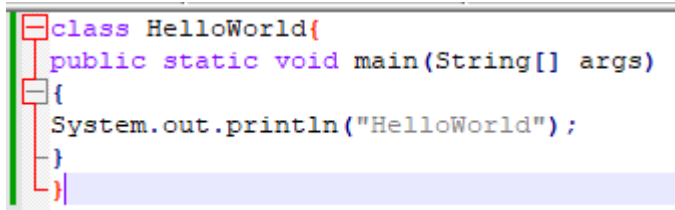


# Assignments 1

## Coding Assignments

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

A screenshot of a code editor with a light blue background. The code is written in Java and is color-coded: 'class' is red, 'HelloWorld' is black, '{' is red, 'public' is blue, 'static' is blue, 'void' is blue, 'main' is black, 'String[]' is black, 'args' is black, '}' is red, 'System.out.println' is black, and 'HelloWorld' is black. The code is as follows:

```
class HelloWorld{  
    public static void main(String[] args)  
    {  
        System.out.println("HelloWorld");  
    }  
}
```

- **Compile with Verbose Option:** Compile your Java file using the `-verbose` option with `javac`. Check the output.
- The `-verbose` option will provide detailed information about the compilation process, including the loading and processing of classes and the creation of `.class` files.

Command Prompt

```
D:\CdacAug24>javac -verbose HelloWorld.java
[parsing started SimpleFileObject[D:\CdacAug24\HelloWorld.java]]
[parsing completed 46ms]
[loading /modules/jdk.crypto.cryptoki/module-info.class]
[loading /modules/jdk.nio.mapmode/module-info.class]
[loading /modules/java.rmi/module-info.class]
[loading /modules/java.xml/module-info.class]
[loading /modules/jdk.jcmd/module-info.class]
[loading /modules/java.logging/module-info.class]
[loading /modules/jdk.accessibility/module-info.class]
[loading /modules/jdk.javadoc/module-info.class]
[loading /modules/jdk.httpserver/module-info.class]
[loading /modules/jdk.internal.vm.compiler/module-info.class]
[loading /modules/jdk.jstatd/module-info.class]
[loading /modules/java.base/module-info.class]
[loading /modules/jdk.internal.opt/module-info.class]
[loading /modules/jdk.jlink/module-info.class]
[loading /modules/jdk.internal.jvmsstat/module-info.class]
[loading /modules/jdk.crypto.ec/module-info.class]
[loading /modules/jdk.net/module-info.class]
[loading /modules/jdk.security.jgss/module-info.class]
[loading /modules/java.scripting/module-info.class]
[loading /modules/java.sql/module-info.class]
[loading /modules/jdk.naming.dns/module-info.class]
[loading /modules/java.datatransfer/module-info.class]
[loading /modules/java.transaction.xa/module-info.class]
[loading /modules/jdk.naming.rmi/module-info.class]
[loading /modules/jdk.security.auth/module-info.class]
[loading /modules/jdk.management/module-info.class]
[loading /modules/jdk.crypto.mscapi/module-info.class]
[loading /modules/jdk.internal.le/module-info.class]
```

- **Inspect Bytecode:** Use the `javap` tool to examine the bytecode of the compiled `.class` file. Observe the output.
- The `javap` output gives you an overview of the bytecode instructions used in the main method, such as loading constants (`ldc`), storing them (`astore`), and invoking methods (`invokevirtual`).

```
D:\CdacAug24>javap -c HelloWorld
Compiled from "HelloWorld.java"
class HelloWorld {
    HelloWorld();
        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 HelloWorld
            5: invokevirtual #15         // Method java/io/PrintStream.println:(Ljava/lang/String;)V
            8: return
}
```