C
Computer Science

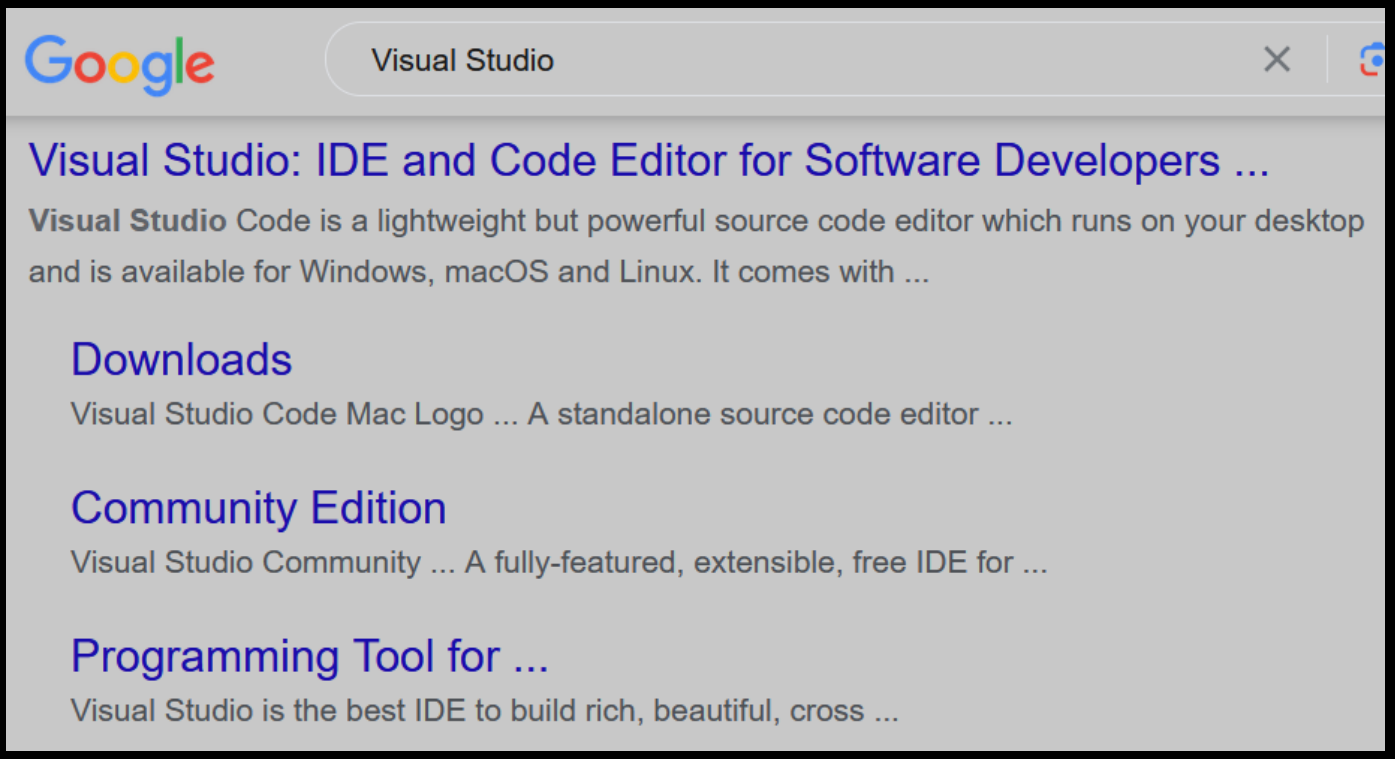
by

Christopher Andrew Topalian

Christopher Topalian

Dedicated to God the Father

Download Visual Studio - Search Google



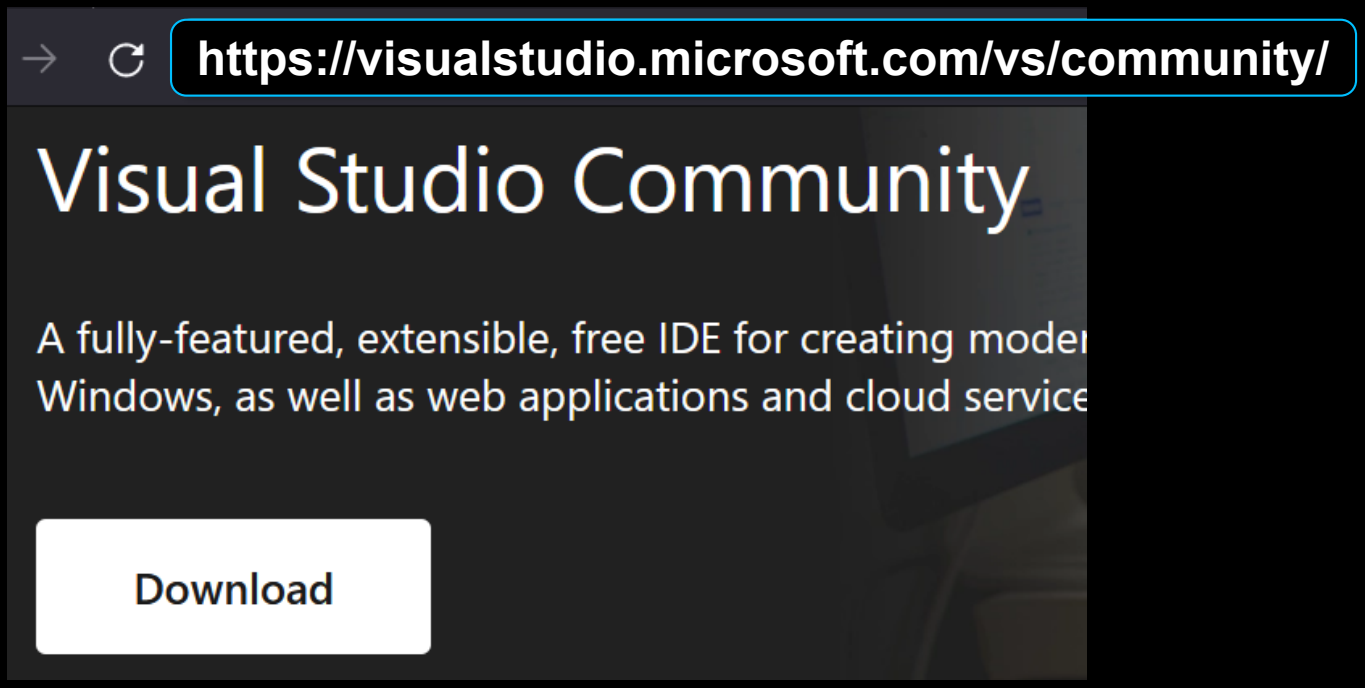
We Go To: google.com

We Search for: Visual Studio

We Left Click on: Downloads

Or, we can go directly
to the Visual Studio website
as shown on the next page.

Download Visual Studio - Directly from Website



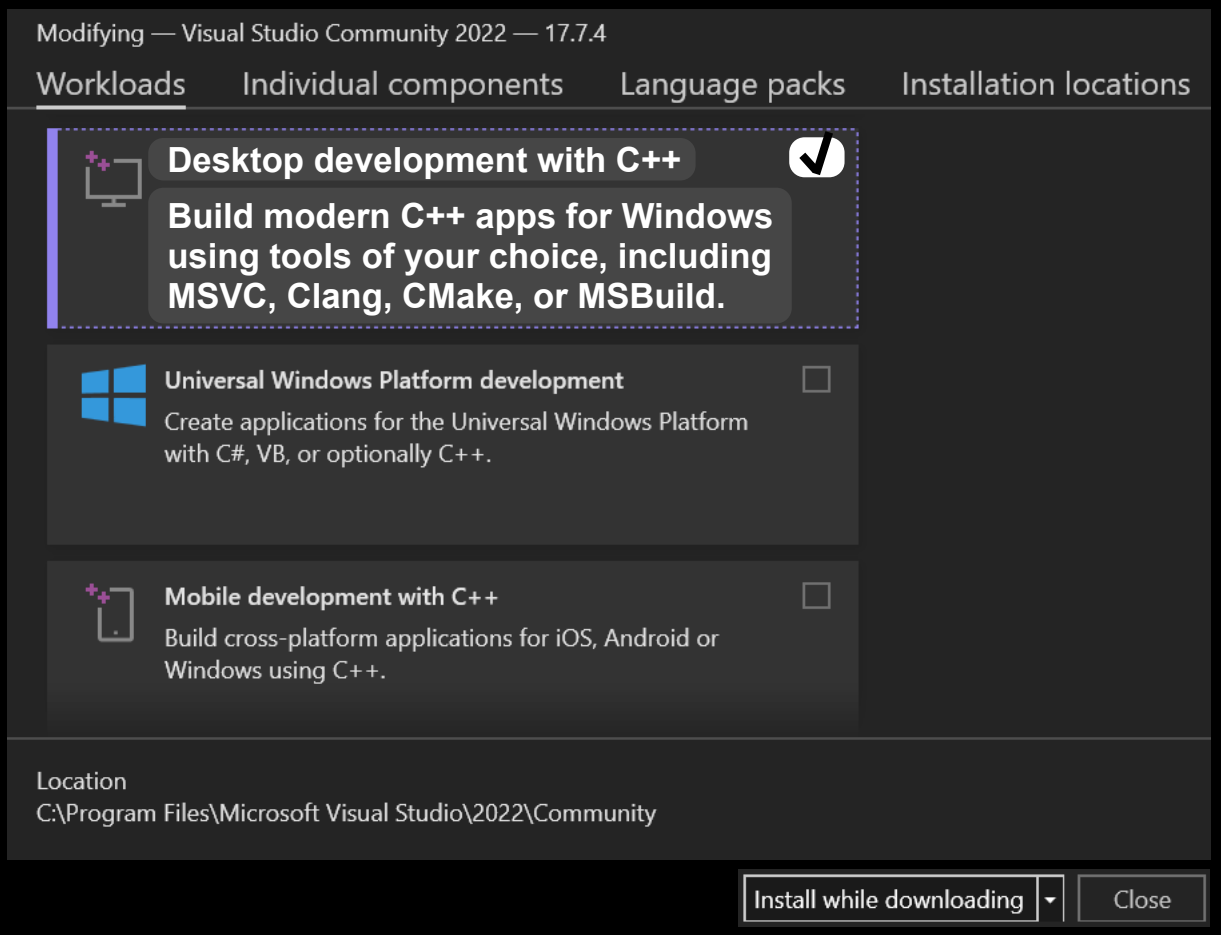
We Go To:
https://visualstudio.microsoft.com/vs/community

We Download: **Visual Studio Installer**

We Go To our Downloads Folder and:
Double Left Click the Install file to Install it.

After it is installed, we can open VS Studio.
Once open, we can then install the C++ package, which has in it, the ability to create C apps too.

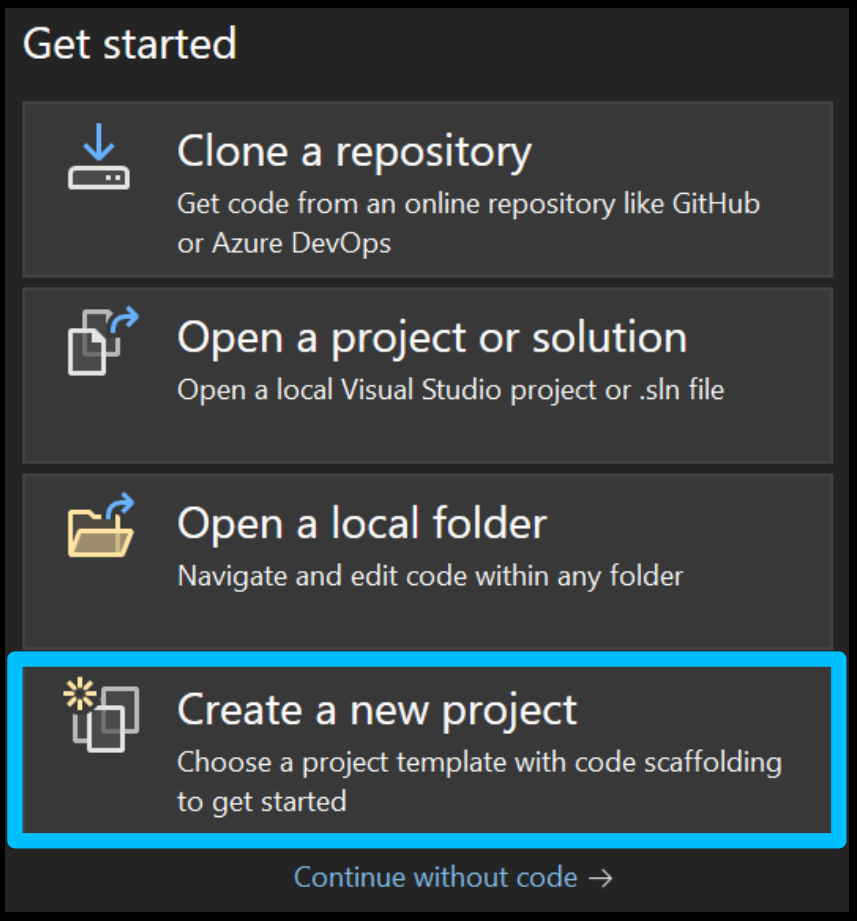
We Download and Install:
Desktop development with C++



We Put a Checkmark in the box
and then Left Click the
Install while downloading button

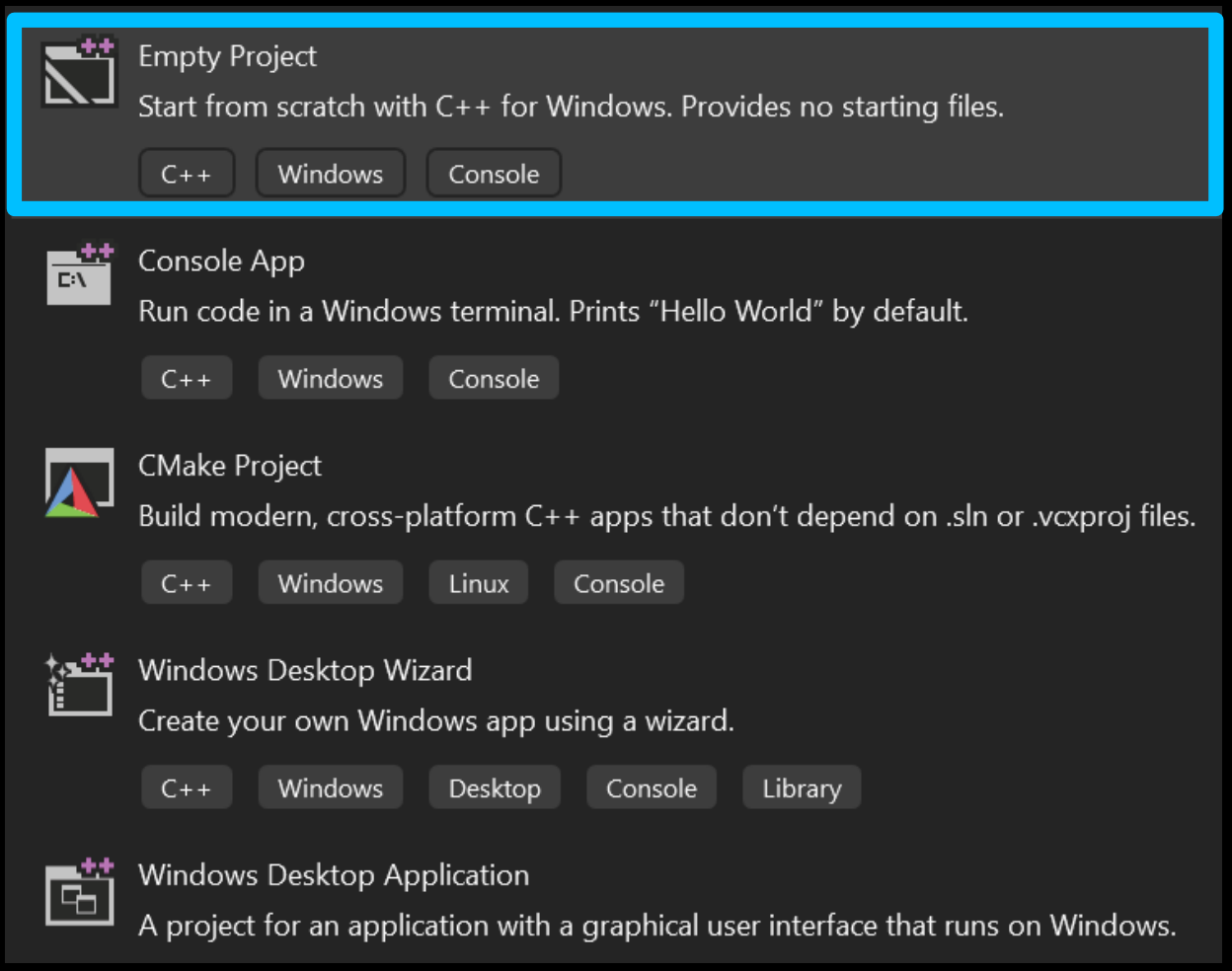
This will download and install the ability to use Visual Studio to create C++ Desktop Applications, but also, C applications too.

Create a New Project

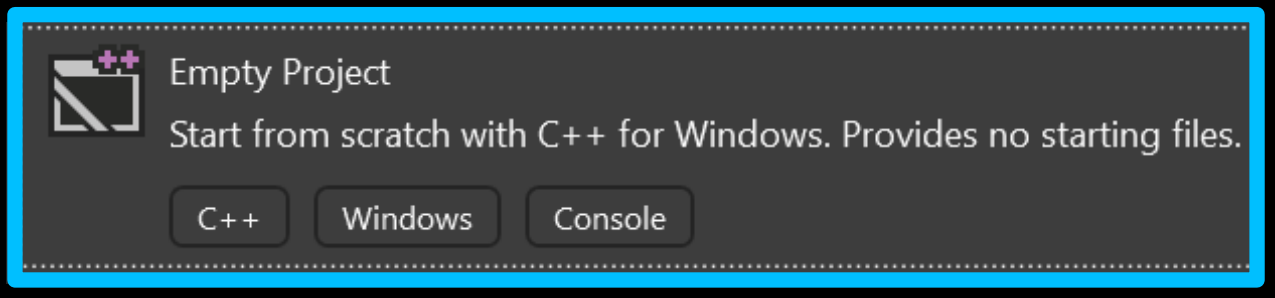


We Choose: Create a new project

Choices for a New Project

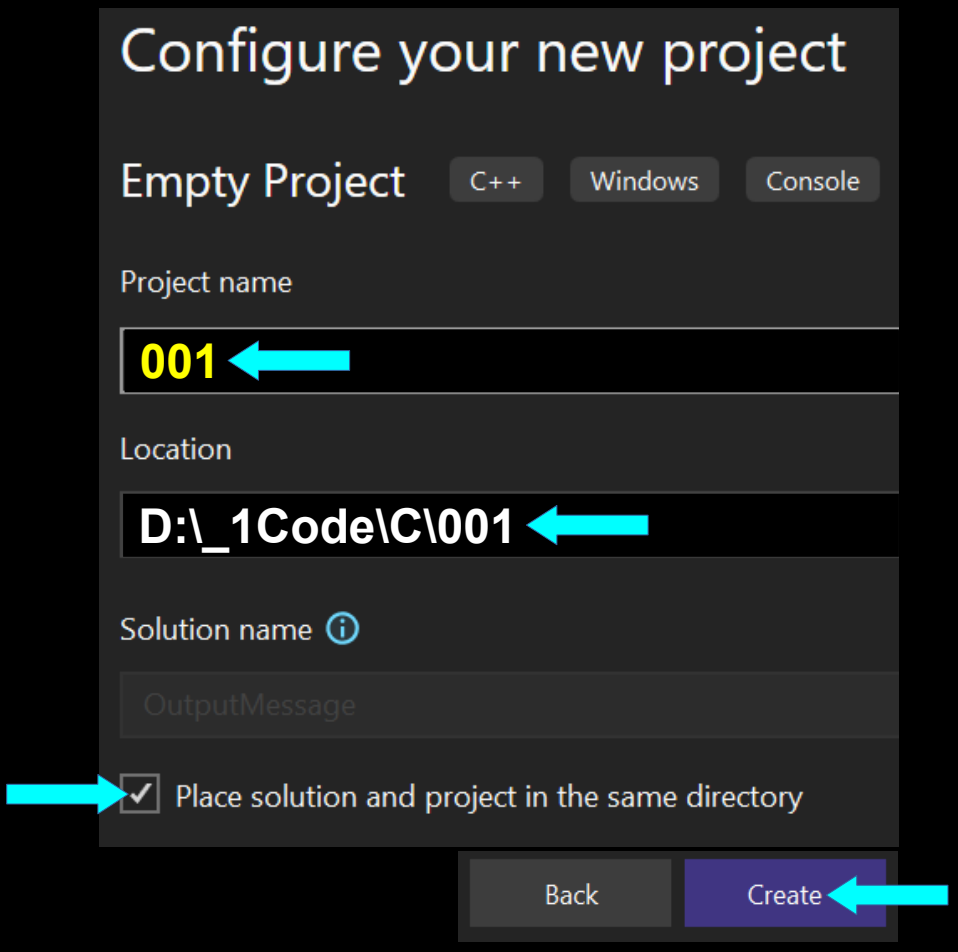


We Left Click: Empty Project



We Left Click: Next Button

Project Name - 001



We name our first project as: 001

We put a Checkmark in: Place solution and project in the same directory

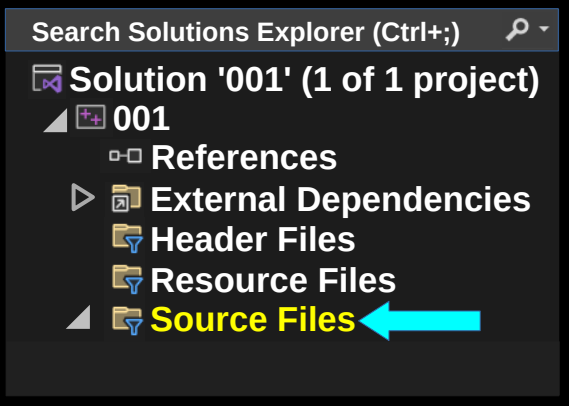
We Left Click: Create Button

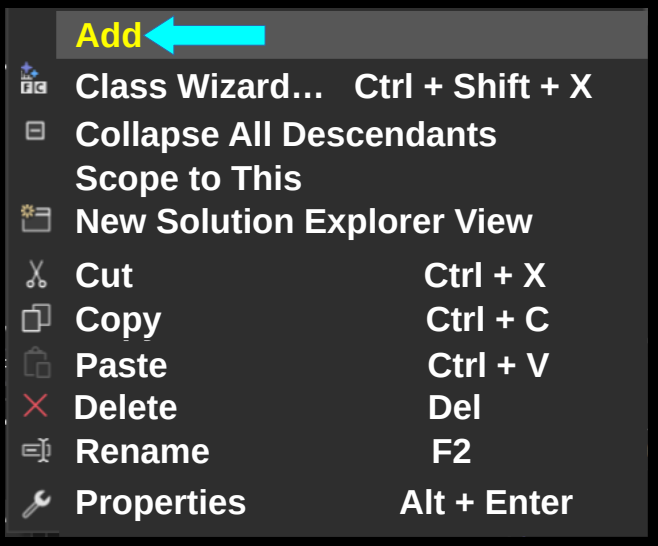
Creating our main.c file in Source Files Folder

We Right click on: Source Files Folder

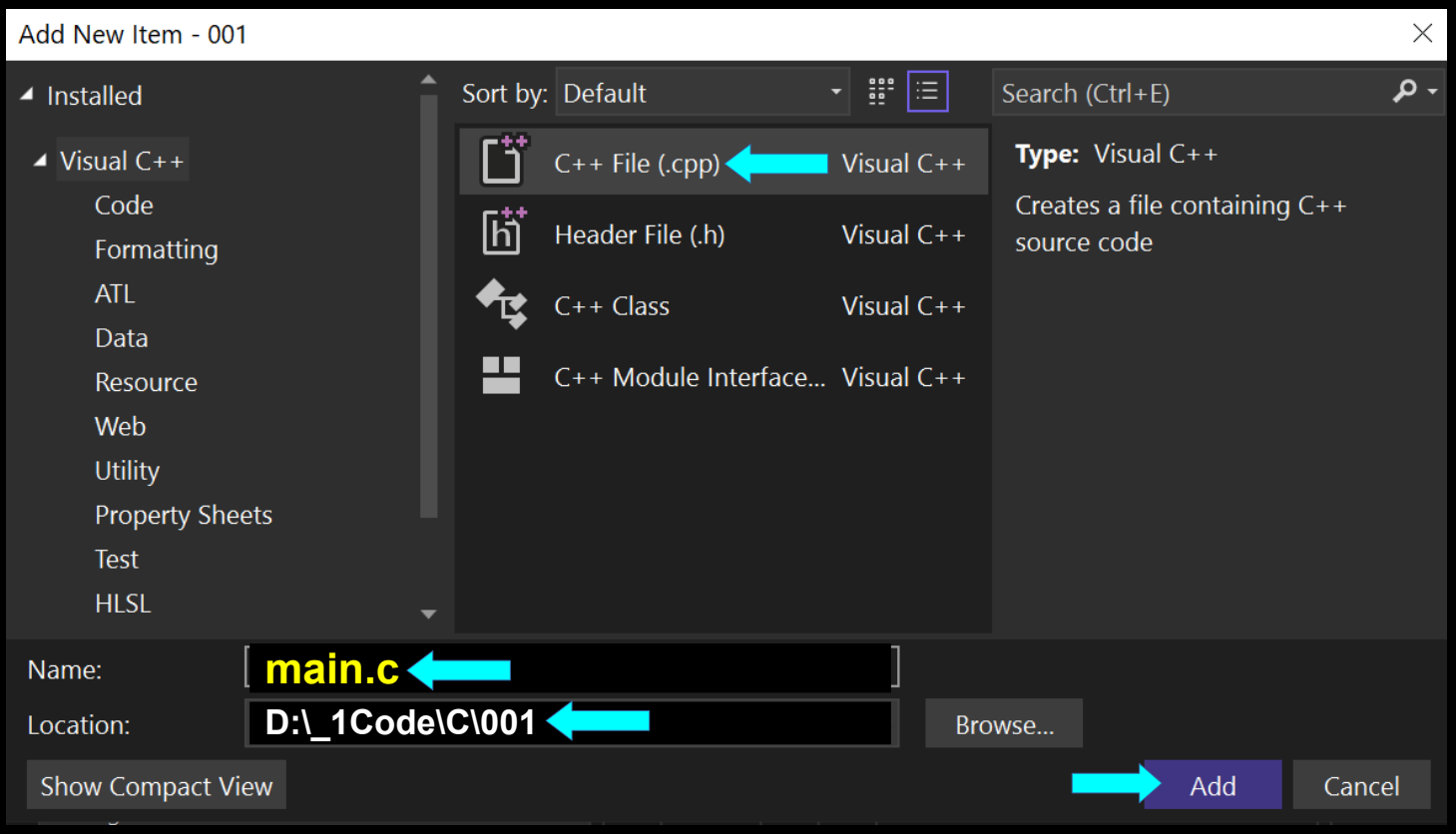
We Choose: Add

We Choose: New Item...





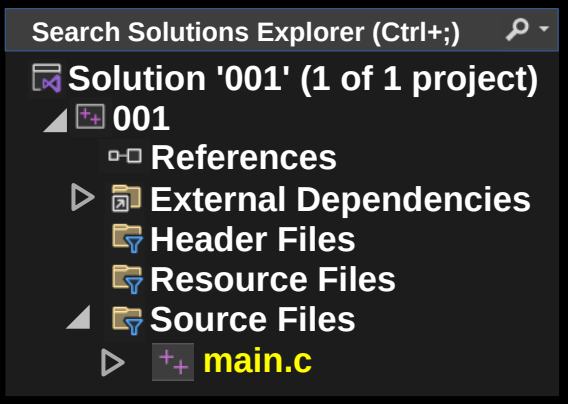
We Choose: C++ File (.cpp)



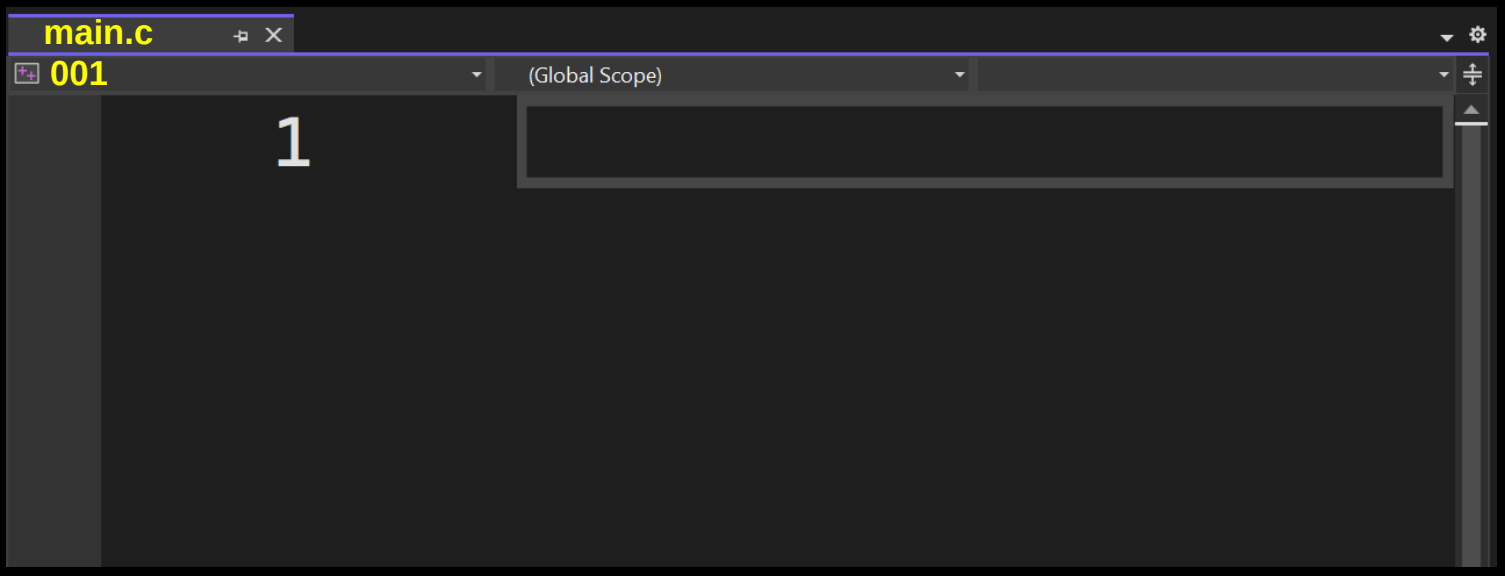
We Name our File: main.c

We Left Click: Add button

We see our created file: main.c



main.c is now open



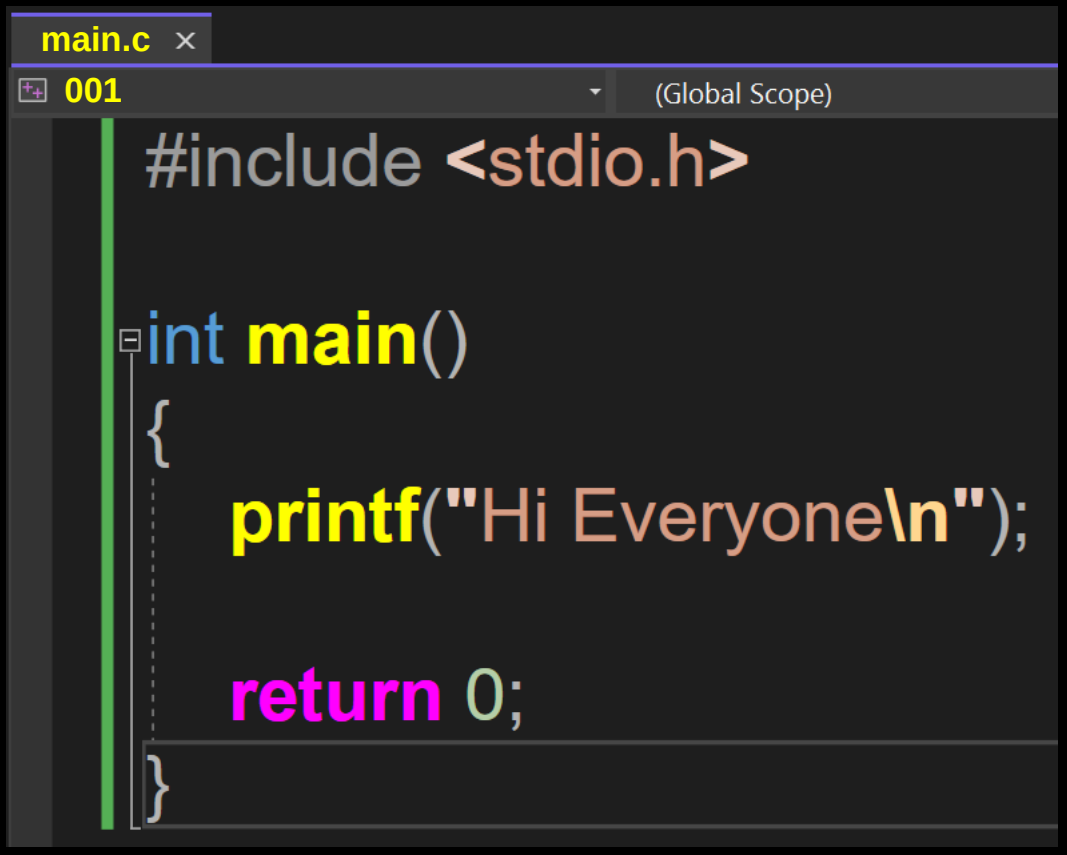
We can now: Type our Code :-)

We make Bigger Font by:
Control + Scroll Wheel Forward

We make Smaller Font by:
Control + Scroll Wheel Backward

main.c Code - Screenshot

Here is a screenshot of: Our C Code

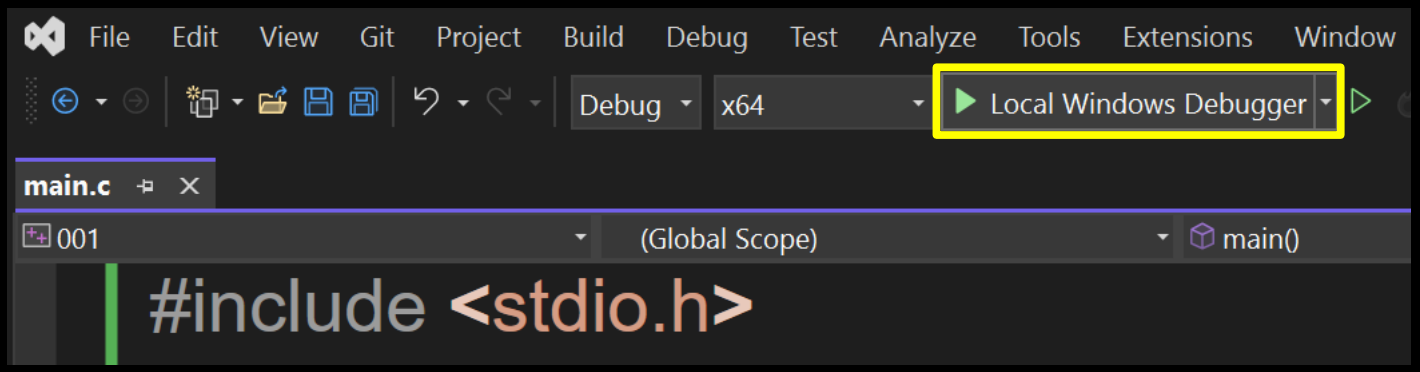


On the next page
we show the same code,
but with better font.

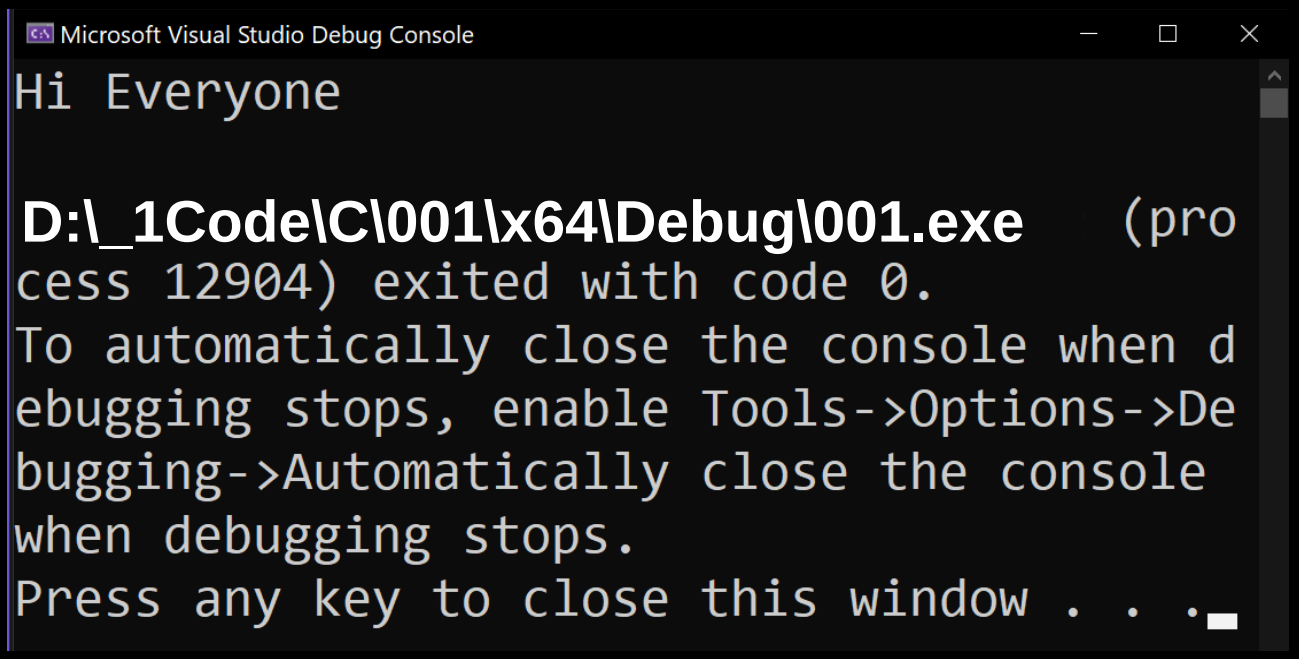
```c
// Outputting Text
// main.c

#include <stdio.h>

int main()
{
    printf("Hi Everyone\n");

    return 0;
}
```

// Building and Running our App

We Left Click on: Local Windows Debugger



Our app opens in the Debug Console Window, with the message of: Hi Everyone



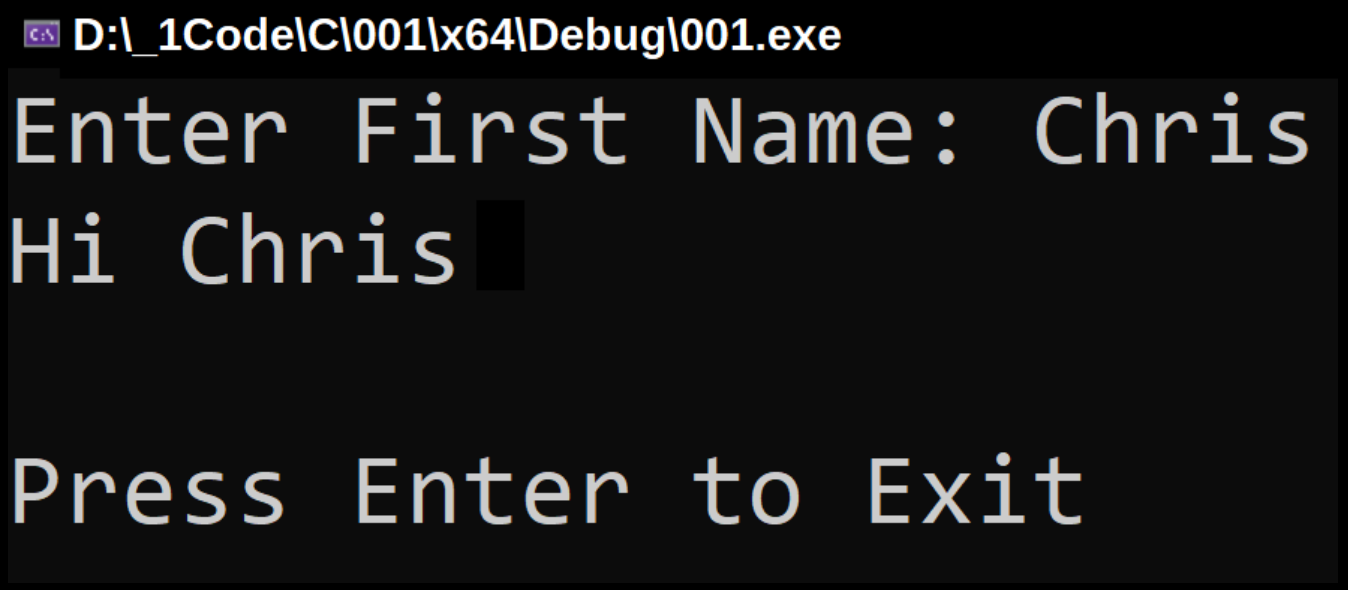
We make Bigger Console Font by:
Control + Scroll Wheel Forward

We make Smaller Console Font by:
Control + Scroll Wheel Backward

```c
// Outputting Text, Exit by Pressing Enter
// main.c

#include <stdio.h>

int main()
{
    printf("Hi Everyone\n");

    printf("Press Enter to Exit\n");

    // wait for user to press Enter
    getchar();

    return 0;
}
```

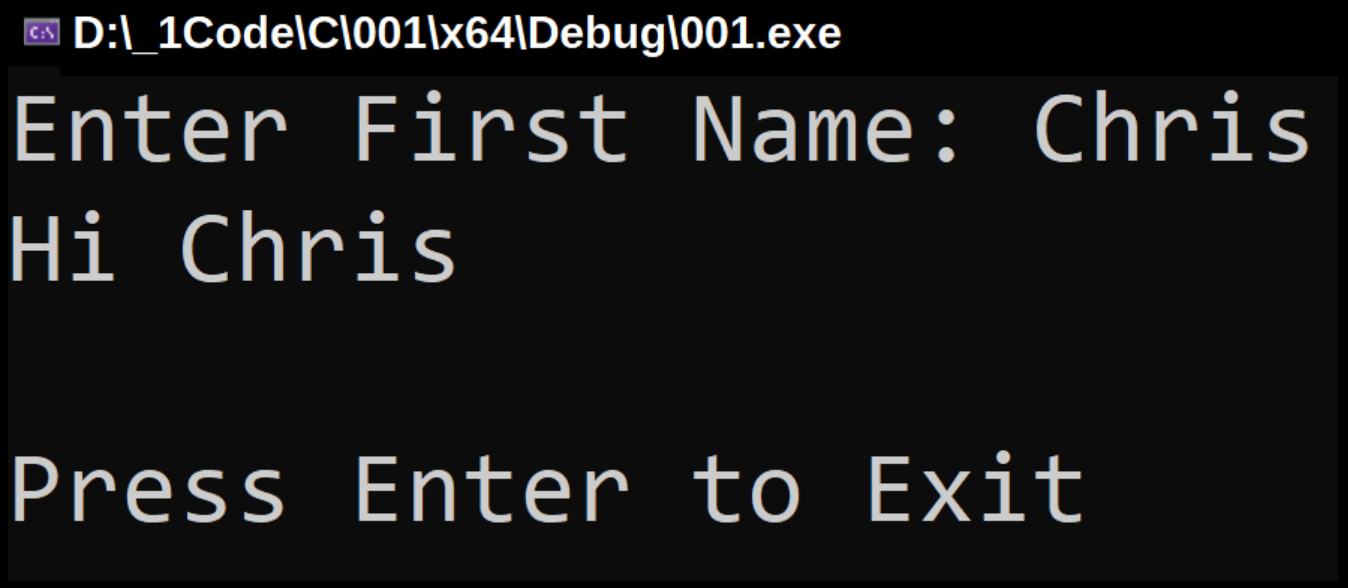
```c
// Input from user
// main.c

#include <stdio.h>

int main()
{
    // max name is 100 chars + null terminator
    char name[101];

    printf("Enter First Name: ");

    // read input from user and prevent buffer overflow
    scanf_s("%s", name, (unsigned int)sizeof(name));

    printf("Hi %s\n", name);

    printf("\nPress Enter to Exit\n");

    // remove newline char left in input buffer
    getchar();

    // wait for user to press Enter
```

```
    getchar();

    return 0;
}
```



```
// Custom Function - askName
// main.c

#include <stdio.h>

void askName(char* name)
{
    printf("Enter First Name: ");

    // read input from user and prevent buffer
overflow
    scanf_s("%s", name, (unsigned
int)sizeof(name));
}

int main()
{
    // max name is 100 chars + null terminator
    char userName[101];

    askName(userName);
    printf("Hi %s\n", userName);

    printf("\nPress Enter to Exit\n");
```

```
    // remove newline char left in input buffer by scanf_s
    getchar();

    // wait for user to press Enter
    getchar();

    return 0;
}
```



```c
// Custom Function - consoleLog
// main.c

#include <stdio.h>

void consoleLog(const char *message)
{
    printf("%s\n", message);
}

int main()
{
    consoleLog("Hi Everyone");

    printf("Press Enter to Exit\n");

    // wait for user to press Enter
    getchar();

    return 0;
}
```

Header File - We define our function in a header file for easy use

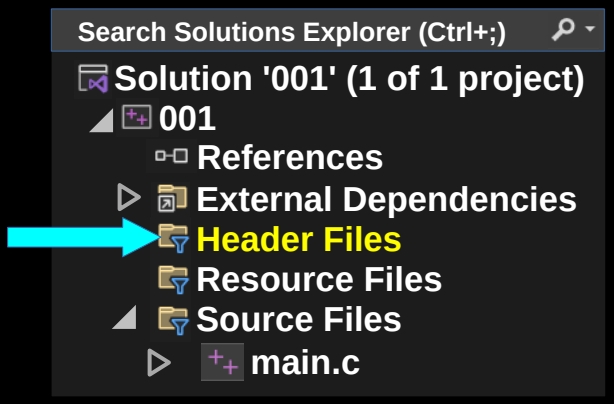
Instead of pasting this useful function in every script in our application, we will instead type it once in a header file and put it in the Header Files Folder.

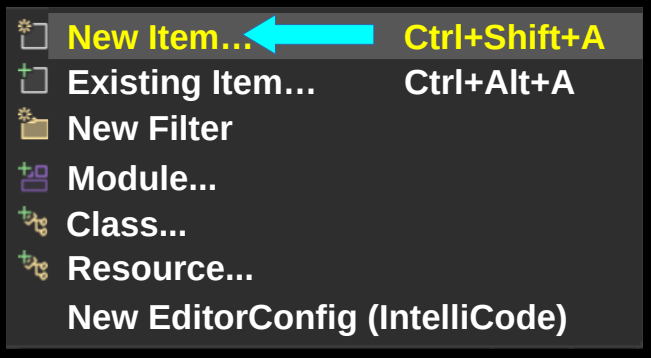
Using a header file is easier, because we place the header files in the Header Files folder and then include that header file with a reference in our main.c and other files.

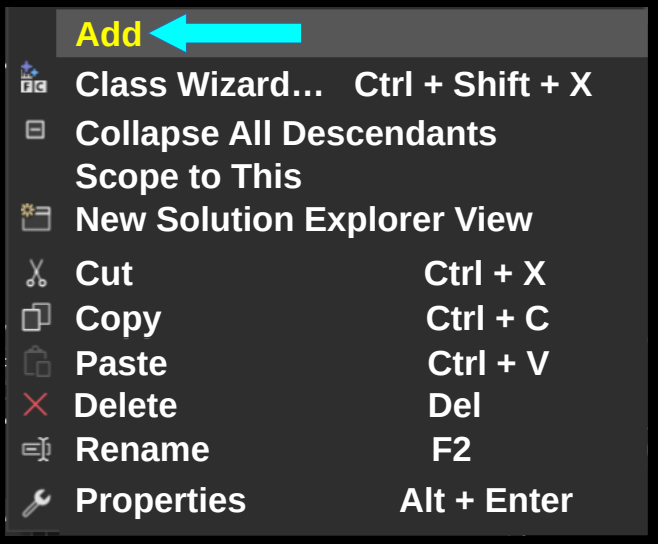
In our header file, we type the terms
#ifndef
and
#define
to designate that it will be used in other files.

Header File - Add - New Item

We right click on: Header Files folder



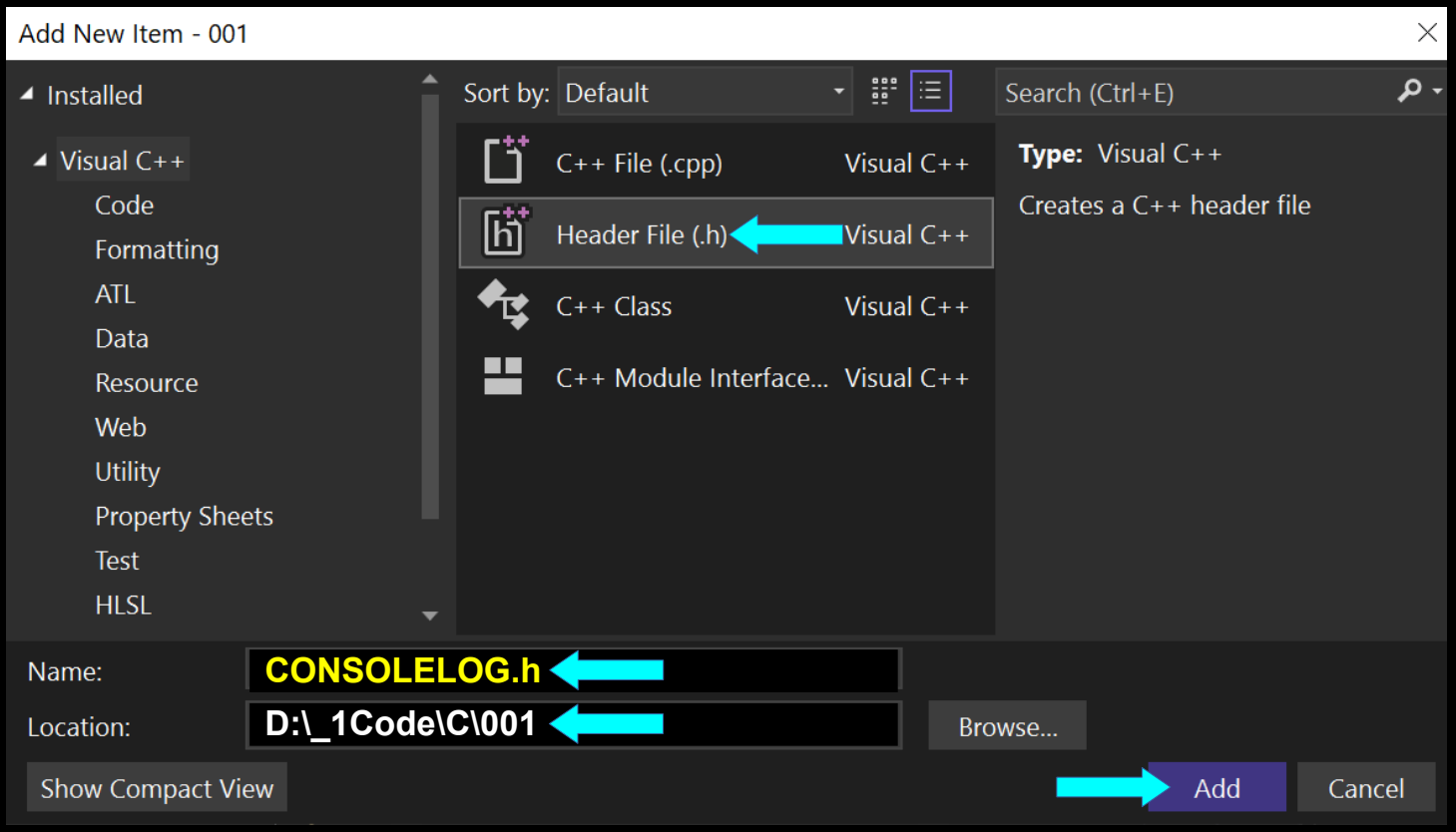




We choose: Add

We choose: New Item

We choose: Header File .h



We name it: CONSOLELOG.h

We Left Click: Add button

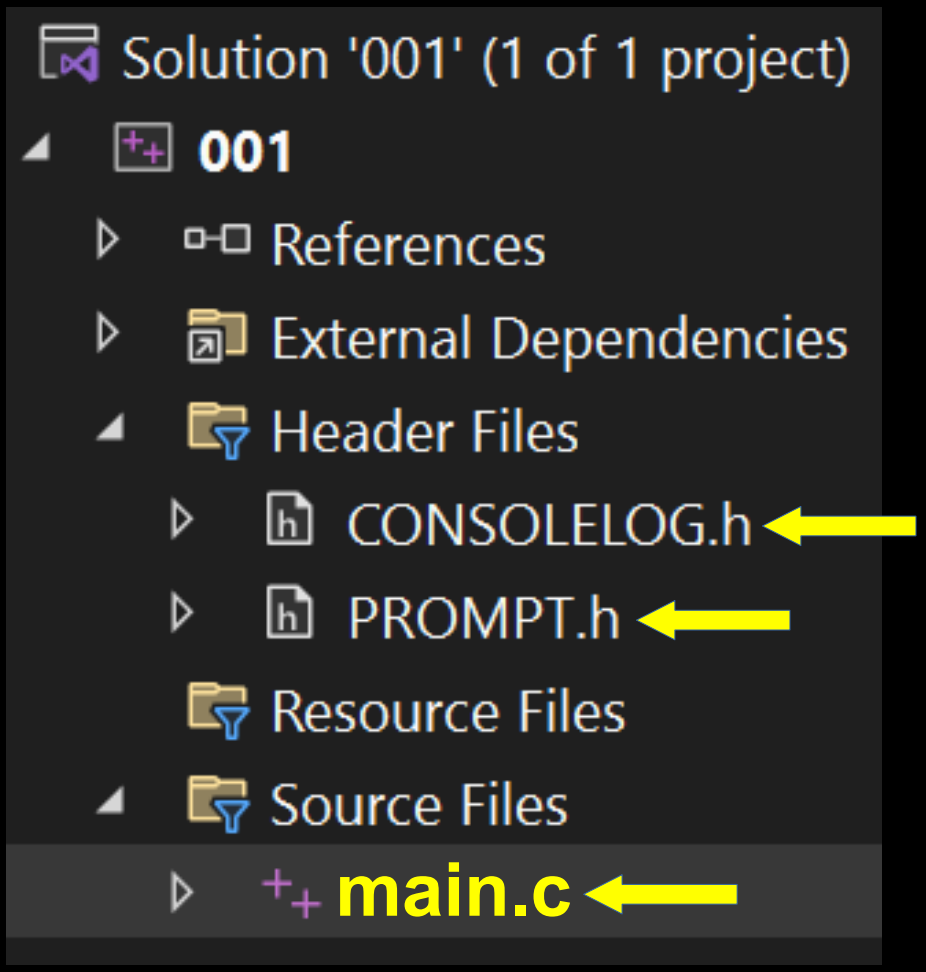
```
// CONSOLELOG.h header file
// CONSOLELOG.h

#ifndef CONSOLELOG
#define CONSOLELOG
#include <stdio.h> // printf

void consoleLog(const char *message)
{
    printf("%s\n", message);
}

#endif
```

```
// Our main.c uses CONSOLELOG.h header file
// main.c

#include "CONSOLELOG.h"
#include <stdio.h> // printf

int main()
{
    consoleLog("Hi Everyone");

    printf("Press Enter to Exit\n");

    // wait for user to press Enter
    getchar();

    return 0;
}
```

```
// PROMPT.h header file
// PROMPT.h

#ifndef PROMPT
#define PROMPT
#include <stdio.h> // scanf_s
#include <string.h> // strlen

void prompt(char* userInput)
{
    // read input from user and prevent buffer overflow
    scanf_s("%s", userInput, 101);

    // wait for user to press Enter
    getchar();
}

#endif
```

```
// main.c uses CONSOLELOG.h and PROMPT.h
// main.c

#include "PROMPT.h"
#include "CONSOLELOG.h"
#include <stdio.h> // printf, scanf

int main()
{
    // max input is 100 chars + null terminator
    char input[101];

    consoleLog("Enter First Name");
    prompt(input);

    printf("Hi %s\n", input);

    consoleLog("Press Enter to Exit\n");

    // wait for user to press Enter
    getchar();

    return 0;
}
```

File Structure of the previous Examples

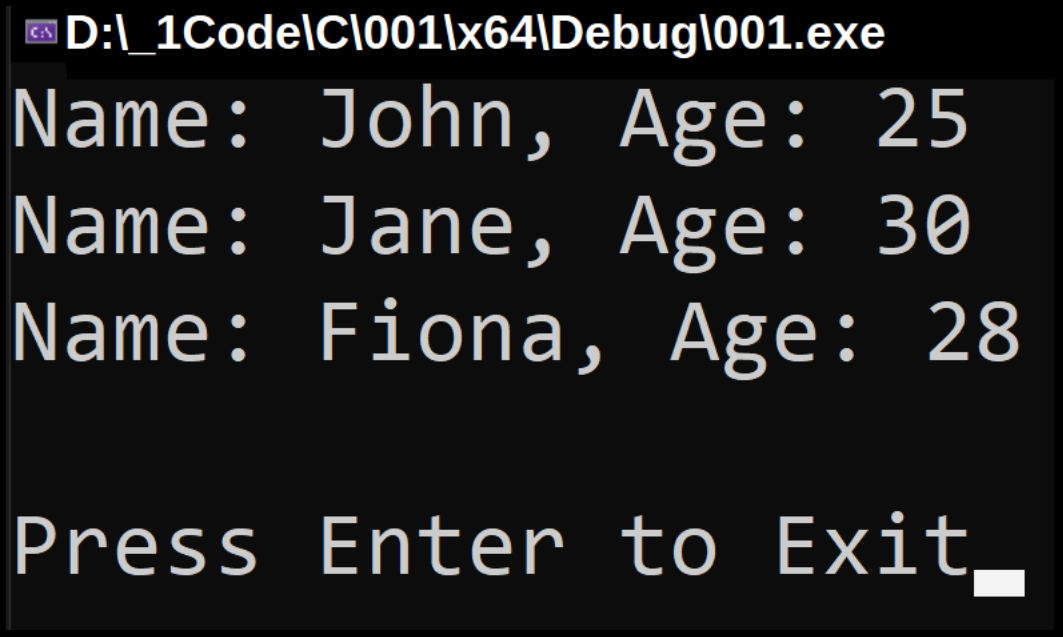


We have 2 Header Files:
CONSOLELOG.h
and
PROMPT.h

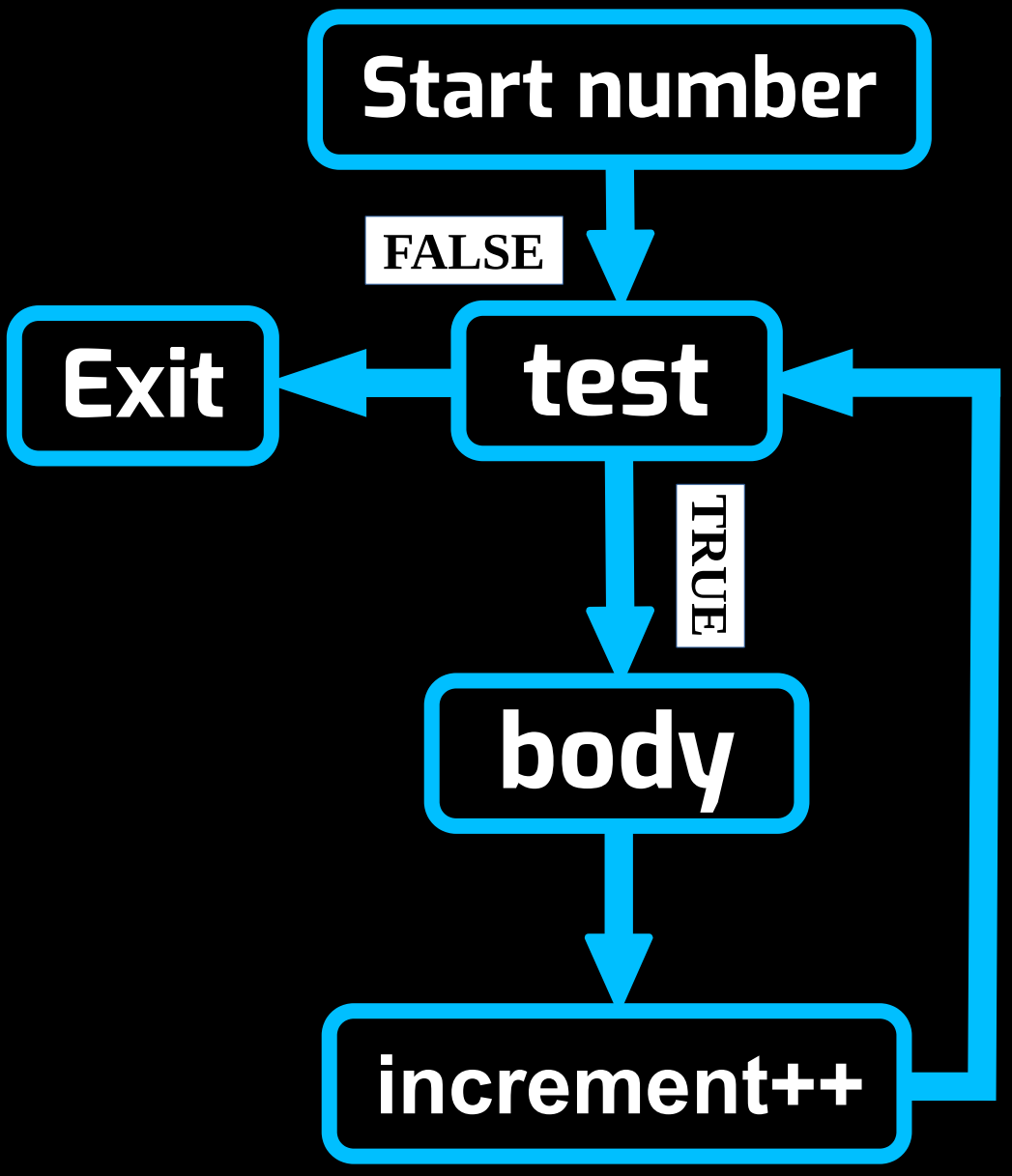
We have 1 main.c file:
main.c uses CONSOLELOG.h and PROMPT.h header files

```c
// Array of Objects
// main.c

#include <stdio.h> // printf

// define a structure to represent a person
struct Person
{
    char name[50];
    int age;
};

int main()
{
    // create an array of Person structs
    struct Person people[] =
    {
        { "John", 25 },
        { "Jane", 30 },
        { "Fiona", 28 }
    };

    // calculate number of elements in array
    int numPeople = sizeof(people) /
sizeof(people[0]);
```

```
    // iterate over each person in the array
    for (int i = 0; i < numPeople; i++)
    {
        printf("Name: %s, Age: %d\n",
people[i].name, people[i].age);
    }

    printf("\nPress Enter to Exit");

    // wait for user to press Enter
    getchar();

    return 0;
}
```



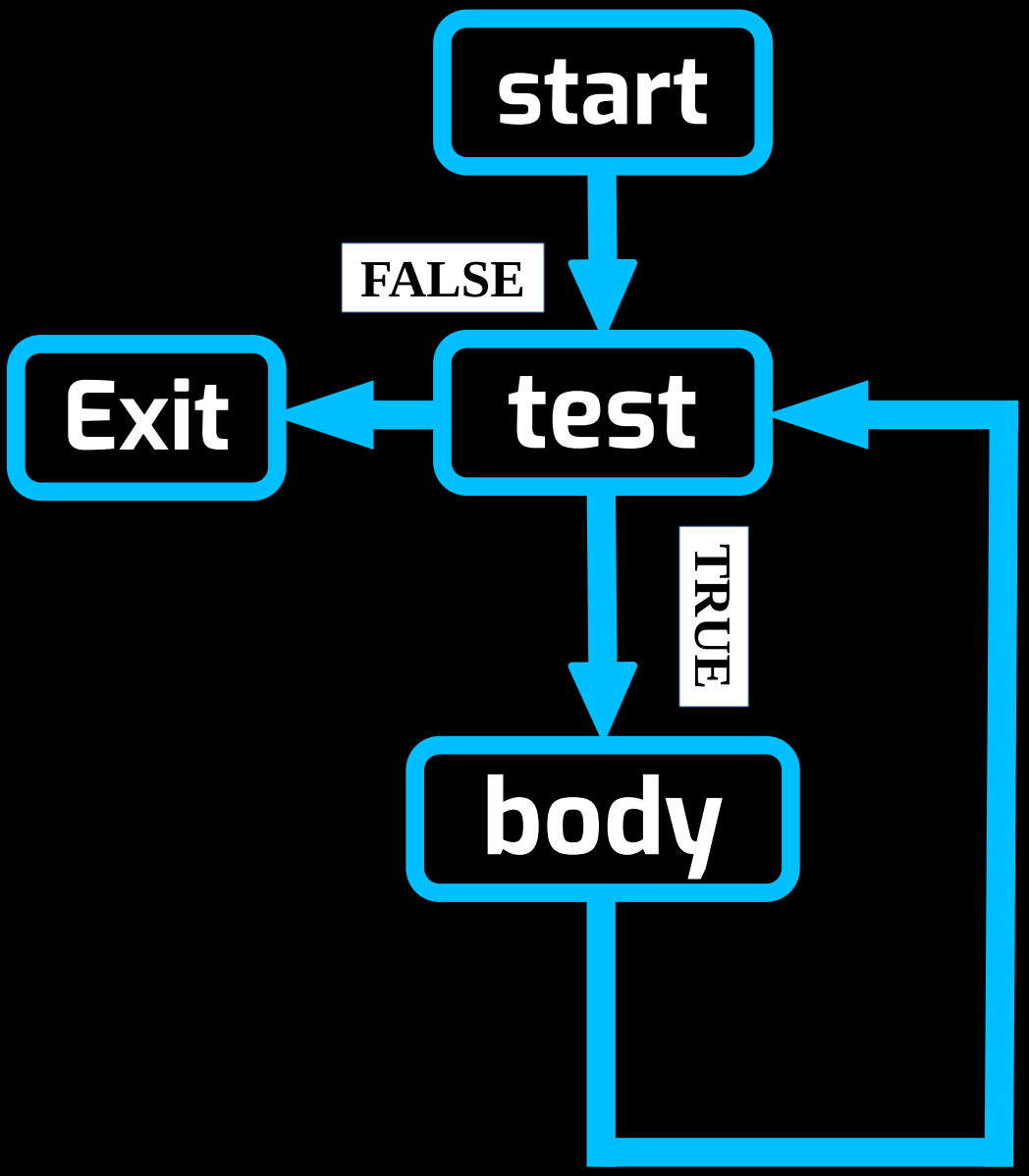
// for loop diagram



```c
// for loop
// main.c

#include <stdio.h> // printf

int main()
{
    for (int i = 1; i <= 100; i++)
    {
        printf("%d ", i);
        printf("\n");
    }

    printf("\nPress Enter to Exit");

    // wait for user to press Enter
    getchar();

    return 0;
}
```

Christopher Topalian

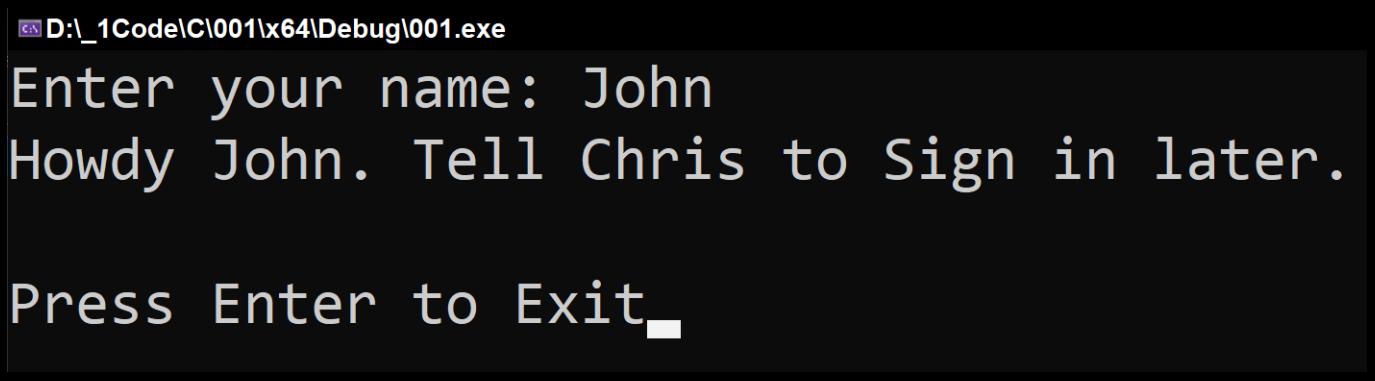
// while loop diagram



```c
// while loop
// main.c

#include <stdio.h> // printf

int main()
{
    for (int i = 1; i <= 100; i++)
    {
        printf("%d ", i);
        printf("\n");
    }

    printf("\nPress Enter to Exit");

    // wait for user to press Enter
    getchar();

    return 0;
}
```

```
// if else - strcmp, without null terminating the string
// main.c

#include <stdio.h>  // printf, scanf_s
#include <string.h> // strcmp

int main()
{
    char name[101]; // buffer to store the name

    printf("Enter your name: ");

    // read input from user and prevent buffer overflow
    scanf_s("%s", name, (unsigned int)sizeof(name));

    // remove newline char left in input buffer
    getchar();

    if (strcmp(name, "Chris") == 0)
    {
        printf("Hi Chris.\nIt is good that you are visiting Earth.\n");
```

```
    }
    else
    {
        printf("Howdy %s. Tell Chris to Sign in later.\n", name);
    }

    printf("\nPress Enter to Exit");

    // wait for user to press Enter
    getchar();

    return 0;
}
```



```c
// if else - strcmp, null terminating the string
// main.c

#include <stdio.h>
#include <string.h>

int main()
{
    char name[101]; // buffer to store the name

    printf("Enter your name: ");

    // read input from user and prevent buffer
overflow
    scanf_s("%100s", name, (unsigned
int)sizeof(name) - 1);

    // explicitly null terminate the string
    name[sizeof(name) - 1] = '\0';

    // remove newline char left in input buffer
    getchar();

    if (strcmp(name, "Chris") == 0)
    {
```

```
        printf("Hi Chris.\nIt is good that you are visiting Earth.\n");
    }
    else
    {
        printf("Howdy %s. Tell Chris to Sign in later.\n", name);
    }

    printf("\nPress Enter to Exit");

    // wait for user to press Enter
    getchar();

    return 0;
}

// This version makes sure that we first null terminate the string before comparison.
```

```
// Open Browser to a URL
// main.c

#include <windows.h>

int main()
{
    ShellExecuteA(NULL, "open",
"https://www.google.com", NULL, NULL,
SW_SHOWNORMAL);

    return 0;
}
```

```
// Open Browser to a URL using char url
// main.c

#include <windows.h>

int main()
{
    // URL to open
    const char* url = "https://www.google.com";

    // open web browser to specified URL
    ShellExecuteA(NULL, "open", url, NULL, NULL, SW_SHOWNORMAL);

    return 0;
}
```

```
// Custom Function - Open Browser to a URL
// main.c

#include <windows.h>

void openURL(const char *url)
{
    ShellExecuteA(NULL, "open", url, NULL,
NULL, SW_SHOWNORMAL);
}

int main()
{
    const char *url = "https://www.google.com";

    openURL(url);

    return 0;
}
```

```
// Create Text File with Data
// main.c

#include <stdio.h>

int main()
{
    // declare FILE pointer
    FILE* outputFile;

    // open file for writing
    if (fopen_s(&outputFile, "ourTextFile.txt",
"w") == 0)
    {
        // write data to file using fprintf_s
        if (fprintf_s(outputFile, "Hi Everyone\n") >
0)
        {
            // close file
            fclose(outputFile);

            printf("Data written successfully.\n");
        }
        else
        {
```

```
            printf("Error writing data to file.\n");
            // close file if an error occurs
            fclose(outputFile);
        }
    }
    else
    {
        printf("Failed to open file.\n");
    }

    return 0;
}
```

```
// Custom Function - Create Text File with Data
// main.c

#include <stdio.h>

void writeToFile(const char* fileName, const
char* content)
{
    // declare FILE pointer
    FILE* outputFile;

    // open file for writing
    if (fopen_s(&outputFile, fileName, "w") == 0)
    {
        // check if file opened successfully
        if (outputFile != NULL)
        {
            // write data to file
            fprintf(outputFile, "%s\n", content);

            // close file
            fclose(outputFile);

            printf("Data written to %s successfully.\
n", fileName);
```

```
        }
        else
        {
            printf("Failed to open file %s.\n",
fileName);
        }
    }
    else
    {
        printf("Failed to open file %s.\n",
fileName);
    }
}

int main()
{
    const char* fileName = "ourTextFile.txt";
    const char* content = "Hi Everyone";

    writeToFile(fileName, content);

    return 0;
}
```

```
// Read a Text File
// main.c

#include <stdio.h>

void displayFileContents(const char* fileName)
{
    // declare FILE pointer
    FILE* inputFile;

    // open file for reading
    if (fopen_s(&inputFile, fileName, "r") == 0)
    {
        // check if file opened successfully
        if (inputFile != NULL)
        {
            // max line length is 255 chars + 1 null
terminator
            char line[256];

            printf("Contents of %s:\n", fileName);

            // read, display file contents line by line
            while (fgets(line, sizeof(line), inputFile) !
= NULL)
```

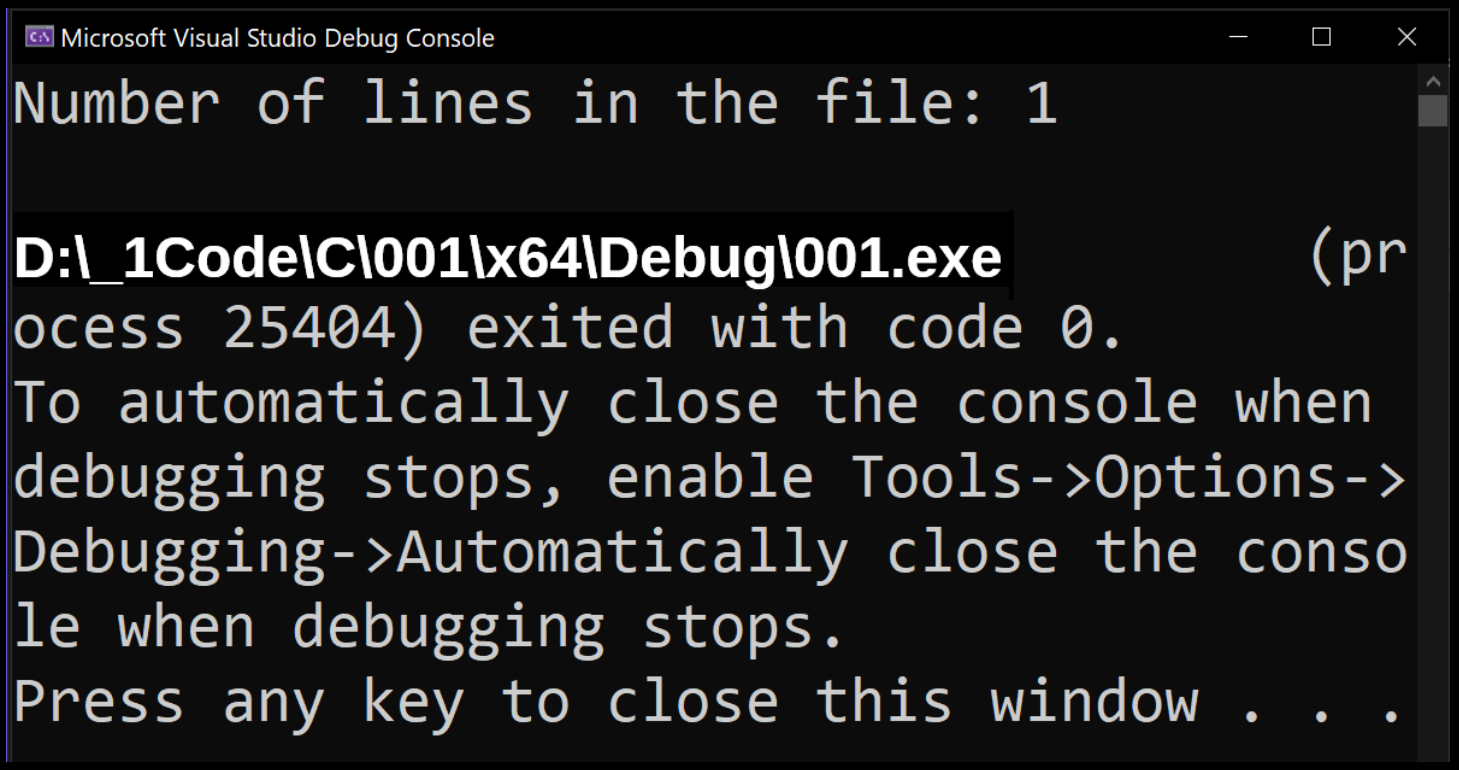
```
            {
                printf("%s", line);
            }

            // close the file
            fclose(inputFile);
        }
        else
        {
            printf("Error opening file: %s\n",
fileName);
        }
    }
    else
    {
        printf("Error opening file: %s\n",
fileName);
    }
}

int main()
{
    const char* fileName = "ourTextFile.txt";

    displayFileContents(fileName);
```

```
    printf("\nPress Enter to Exit");

    // wait for user to press Enter
    getchar();

    return 0;
}
```

```c
// Count Number of Lines in a Text File
// main.c

#include <stdio.h>

int main()
{
    // declare FILE pointer
    FILE* inputFile;

    // open text file for reading
    if (fopen_s(&inputFile, "ourTextFile.txt", "r")
== 0)
    {
        // check if the file opened successfully
        if (inputFile == NULL)
        {
            fprintf(stderr, "File won't open.\n");
            return 1; // return an error code
        }

        // variable to store the count of lines
        int lineCount = 0;
```

```
        // temporary buffer to store each line read
from file
        // max line length is 255 chars + 1 null
terminator
        char line[256];

        // read file line by line and count lines
        while (fgets(line, sizeof(line), inputFile) !=
NULL)
        {
            lineCount++;
        }

        // close the file
        fclose(inputFile);

        // display the total number of lines
        printf("Number of lines in the file: %d\n",
lineCount);
    }
    else
    {
        fprintf(stderr, "File won't open.\n");
        return 1; // return an error code
    }
```

```
    return 0;
}
```

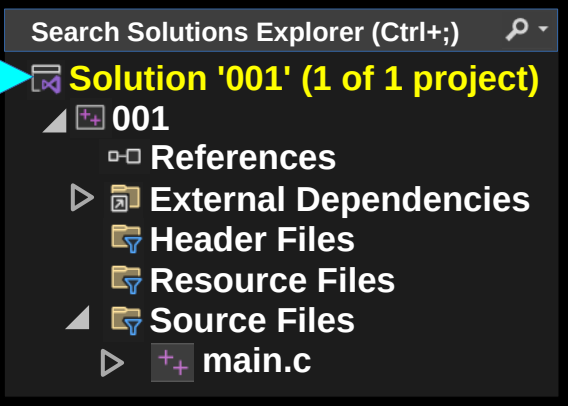

Microsoft Visual Studio Debug Console
Number of lines in the file: 1
D:_1Code\C\001\x64\Debug\001.exe (process 25404) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .

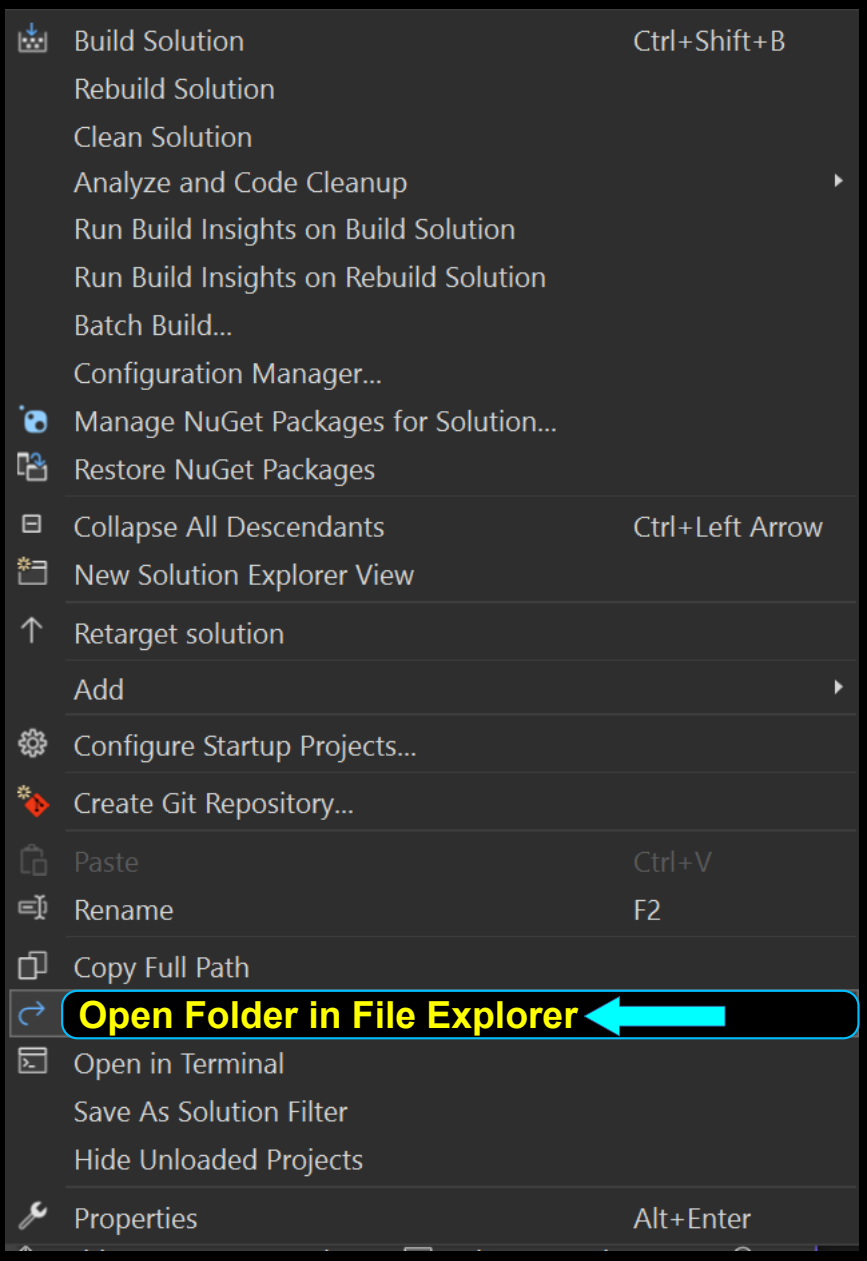
```
// Calculate Hard Drive Memory Statistics

#include <stdio.h>
#include <windows.h>

int main()
{
    ULARGE_INTEGER freeBytesAvailable;
    ULARGE_INTEGER totalBytes;
    ULARGE_INTEGER totalFreeBytes;

    if (GetDiskFreeSpaceEx(NULL,
&freeBytesAvailable, &totalBytes,
&totalFreeBytes))
    {
        printf("Total space: %llu bytes\n",
totalBytes.QuadPart);

        printf("Free space: %llu bytes\n",
totalFreeBytes.QuadPart);

        printf("Available space: %llu bytes\n",
freeBytesAvailable.QuadPart);
    }
    else
    {
```

```
        perror("Error getting disk space
information");
        return 1; // return an error code
    }

    printf("\nPress Enter to Exit");

    // wait for user to press Enter
    getchar();

    return 0;
}
```

// How to Find Our Application .exe File

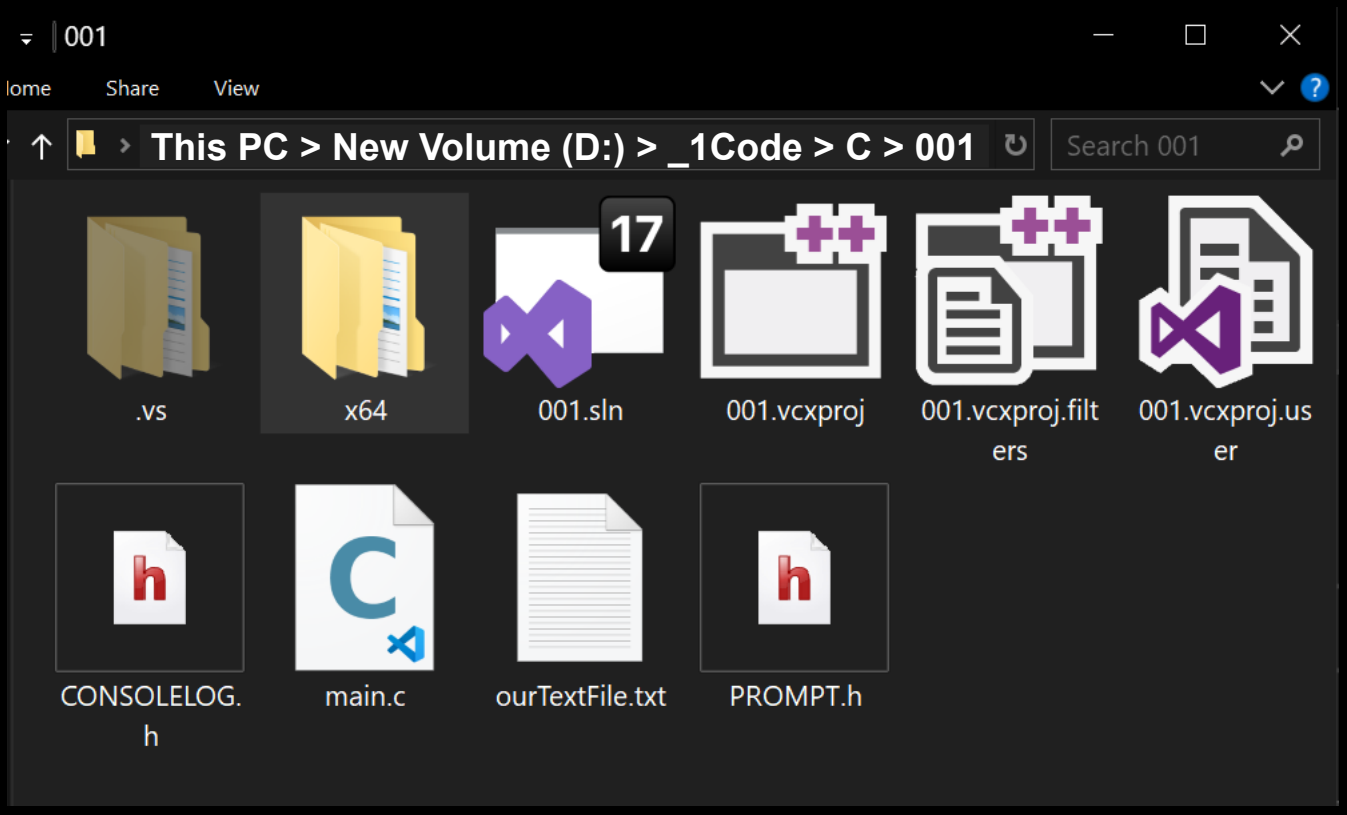
We put mouse arrow on: Solution '001' (1 of 1 project)



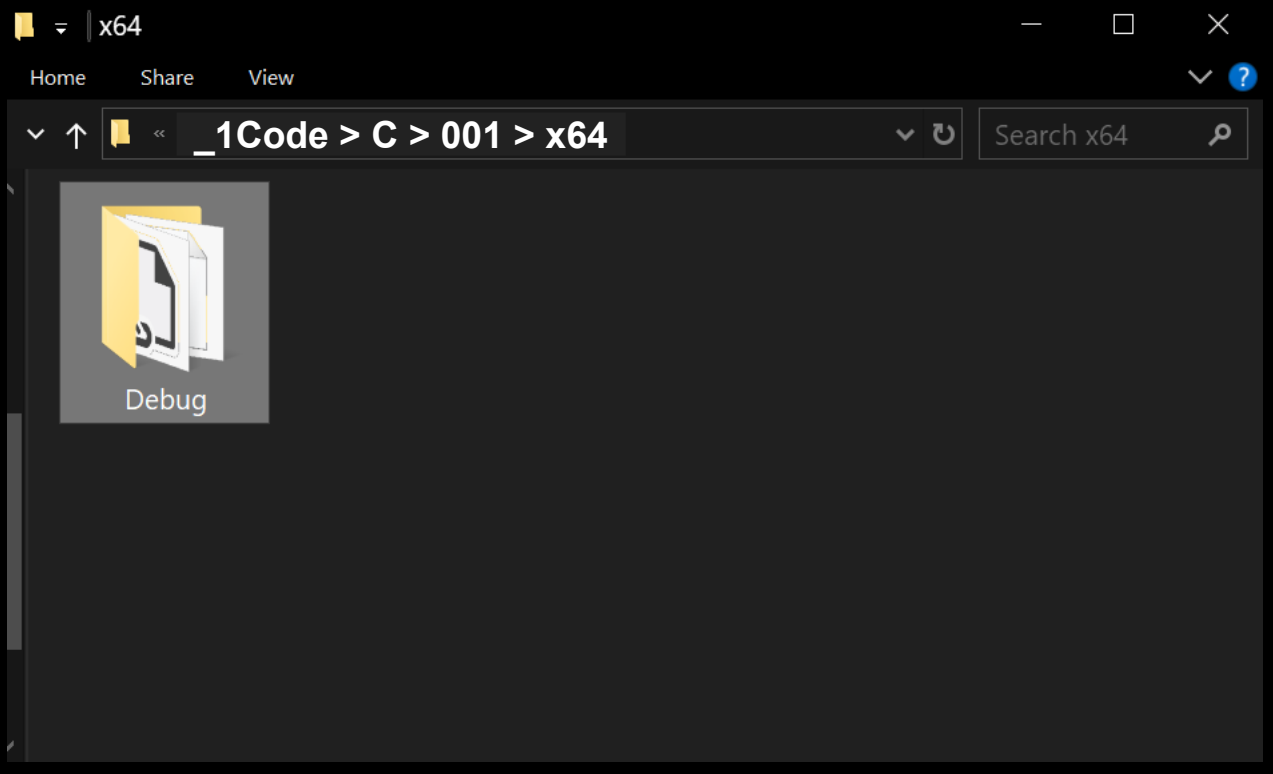


We Choose: Open Folder in File Explorer

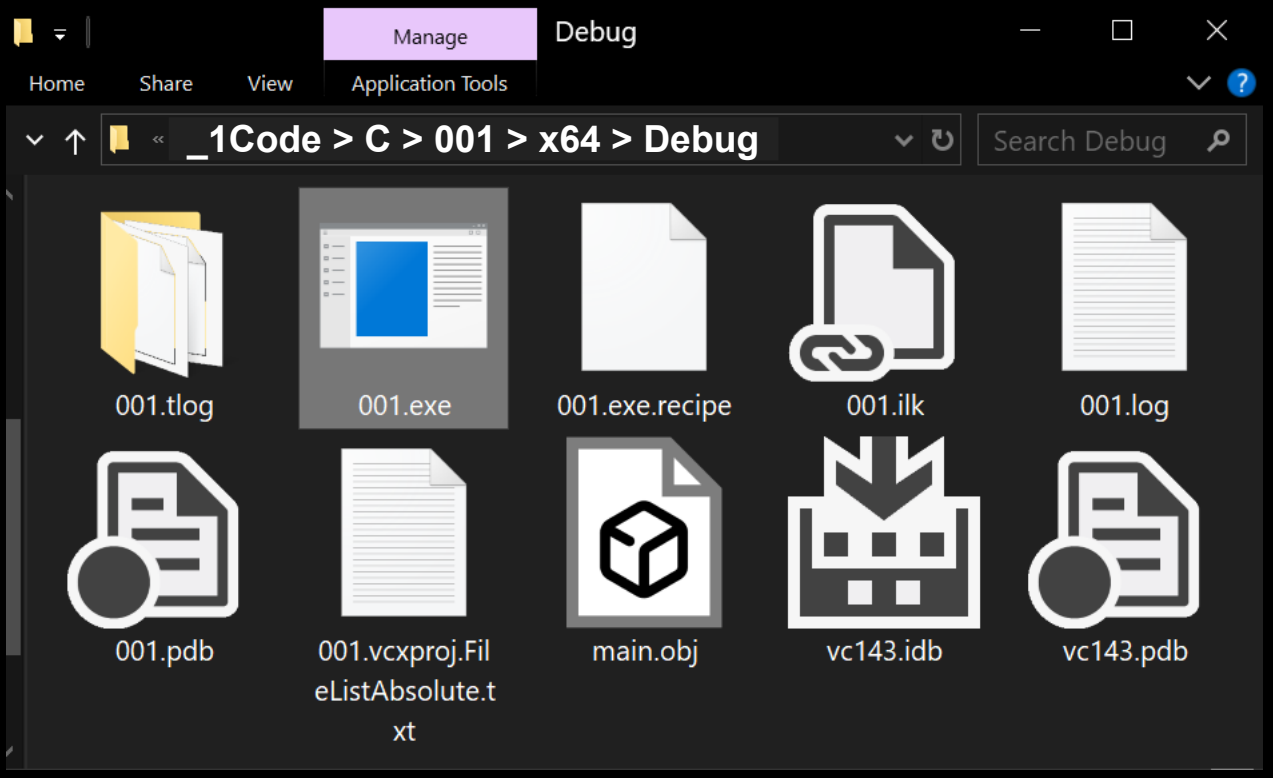
We see that our Project Folder has opened:



We Open: x64 Folder to find the Debug Folder



We Open: Debug Folder to find 001.exe



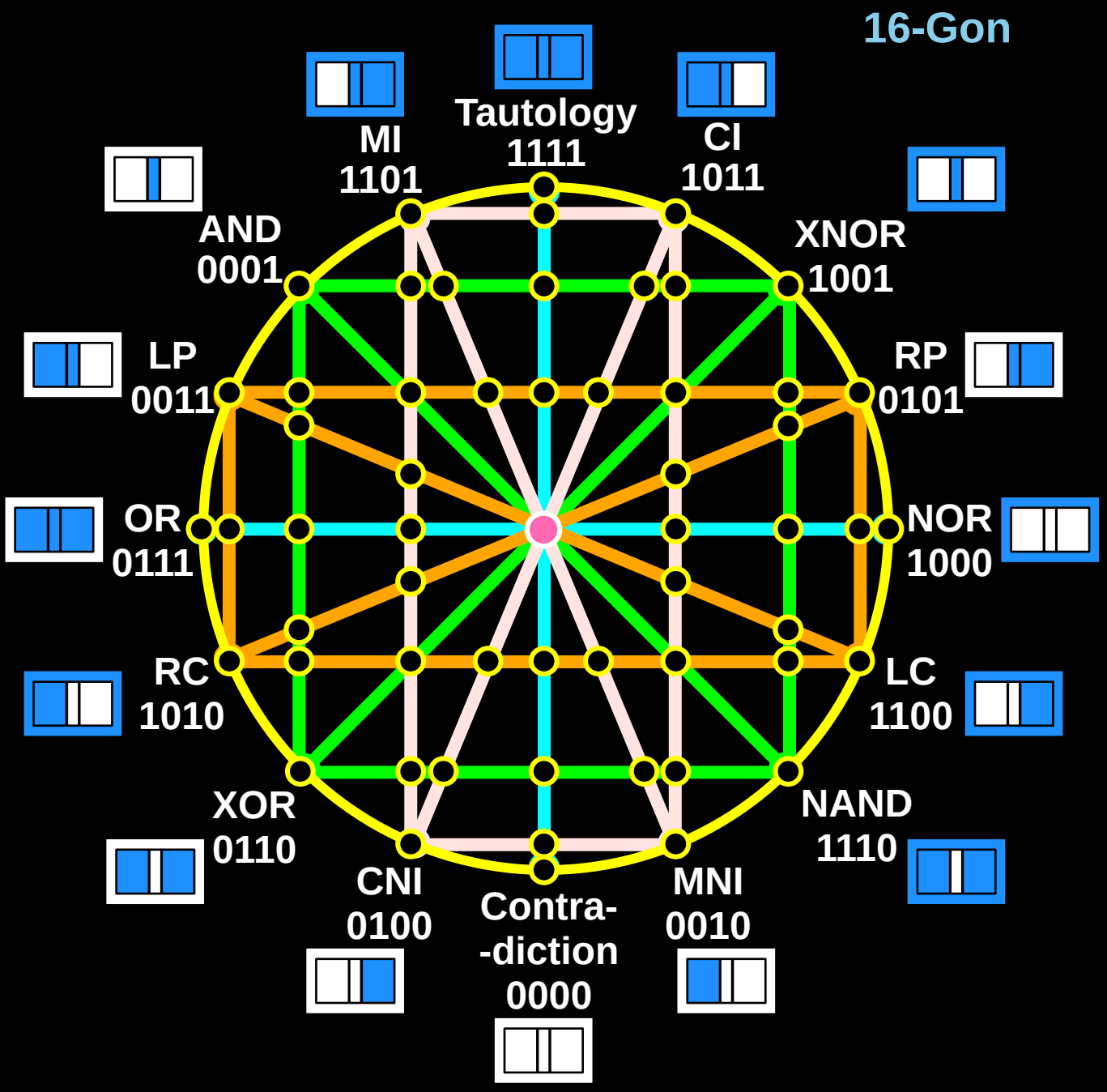
We Double Left Click: 001.exe

Our application should activate.

Happy Programming :-)

Christopher Topalian

True Artificial Intelligence System



For More Tutorials:

CollegeOfScripting.weebly.com

CollegeOfScripting.wordpress.com

GitHub.com/ChristopherTopalian

GitHub.com/ChristopherAndrewTopalian

Youtube.com/ScriptingCollege

Twitter.com/CollegeOfScript

Sites.google.com/view/CollegeOfScripting

Dedicated to God the Father

This book is created by the
College of Scripting Music & Science.

Always remember, that each time you write a script with a pencil and paper, it becomes imprinted so deeply in memory that the material and methods are learned extremely well. When you Type the scripts, the same is true.

The more you type and write out the scripts by keyboard or pencil and paper, the more you will learn programming!

Write & Type EVERY example that you find.
Keep all of your scripts organized.
Every script that you create increases your programming abilities.

SEEING CODE, is one thing,
but WRITING CODE is another.
Write it, Type it, Speak it, See it, Dream it.

www.CollegeOfScripting.weebly.com