# ASSIGNMENT NO.1

## 1. Reading Assignment: A Short History of Java

- **Task**: Read about the history and development of Java.
- **Link**: http://sunsite.uakom.sk/sunworldonline/swol-07-1995/swol-07-java.html

## 2. Reading Assignment: Java Language Features

- **Task**: Learn about the main features of Java.
- **Link**: https://javaalmanac.io/features/

## 3. Reading Assignment: Which Version of JDK Should I Use?

- **Task**: Find out which JDK version is right for you.
- **Link**: https://whichjdk.com/

## 4. Reading Assignment: JDK Installation Directory Structure

- **Task**: Understand the folder structure and files in the JDK installation.
- **Link**: https://docs.oracle.com/javase/8/docs/technotes/tools/windows/jdkfiles.html

## 5. Reading Assignment: About Java Technology

- **Task**: Read about the basics of Java technology and its components.
- **Link**: https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html

## 6. Coding Assignments

1. **Hello World Program**: Write a Java program that prints "Hello World!!" to the console.

   ```
   Sol:- class Ayush{

     public static void main[String[] args]{

        System.out.println("Hello,World!");

     }

   }
   ```
2. **Compile with Verbose Option**: Compile your Java file using the `-verbose` option with `javac`. Check the output.
   -verbose option helps us see some additional information about what compiler is doing while compiling our source file like it loads a bunch of modules before searching of jdk in program files folder like some  modules classes  related to sql , xml,security etc and then searches for path we set in environment path variable and then loads .class files like stream files string .class files etc and then checks the main class in src as we used print in the program it also loads class files of outputstream .class and some other .class files and then finally writes our src to a .class file

3. **Inspect Bytecode**: Use the `javap` tool to examine the bytecode of the compiled `.class` file. Observe the output.

   As we know our .class has bytcode of the src we wrote after compilation but we cannot read byte code as it is in form of 1 and 0's so inorder to read the .class we can use javap tool that shows the what is written in side .class file like

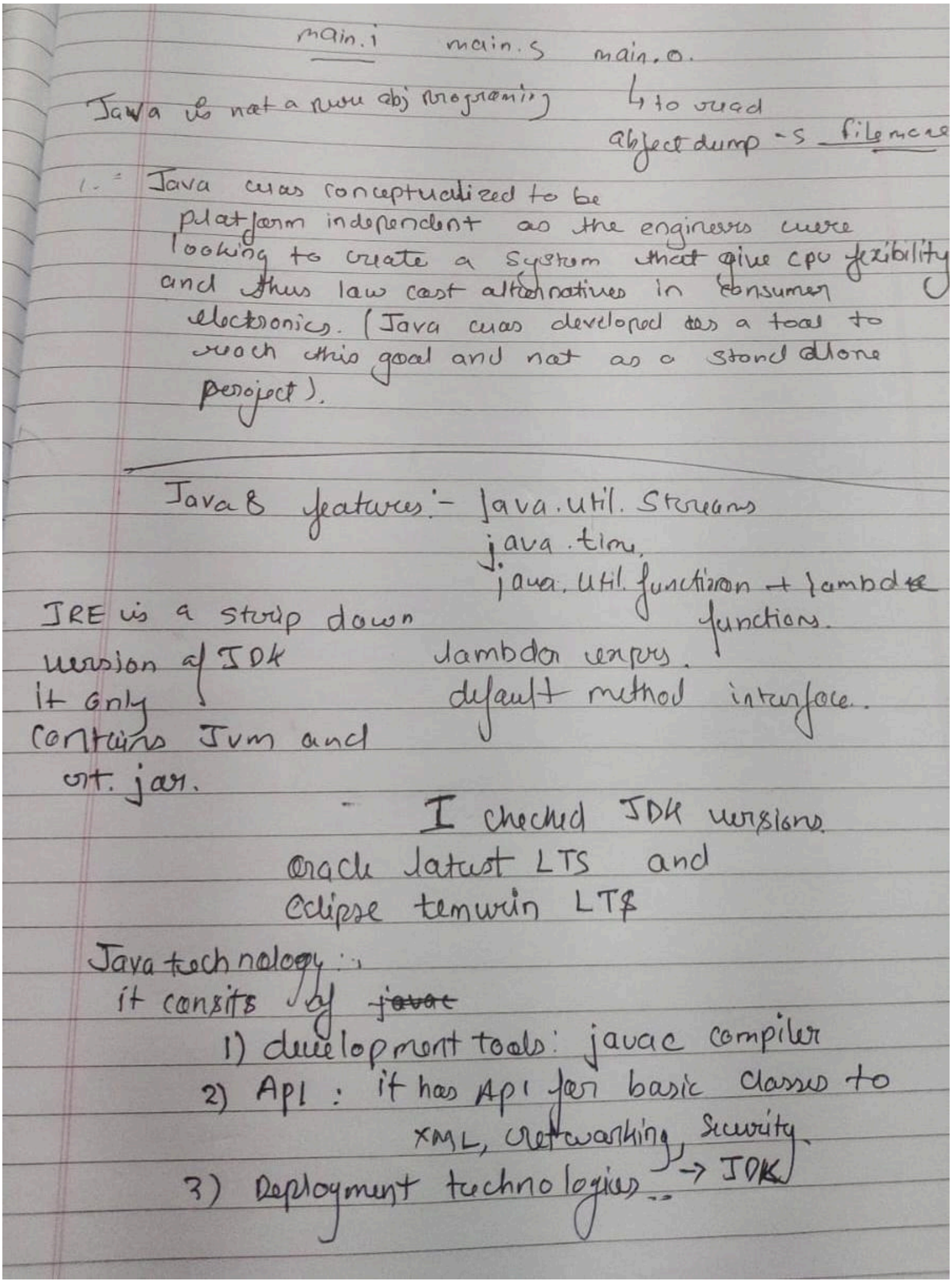
```
Compiled from "hello.java"

class Aysuh {

  Aysuh();

    Code:

       0: aload_0

       1: invokespecial #1                  // Method java/lang/Object."<init>":()V

       4: return


  public static void main(java.lang.String[]);

    Code:

       0: getstatic     #7                  // Field java/lang/System.out:Ljava/io/PrintStream;

       3: ldc           #13                 // String Hello,World!

       5: invokevirtual #15                 // Method
java/io/PrintStream.println:(Ljava/lang/String;)V

       8: return

}
```

### 7. Reading Assignment: The JVM Architecture Explained

- **Task**: Learn about how the Java Virtual Machine (JVM) works.
- **Link**: https://dzone.com/articles/jvm-architecture-explained

### 8. Reading Assignment: The Java Language Environment: Contents

- **Task**: Explore the content and features of the Java language environment.
- **Link**: https://www.oracle.com/java/technologies/language-environment.html



main.i main.s main.o.

Java is not a pure obj programing ↳ to read

abject dump -s filename

1.- Java was conceptualized to be platform independent as the engineers were looking to create a system that give cpu flexibility and thus low cost alternatives in consumer electronics. (Java was developed as a tool to reach this goal and not as a stand alone perojact).

Java 8 features:- Java.util. Streams
java.time.
java.util.function + lambda functions.
lambda expres.
default method interface.

JRE is a strip down version of JDK
it only contains Jvm and rt. jar.

I checked JDK versions.
Oracle latest LTS and
eclipse temurin LTS

Java technology:
it consits of ~~javac~~

1) development tools: javac compiler
2) Api : it has Api for basic classes to XML, networking, security.
3) Deployment technologies. → JDK