> Object Code (Machine Language)**

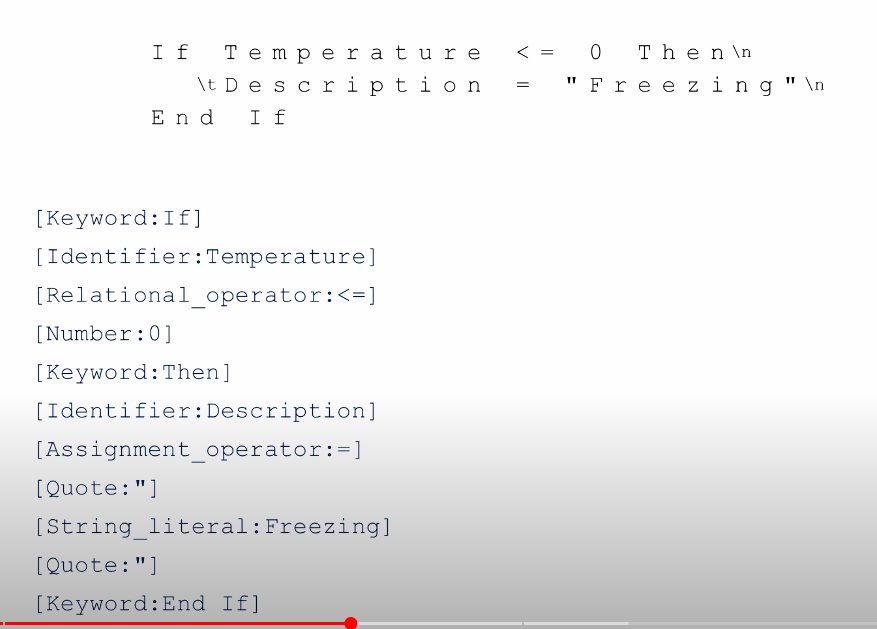
* 1. Compiler
* Translate the program as a whole
* The source code must be error free to be translated successfully by a compiler
* If there is any error in the source code, a compiler will specify that error at the end of compilation with the line number where the error was detected. You have to remove or correct your code before compiling your source code

**Source Code (High Level Language) -> Compiler -> Object Code(Machine Language)**

* 1. Interpreter
* An interpreter translates one line at a time and then executes it after moving to the next line.
* Does not convert sourcode into Object Code
* It will terminate its translating process if there is any error detected in the statement and only continue this process after the removal of that error.

STEPS IN **COMPILATION PROCESS:**



* 1. Lexical analysis
* First phase of Natural Language Processing
* Purpose:
* Take the input as the source codes and break them into smaller and meaningful elements called tokens (tokenization)
* A token can be an individual words, symbols in a sentence such as keywords, variable names, numbers, punctuation
* [link tham khảo](https://www.youtube.com/watch?v=sJKFLcsysVs)
* Lexeme: The sequence of characters matched by a pattern to form a corresponding type of token
* How Lexical Analyzer works:
* Input preprocessing: take the input text -> strip out the whitespace, comments, non essential characters = eliminating all redundant information like white space and comments
* Tokenization: break the preprocessed input text into smaller, meaningful parts by matching sequences of characters against a set of patterns which define different tokens type
* Token classification: determine type of each token. Normally in most programming languages, lexer may classify identifiers, keywords, operators, punctuation symbols as different types of token.
* Token Validation: check if each token is valid according to rules of programming language?.
* Output generation: list of tokens, passed to the next stage of compilation

=> It essentially acts as a bridge between human-understandable code and machine-understandable structure.

* 1. Syntax Analysis
  2. Semantic Analysis
  3. Intermediate Code Generation
  4. Code Optimization
  5. Code Generation
  6. Error Handling
  7. What is the difference between compiler and CPU?

1. Development Tools (JDK, JRE, JVM)(phần này rất quan trọng nhé) (48 hours)
   1. Development Tools (JDK)
   2. Java Runtime Environment (JRE)
   3. What is JVM? Structure of the JVM
   4. Class Loader in JVM
   5. Memory Areas in JVM
   6. Execution Engine
   7. Garbage Collection in JVM
   8. Native Interface
2. Setting Up the Development Environment (set up không được thì báo mình) (6 hours)
   1. Introduction to IDEs (Integrated Development Environments)
   2. Java Build Tools: Maven and Gradle
   3. Tools Needed for Local Environment Setup
   4. Setting Up Your Local Environment and IDE for the Course
   5. Your First Java Program
   6. Run Java with IDE or Command line
3. Basic Java Syntax (8 hours)
   1. Terminologies of a Basic Program Structure
   2. Java Keywords
   3. Java Identifier
   4. Comments in Java

* “//” : inline comment
* “/\*”: multiline comment
  1. Package in Java
  2. Imports in Java
  3. Java docs

1. Data Types (8 hours)
   1. What Are Data Types?

* Data type limits the value which a available can hold, limit the operation supported on that variable,
* Consists of 2 main data types:
* Primitive Types:
* Numeric Types: integer, float, double, byte, short, long
* Boolean Type
* Reference Type:
* Class
* Array
* Type
* Interface
* Normal Interface: 1 or more than 1 method
* Functional Interface: 1 method
* Marker: no method
  1. Primitive Data Types
  2. Non-Primitive Data Types (Reference Types)
  3. Literals in Java
  4. Literals - Integer literals
  5. Literals - Floating-Point Literals
  6. Literals - Boolean and Characters
  7. Escape sequences, Null LiteralsTextBlocks

1. Variables ( 8 hours)
   1. What Is a Variable?
   2. Declaring Variables in Java (Rules)
   3. Initializing Variables in Java
   4. Types of Variables (Local, Instance, Static)
   5. Variable Scope
   6. Type Casting (Implicit and Explicit Casting)
2. Input, output (8 hours)
   1. Taking Input from Users
   2. Using the Scanner Class
   3. Using the BufferedReader Class
   4. Displaying Output to Users
   5. print() vs println() in Java
   6. Formatted Output in Java
   7. Logging Output
3. Operators (8 hours)
   1. What Are Operators?
   2. Arithmetic Operators
   3. Unary Operators
   4. Assignment Operator
   5. Relational Operators
   6. Logical Operators
   7. Ternary Operator
   8. Bitwise Operators
   9. Shift Operators
   10. instance of operator
4. Control Structures (8 hours)
   1. Decision making in Java
   2. If statement
   3. Else statement
   4. if else if statement
   5. Switch statement
   6. loops
   7. for loop
   8. while
   9. do while
   10. for each
   11. continue
   12. break
   13. return
5. Arrays (8 hours)
   1. What are arrays in java?
   2. Declaring and Initializing Arrays
   3. Methods in java.util.Arrays
   4. Multi-Dimensional Arrays
   5. java.lang.reflect.Array Class
6. Methods (8 hours)
   1. Method Syntax
   2. Methods with Parameters
   3. The main Method
   4. Method Recursion