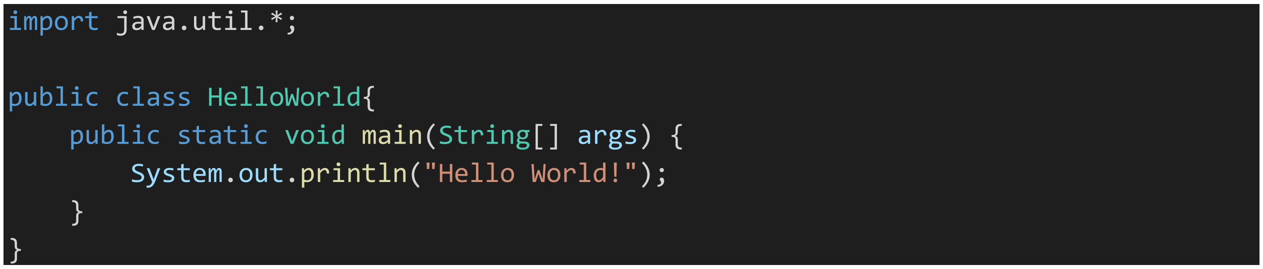
CS2263 Lab Report

Kohdy Nicholson

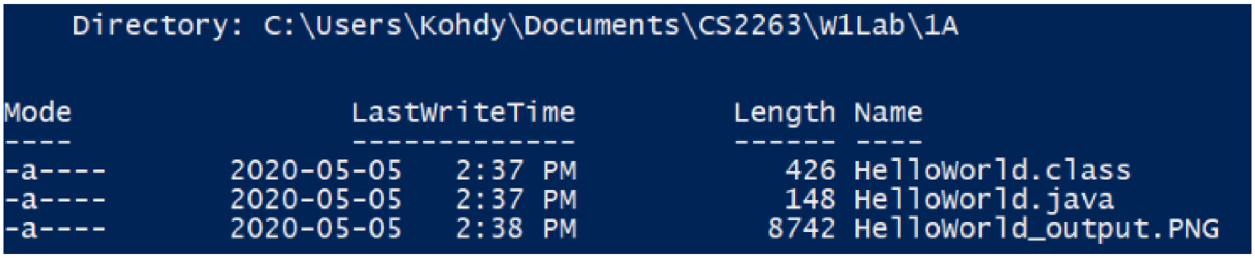
1A

Source:



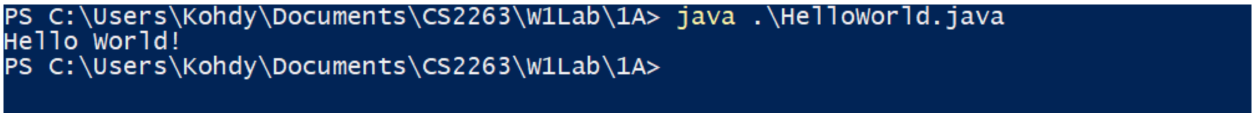
```
import java.util.*;

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

Directory listing:



```
    Directory: C:\Users\Kohdy\Documents\CS2263\W1Lab\1A


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2020-05-05   2:37 PM            426 HelloWorld.class
-a----        2020-05-05   2:37 PM            148 HelloWorld.java
-a----        2020-05-05   2:38 PM           8742 HelloWorld_output.PNG
```

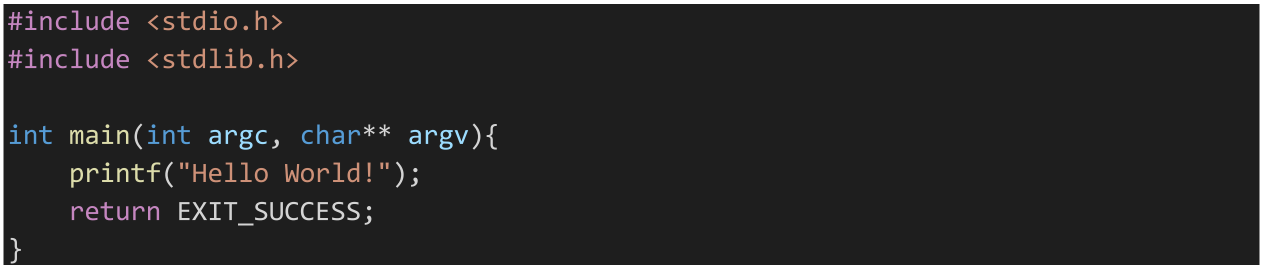
Output:



```
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\1A> java .\HelloWorld.java
Hello World!
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\1A>
```

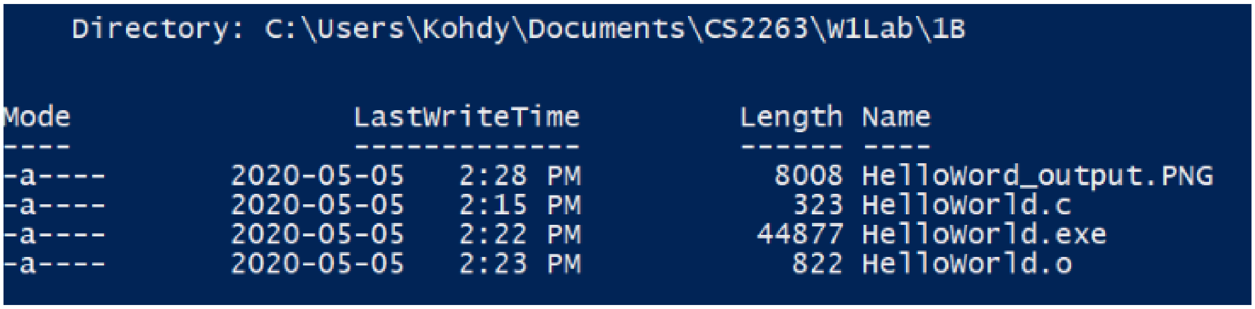
1B

Source:



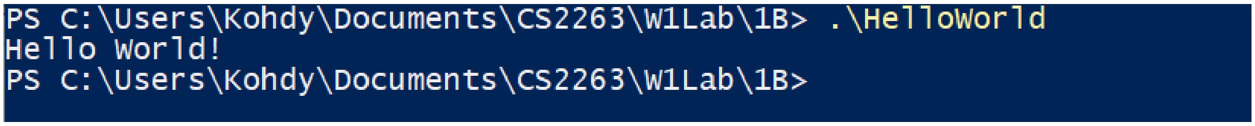
```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv){
    printf("Hello World!");
    return EXIT_SUCCESS;
}
```

Directory Listing:



```
    Directory: C:\Users\Kohdy\Documents\CS2263\W1Lab\1B


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2020-05-05   2:28 PM           8008 HelloWord_output.PNG
-a----        2020-05-05   2:15 PM            323 HelloWorld.c
-a----        2020-05-05   2:22 PM          44877 HelloWorld.exe
-a----        2020-05-05   2:23 PM            822 HelloWorld.o
```

Output:



```
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\1B> .\HelloWorld
Hello World!
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\1B>
```

1C

The difference between the compilation of Java and C is the steps in the compilation process. In C, we first compile the source code to object code. This object code is machine language code designed for the specific operating system it is running on. Depending on which header files are included, the object code will then need to be sent through a linker which will link other object code to the source object code. After all the necessary library's object code are linked to the source object code, we produce an executable file.

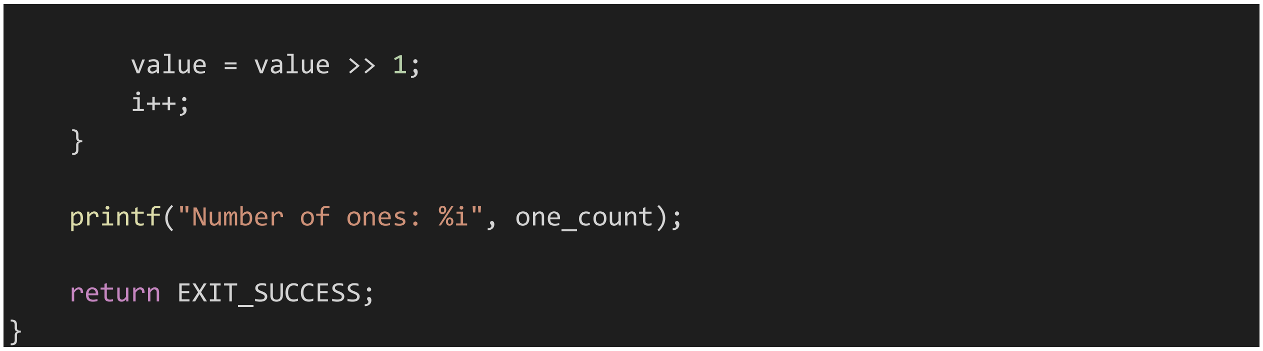
In Java, we compile the source code to an intermediate language known as bytecode. Bytecode is lower level code compared to java, and is interpreted by the java virtual machine (JVM) inside the java runtime environment (JRE). The difference between bytecode and object code is, bytecode works cross-platform while object code does not.

Object code interacts directly with the platform it was built on, giving you more control over the internal functions of the system. On the other hand, the operating system and bytecode are separated by the JVM. The JVM will handle standard and basic behavior such as memory cleanup, also known as garbage collection. Therefore, the same java code can be compiled are run on multiple platforms. This is because the code stays the same, however the JRE and JVM will change depending on which platform it is installed on.

2

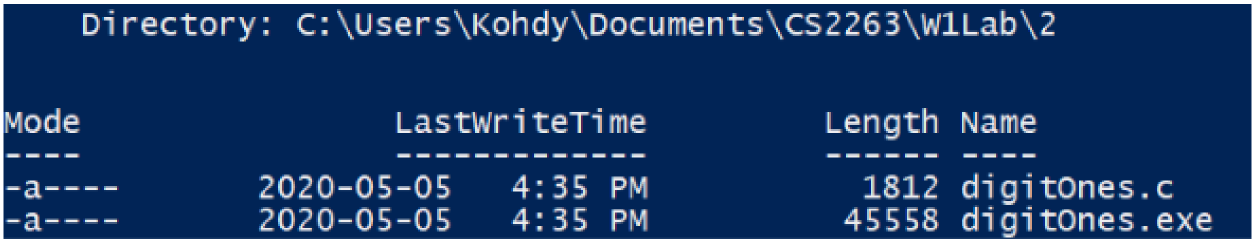
Source:

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>

int main(int argc, char** argv){
    int value;
    int iErr;
    int bits, bits_per_byte, bytes;
    int one_count;

    printf("Integer to examine (integer 'n' must be '0 < n < 2147483648'): ");
    iErr = scanf("%i",&value);

    // Check for NaN inputs
    if(iErr != 1){
        printf("Unable to read the value.\n");
        return EXIT_FAILURE;
    }

    // Check for inputs outside limits.
    if(value < 1 || value > INT_MAX){
        printf("Invalid integer value. Please enter an integer 'n' which follows '0 < n < 2147483648'.\n");
        return EXIT_FAILURE;
    }

    bits_per_byte = CHAR_BIT; //CHAR_BIT is how many bits are in a char (1 byte) for the system (typically will be 8 bits)
    bytes = sizeof(value); //sizeof(value) gets the amount bytes allocated in memory for the value variable type

    bits = bits_per_byte * bytes; //We multiple bits_per_byte * bytes to get the amount of bits for the value variable

    // Loop through the bits in value while using bitwise operators for anding and shifting to determine the amount of 1's
    one_count = 0;
    for(int i = 0; i < bits; i++){
        int bitwise_result = value & 1;
        if(bitwise_result == 1){
            one_count++;
        }
```



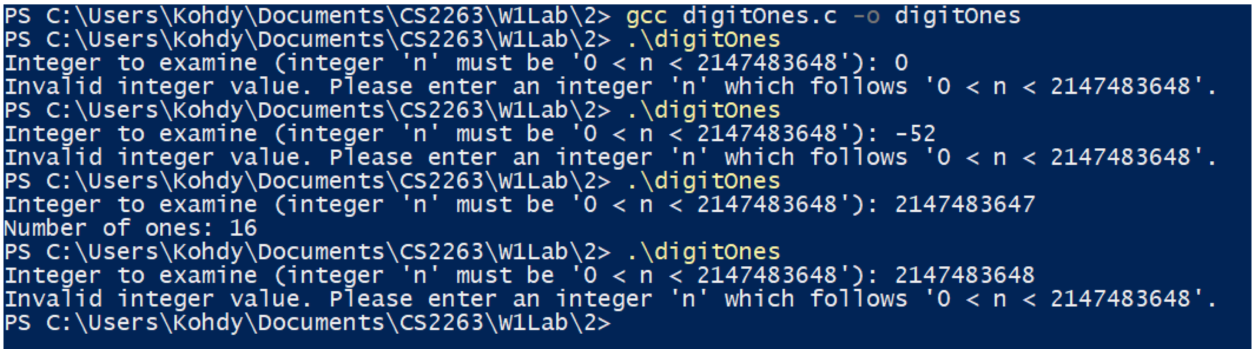
```
        value = value >> 1;
        i++;
    }

    printf("Number of ones: %i", one_count);

    return EXIT_SUCCESS;
}
```

Directory Listing:



```
    Directory: C:\Users\Kohdy\Documents\CS2263\W1Lab\2


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2020-05-05   4:35 PM           1812 digitOnes.c
-a----        2020-05-05   4:35 PM          45558 digitOnes.exe
```

Output:



```
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\2> gcc digitOnes.c -o digitOnes
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\2> .\digitOnes
Integer to examine (integer 'n' must be '0 < n < 2147483648'): 0
Invalid integer value. Please enter an integer 'n' which follows '0 < n < 2147483648'.
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\2> .\digitOnes
Integer to examine (integer 'n' must be '0 < n < 2147483648'): -52
Invalid integer value. Please enter an integer 'n' which follows '0 < n < 2147483648'.
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\2> .\digitOnes
Integer to examine (integer 'n' must be '0 < n < 2147483648'): 2147483647
Number of ones: 16
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\2> .\digitOnes
Integer to examine (integer 'n' must be '0 < n < 2147483648'): 2147483648
Invalid integer value. Please enter an integer 'n' which follows '0 < n < 2147483648'.
PS C:\Users\Kohdy\Documents\CS2263\W1Lab\2>
```