|  |
| --- |
| Computer Programming |
| Course Code: CSC 113 |
| **Assignment 1** |
| Submission Instructions:   * The assignment is to be submitted individually. * Submission deadline is 18 October 2022 |

* For programming questions, include the code and screenshot of output
* Make sure to use this file to submit your solutions.
* **\*Remember to follow best practices, including comments.**

Submission by:

**Name:** Saad Ahmad

**Enrollment Number:** 01-134222-130

**Problem 1.**

You have studied that C++ uses a “Compiler” to process your code. There are some programming languages which do not have a compiler, instead they have an “Interpreter”. Study in detail about compiler-based languages and interpreter-based languages

Structure your answer into following components:

1. Examples of compiler-based languages and how do they work.
2. Examples of interpreter-based languages and how do they work.

**Compiler-based Languages**

**C++:**

C++ is a compiler-based language. Each C++ source file needs to be compiled into an object file. The object files resulting from the compilation of multiple source files are then linked into an executable.

There are three steps to the compilation of a C++ program

1. Preprocessing
2. Compilation
3. Linking

**C:**

C is also a compiler-based language. The code can be written in any text editor but an IDE should be preferred as it makes the process easier. The compiler checks for any errors and if there are none then it creates an executable file for the computer to run.

**Interpreter-based Languages**

**JavaScript:**

JavaScript is an interpreter-based language. It has no compilation step. It is interpreted directly by the browser. Each line of code is interpreted one by one. The code is passed through a parser which checks the code for any error and if none are found it creates a data structure called AST. After creating the AST, the JavaScript code is converted into machine code. Then finally it is sent for execution.

**Python:**

Python is also an interpreter-based language. Instead of translating source code into machine code like C++ and C, Python code is translated into bytecode which can be executed by an interpreter.

**Problem 2.**

Write C++ program that displays the output as given in the video at this link ( <https://youtu.be/IWbQDHpPrJE> )

Include complete code and screenshot of the output

Hint: You will need to use **Sleep()** function defined in **windows.h** header file.

**Code:**

#include <iostream>

#include <windows.h> // This preprocessor directive includes the sleep() function.

using namespace std;

int main(){

Sleep(500);

cout << "Installation at 1%";

//\r is used to move the cursor to the beginning of the line without advancing to the next line

Sleep(600);

cout << "\rInstallation at 10%";

Sleep(600);

cout << "\rInstallation at 20%";

Sleep(600);

cout << "\rInstallation at 30%";

Sleep(600);

cout << "\rInstallation at 40%";

Sleep(600);

cout << "\rInstallation at 50%";

Sleep(600);

cout << "\rInstallation at 60%";

Sleep(600);

cout << "\rInstallation at 70%";

Sleep(600);

cout << "\rInstallation at 80%";

Sleep(600);

cout << "\rInstallation at 90%";

Sleep(1000);

cout << "\rInstallation at 100%";

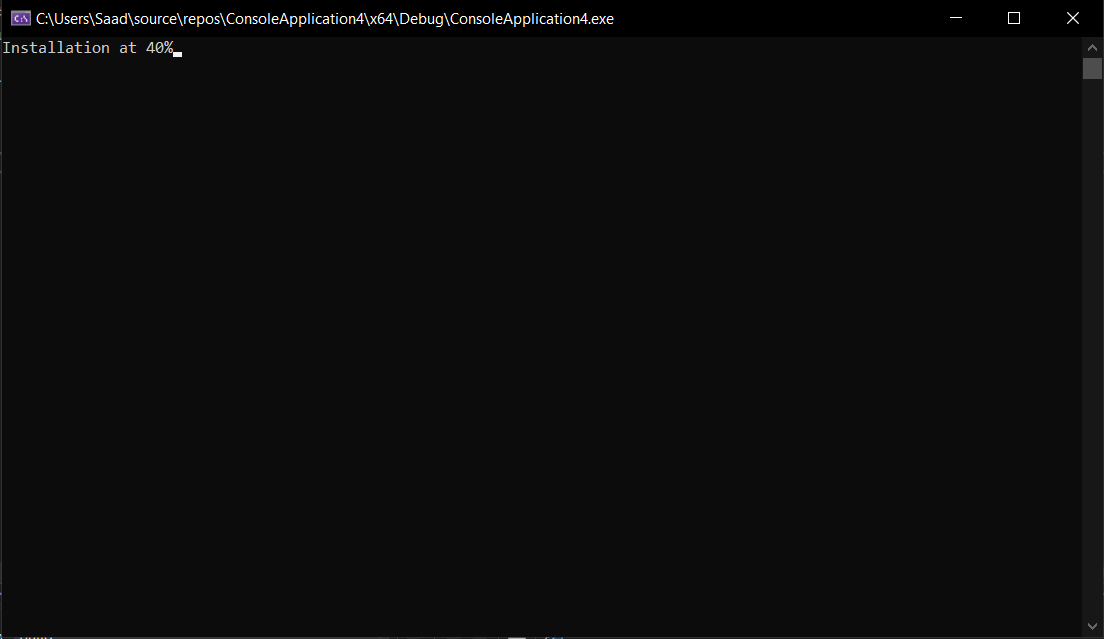
Sleep(800);

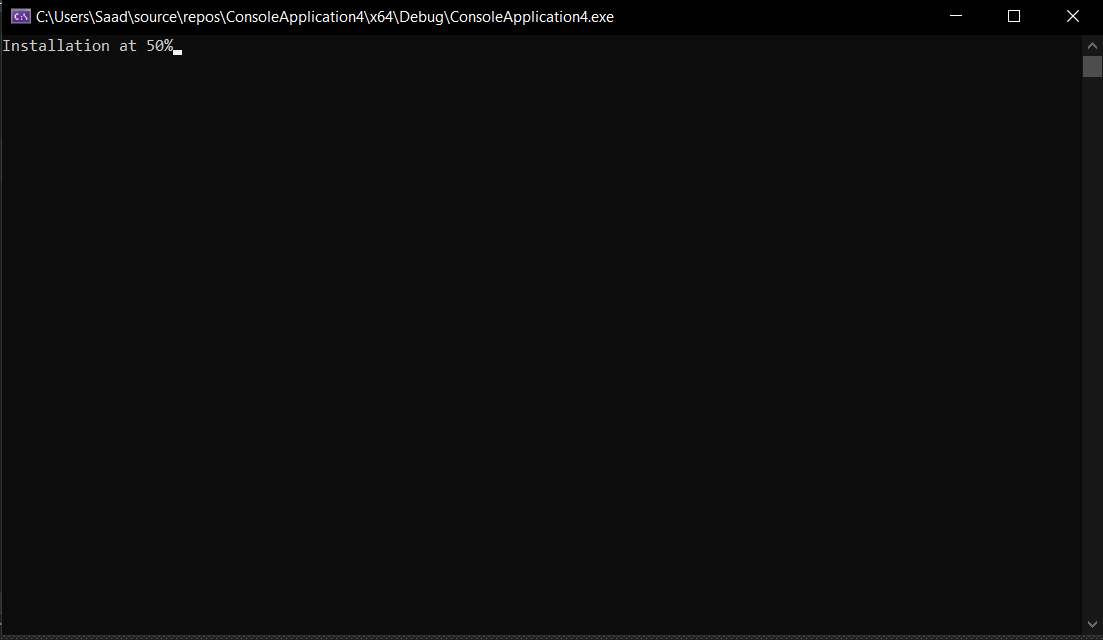
cout << "\rInstallation completed sucessfully" << endl;

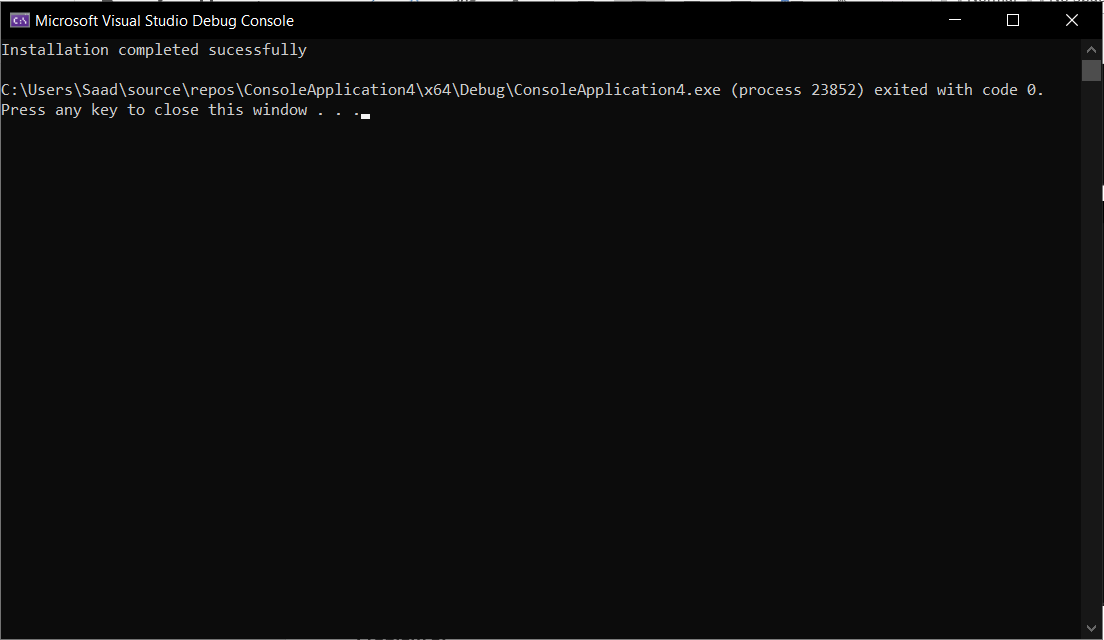
return 0;

}

**Output:**







**Problem 3.**

Your code has an integer variable containing the value 87. Use C++ to convert a decimal number to binary number representation. Include complete code and screenshot of the output.

Hint: You will need to use Modulous (%) and Division (/) Operators.

**Code:**

#include <iostream>

using namespace std;

int main() {

//The number 87 is divided 6 times until the answer is one

int a = 87;

int b = a / 2;

int c = b / 2;

int d = c / 2;

int e = d / 2;

int f = e / 2;

int g = f / 2;

//The mod of a,b,c,d,e,f,g is printed backwards so we get the binary representation of 87

cout << g % 2;

cout << f % 2;

cout << e % 2;

cout << d % 2;

cout << c % 2;

cout << b % 2;

cout << a % 2;

return 0;

}

**Output:**

