

Lab 1



Session: 2021 – 2024

Submitted by:

Uswa Arif

2021-CS-77

Submitted To:

Laeq Khan Niazi

Department of Computer Science
University of Engineering and Technology
Lahore Pakistan

Task 1:

Part 1:

Compiler:

Compiler converts the programming language or any source language into the target language such as machine code. If there is any error in the source code, the compiler shows error and do not create exe file. Compiler executes the complete code and generate exe file. The code will be run through this exe file.

Compiler compiles the code into complete machine code or intermediate code.

Interpreter:

Interpreter converts the programming language or any source language into the target language such as machine code but it did not convert the code into intermediate code or complete machine code. It executes the code line by line and then gives output.

Difference:

Compiler generates the exe file and the output should be shown through this exe file but interpreter do not generates the exe file. It executes the code line by line and then shows output. Compiler executes the program fast then the interpreter.

When to Use in different situation:

- Compiler is used for the applications that require maximum speed, desktop application and application that require more protection. Moreover, compiler is used for the languages like C#, C++ and Rust etc.
- Interpreter is used for the applications that require feedbacks, web applications. Moreover, interpreter is used for the languages like Python, Ruby and JavaScript etc.

Part 2:

JIT (JUST IN TIME) Compilation:

JIT Compilation is the compilation that combines both the compiler and interpreter. The compiler compiles the important code together and but it will compiles it every time when the code is executed.

How it works:

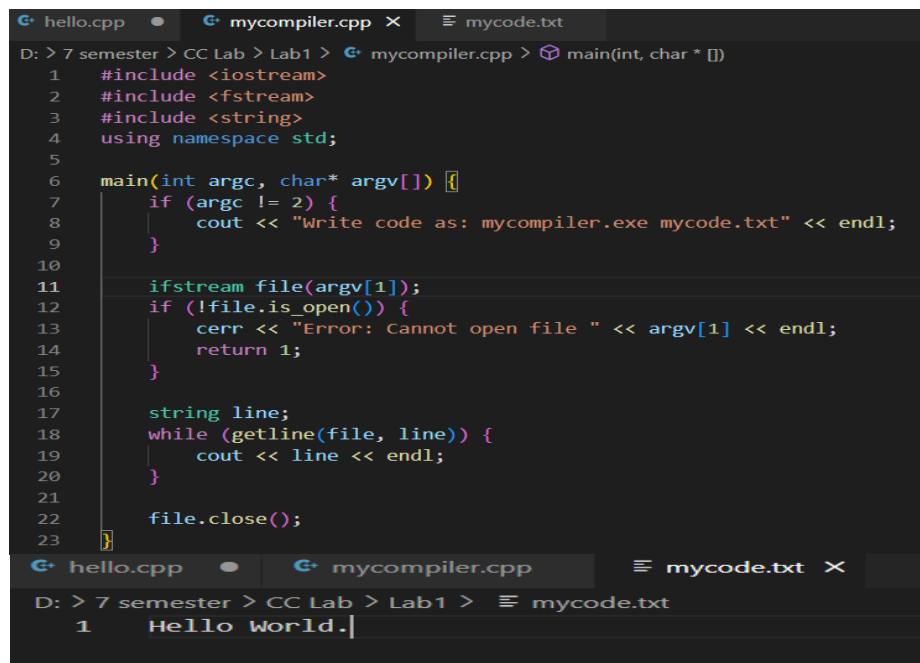
- Suppose there is a code written in common language, for example: C#, Java etc. that needs to be compiled into the machine code for execution.
- When the code starts running, the JIT watches each line of code. JIT finds the important code.
- When the JIT finds important code, it translates it into the machine code quickly.
- The next time when the code is executed, it uses the translated code that is translated by the JIT.

JIT is better?

- JIT is better when there is limited memory, and the applications that needed to start immediate when there are opened like web applications, games etc.
- JIT is not better where is security is important.

Task 2:

Code:



```
hello.cpp  mycompiler.cpp  mycode.txt
D: > 7 semester > CC Lab > Lab1 > mycompiler.cpp > main(int, char * [])
1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  using namespace std;
5
6  main(int argc, char* argv[]) {
7      if (argc != 2) {
8          cout << "Write code as: mycompiler.exe mycode.txt" << endl;
9      }
10
11     ifstream file(argv[1]);
12     if (!file.is_open()) {
13         cerr << "Error: Cannot open file " << argv[1] << endl;
14         return 1;
15     }
16
17     string line;
18     while (getline(file, line)) {
19         cout << line << endl;
20     }
21
22     file.close();
23 }
hello.cpp  mycompiler.cpp  mycode.txt
D: > 7 semester > CC Lab > Lab1 > mycode.txt
1  Hello World. |
```

Output:

```
D:\7 semester\CC Lab\Lab1>mycompiler.exe mycode.txt  
Hello World.
```