CptS 122 – Data Structures

Lab 1: Review of Problem Solving with C and Introduction to Microsoft Visual Studio 2022

Assigned: Week of January 13, 2025
Due: At the end of the lab session

I. Learner Objectives:

At the conclusion of this programming assignment, participants should be able to:
- Analyze a basic set of requirements for a problem
- Compose a small C language program
- Compile a C program using Microsoft Visual Studio 2022
- Create test cases for a program
- Implement pointers in C
- Apply and implement arrays and strings in C
- Apply and implement recursion in C

II. Prerequisites:

Before starting this programming assignment, participants should be able to:
- Access Microsoft Visual Studio 2022 Integrated Development Environment (IDE)
- Apply basic problem solving strategies
- Design and implement small programs in any language

III. Overview & Requirements:

Welcome to CptS 122 – Data Structure's first lab! Labs are constructed to be hands-on. So be ready to get your hands dirty!

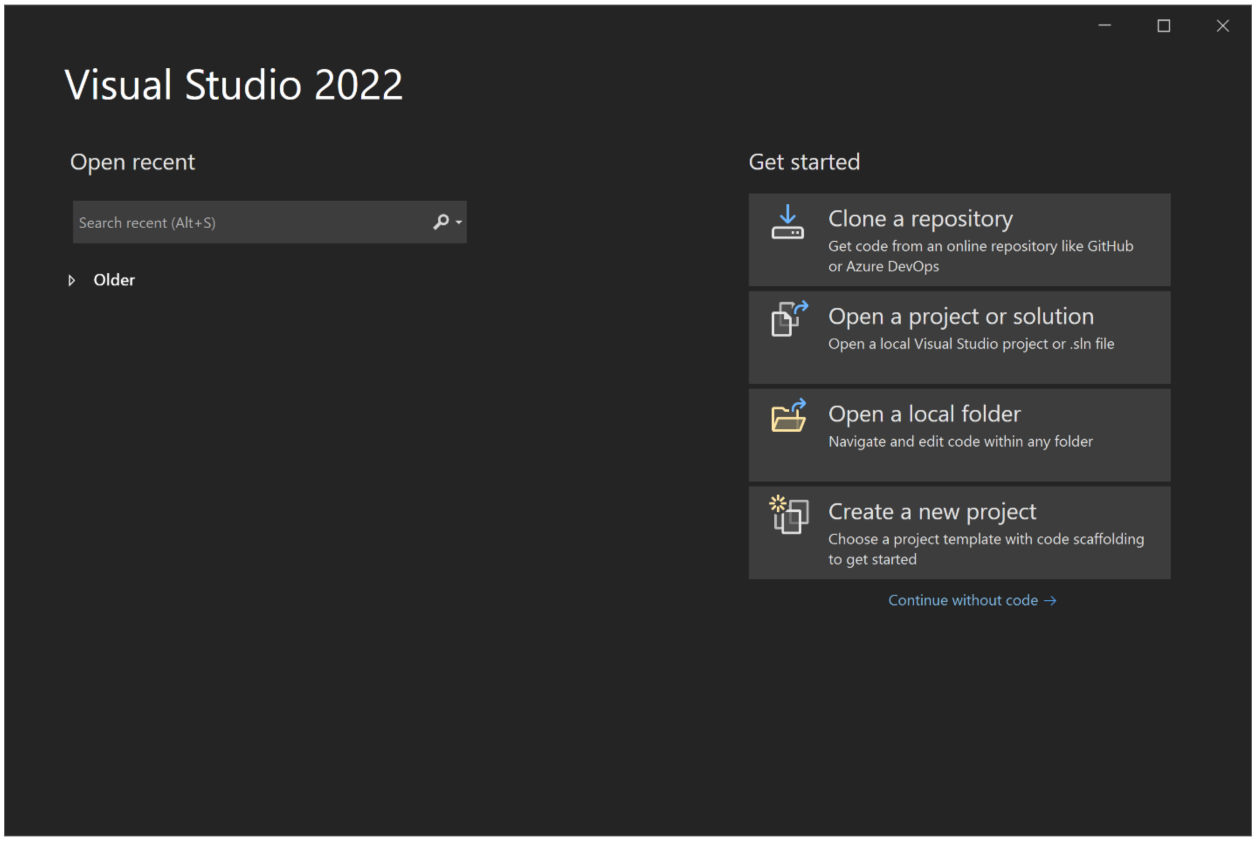
This lab, along with your TA, will help you navigate through Microsoft Visual Studio 2022 Integrated Development Environment (IDE). You will learn how to setup a console application starting from an empty project. You will take advantage of the text editor, compiler, linker, and loader that is built into Visual Studio to construct, execute, and test small C programs that solve basic mathematical problems.

Labs are held in a "closed" environment such that you may ask your TA questions. Please use your TAs knowledge to your advantage. You are required to move at the pace set forth by your TA. You must work in teams assigned by your TA. However, I encourage you to compose your own solution to each problem. Please help other students in need when you are finished with a task. Have a great time! Labs are a vital part to your education in CptS 122 so work diligently.

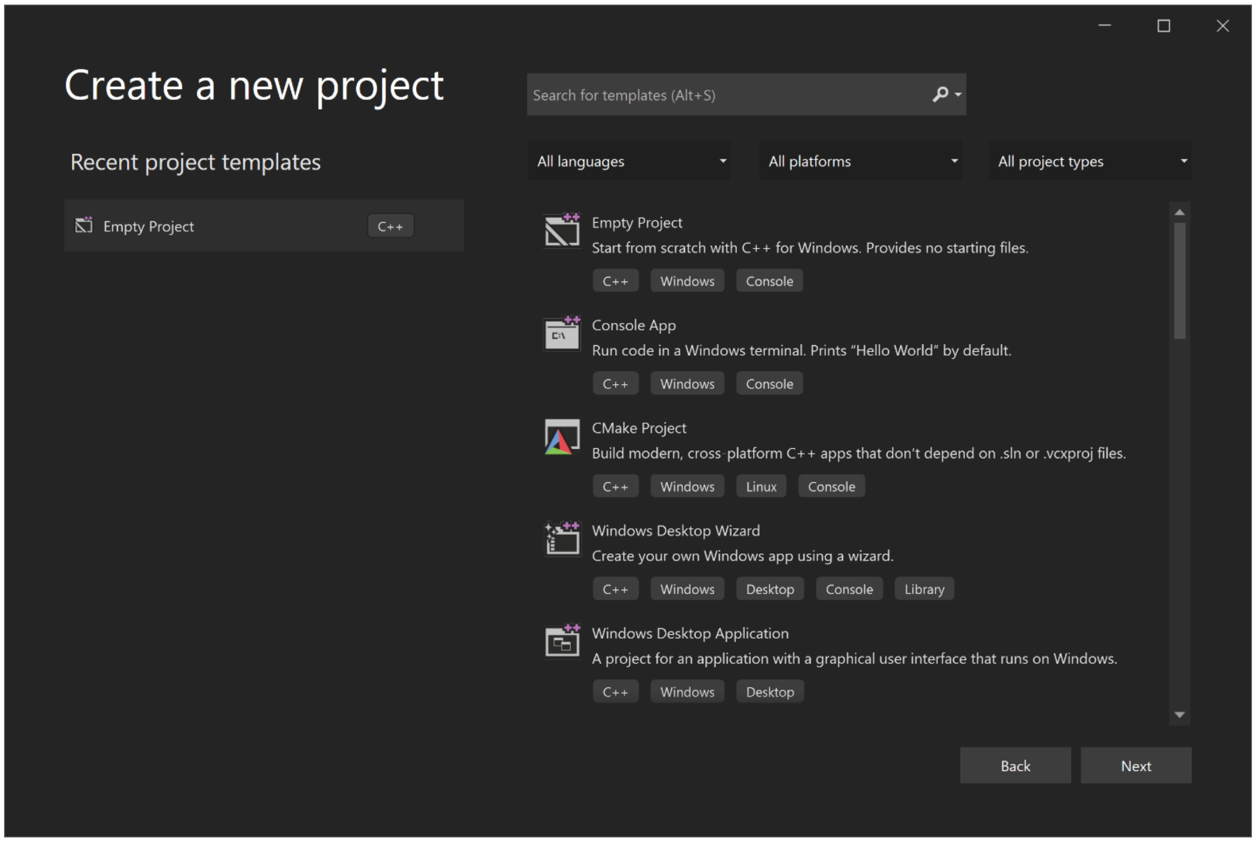
Task 1: Develop a small hello world program utilizing the Visual Studio development environment. All of the applications that we will build in this class will be Console applications. Console applications run in a command window, and have purely textual input and output.

Perform the following steps to build an initially empty console application:

1. Create a folder in an appropriate drive on your computer, for this class, named cpts122. This is where all programs that you create for this class will be saved.
2. From the start menu select Visual Studio 2022 (or equivalently click on the Visual Studio 2022 icon on the desktop if one exists). This will start the visual studio integrated development environment (IDE). The window shown below will load:


Visual Studio 2022
Open recent
Search recent (Alt+S)
Older
Get started
Clone a repository
Get code from an online repository like GitHub or Azure DevOps
Open a project or solution
Open a local Visual Studio project or .sln file
Open a local folder
Navigate and edit code within any folder
Create a new project
Choose a project template with code scaffolding to get started
Continue without code →

Select the “Create a new project” option. This will bring up the window shown below.


Create a new project

Search for templates (Alt+S)

Recent project templates

Empty Project C++

All languages | All platforms | All project types

Empty Project
Start from scratch with C++ for Windows. Provides no starting files.
C++ Windows Console

Console App
Run code in a Windows terminal. Prints "Hello World" by default.
C++ Windows Console

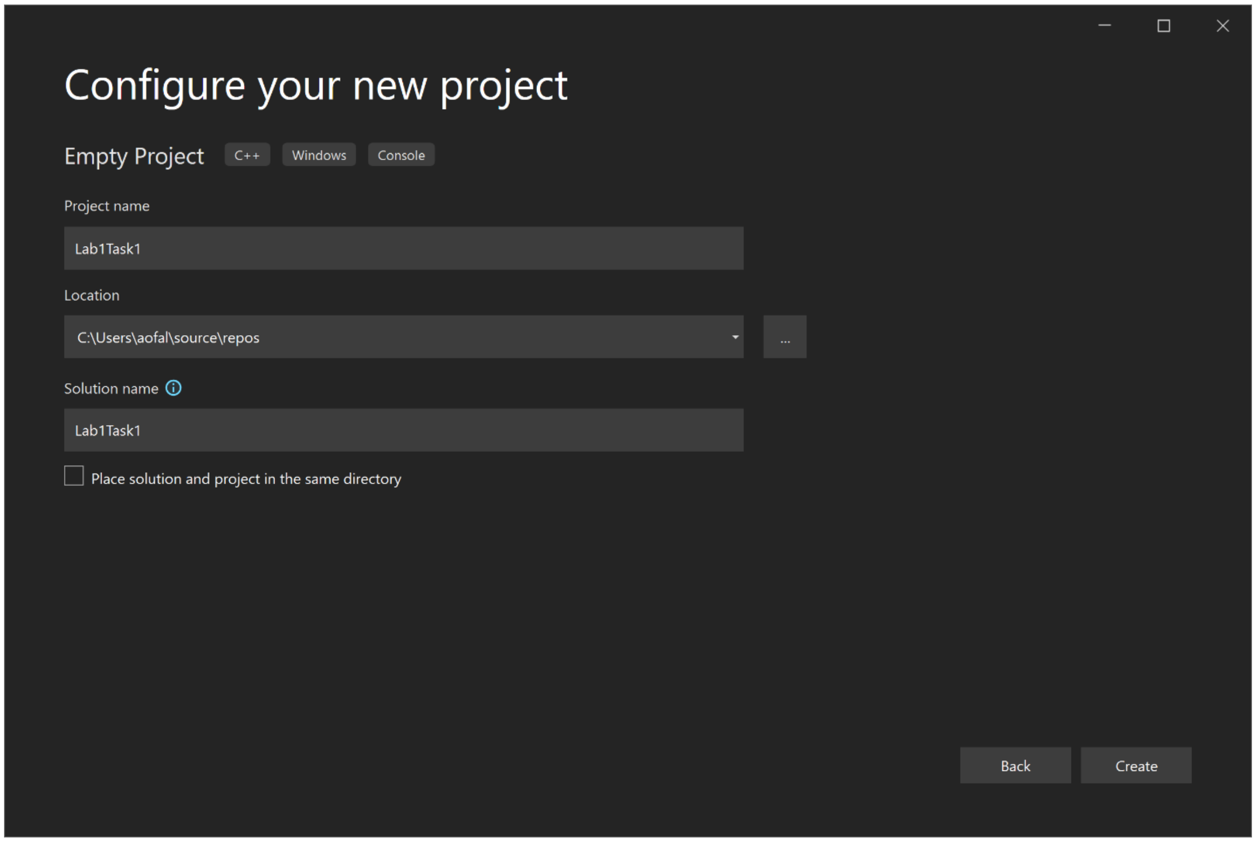
CMake Project
Build modern, cross-platform C++ apps that don't depend on .sln or .vcxproj files.
C++ Windows Linux Console

Windows Desktop Wizard
Create your own Windows app using a wizard.
C++ Windows Desktop Console Library

Windows Desktop Application
A project for an application with a graphical user interface that runs on Windows.
C++ Windows Desktop

In this window, select “Empty Project” (see the shaded entry above). Make sure the project is for “C++”, although we will be using C. If you do not have these project templates, then you may NOT have installed the correct Visual Studio workloads, and will need to rerun the installer (be sure to select C++ desktop tools). Click “Next”.

Next, give the project a name as shown in the image below. A naming scheme will make it easier to find different solutions and projects later. At this time, name this project “Lab1Task1”. You also need to enter the Location; it should be in a location that is easy to find. Click “Create”.


Configure your new project

Empty Project   C++   Windows   Console

Project name

Lab1Task1

Location

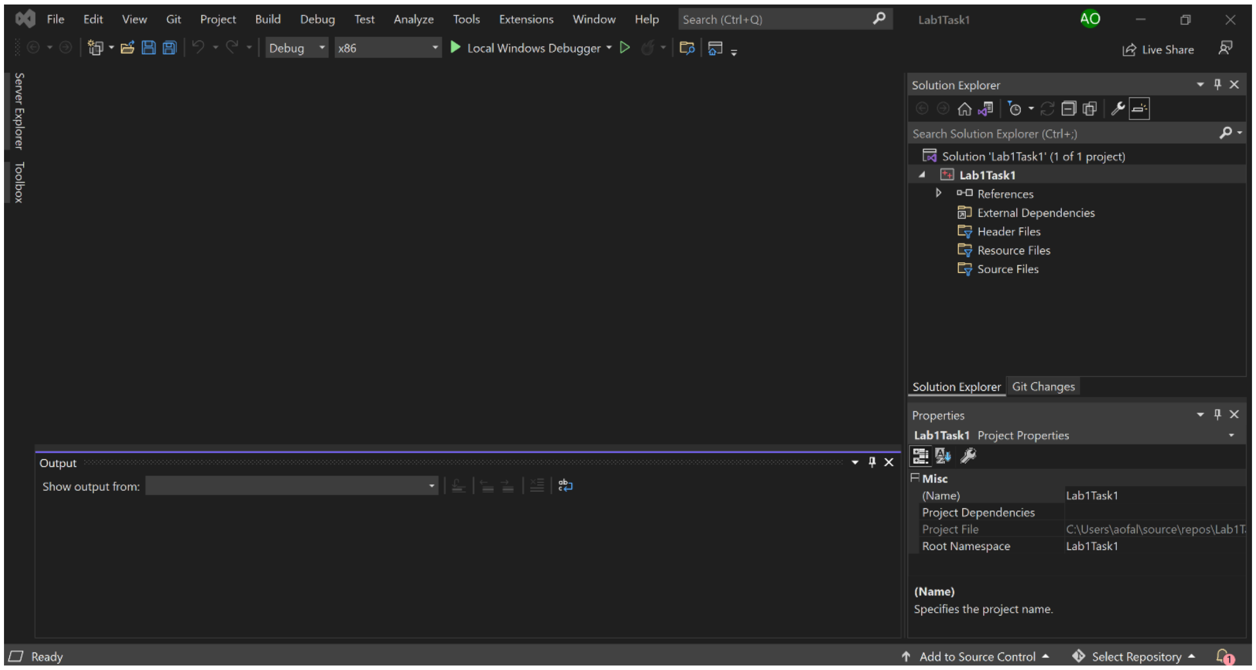
C:\Users\aofal\source\repos

Solution name

Lab1Task1

☐ Place solution and project in the same directory

You will now see the editor load as shown below.



Solution 'Lab1Task1' (1 of 1 project)
- Lab1Task1
  - References
  - External Dependencies
  - Header Files
  - Resource Files
  - Source Files

Lab1Task1 Project Properties

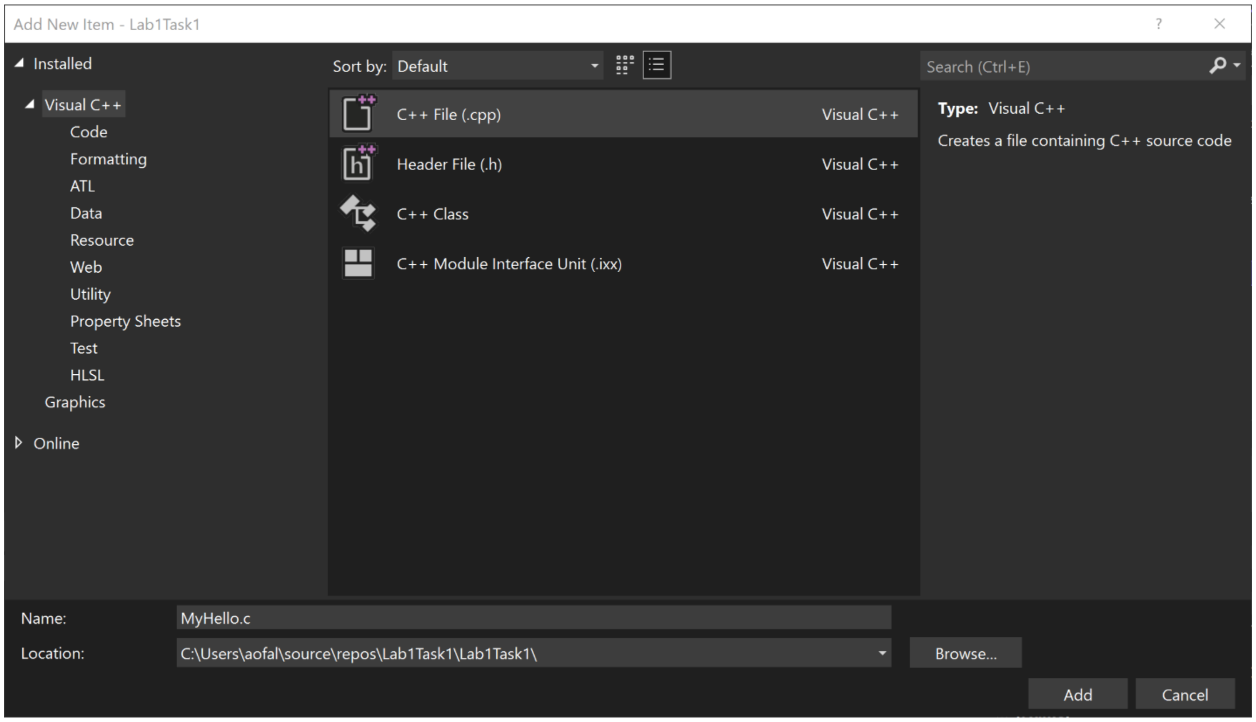
Misc

(Name)	Lab1Task1
Project Dependencies	
Project File	C:\Users\aofal\source\repos\Lab1T
Root Namespace	Lab1Task1

(Name)
Specifies the project name.

The IDE Window has changed a little bit now. There is information on the right hand side of the window in the Solution Explorer. You now need to create a new file for your program. Do this by right clicking on the "Source Files" folder in the "Solution

Explorer", in the Visual Studio IDE, and selecting "Add" --> "New Item". Give your file a name in the filename text field as shown in the image below. Name the file MyHello.c. Make sure that you give your file the .c extension indicating that you would like the C compiler to be invoked once you compile your program. If you do not use the .c extension, the source file will default to a .cpp extension. This will cause the C++ compiler to be invoked. Next click "Add".



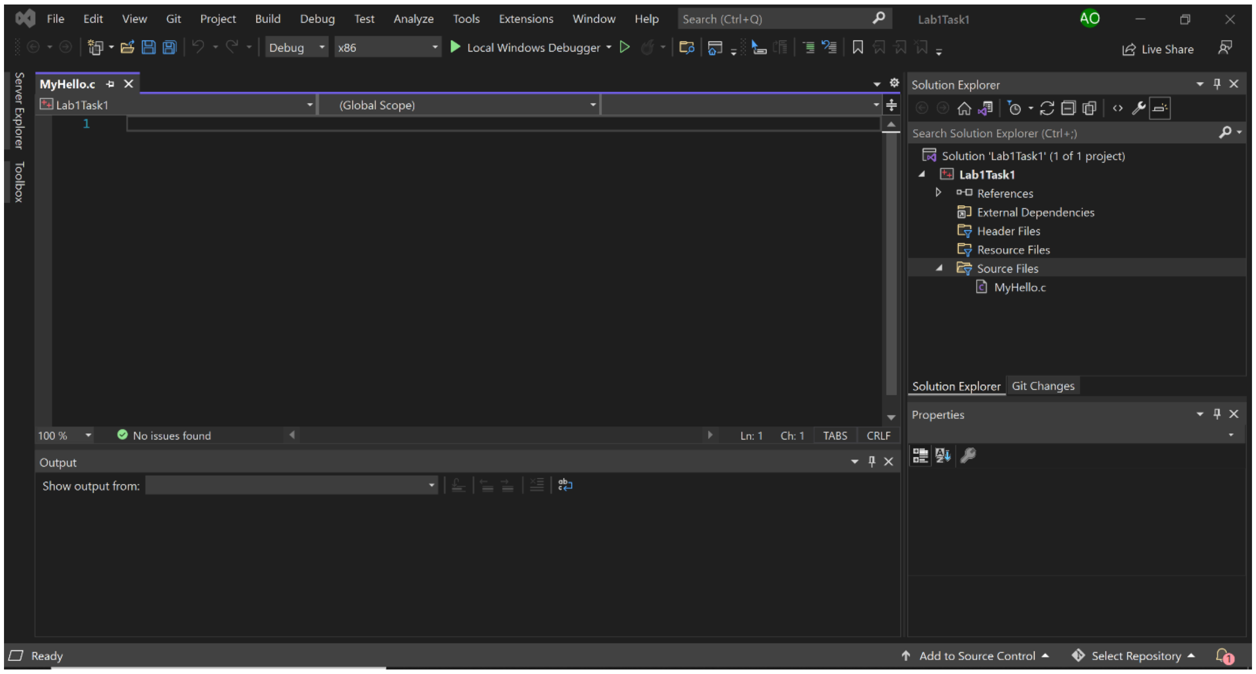
Type: Visual C++

Creates a file containing C++ source code

Name: MyHello.c

Location: C:\Users\aofal\source\repos\Lab1Task1\Lab1Task1\

When you select "Add", you will see a file edit window come up inside the IDE with the name MyHello.c. You can enter your Hello World program here.



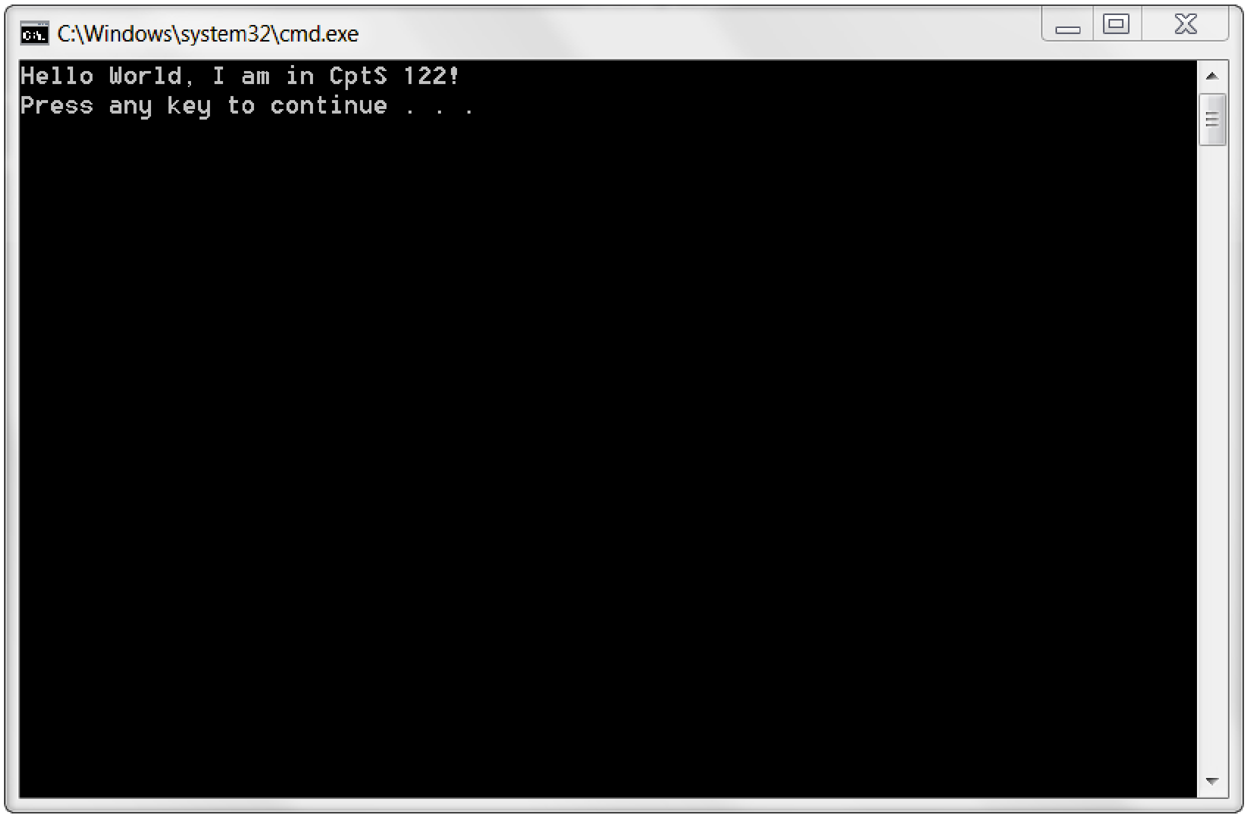
Make sure that for every program that you write, you have the following comment header block at the beginning of your source code file:

```
/*******************************************************************************************
* Programmer: Your Name
* Class: CptS 122, Spring 2025; Lab Section X
* Programming Assignment: Lab1Task1
* Date: January 13, 2025
* Description: This program prints out a simple "hello world" message.
*******************************************************************************************/
```

For this first program you will only need one executable C statement to output a message. Your program should output "Hello World, I am in CptS 122!" with a printf ( ) statement.

Once you have entered the code for your simple hello world program you will build it from the Solution Explorer. Place the mouse pointer over Lab1Task1 in the solution explorer. Right click and choose “Build”. If you correctly entered the program source code you should see a "Build succeeded" message in the Build Window. If you had an error you will have a failed message in the Build window and a fatal error in the Task List Window. Correct your mistakes and try again.

To run your program when it has successfully built, hit Ctrl-F5 or run it from the “Debug” tab in the IDE.



```
Hello World, I am in CptS 122!
Press any key to continue . . .
```

When you have successfully ran your hello world program, show your TA. Once you have completed this you may close this workspace under the file menu. Close this solution and create a new one for the next task. You should be able to move on to the rest of tasks when your TA instructs you to do so.

Before you start the next task, if you have not done so already, please be sure to team up with 2 or 3 other students!!! During all labs, I would like for you and your team to use the whiteboards as much as possible to illustrate ideas and algorithms. You should not start implementing code until you have flushed out most of your ideas as a team! Please be aware that most teams will have a diverse set of skills and I would like for you to encourage and support each other!

Task 2: Design, implement, compile, and test C solutions to the following problems. Once you have completed a problem, demonstrate your solution to your TA.

a. Write a C implementation for the function: `char *my_strcat (char *destination, const char *source)` - This function appends a copy of the string pointed to by source (including the null character) to the end of the string pointed to by destination. The append overwrites the null character at the end of destination. The string pointed to by destination is returned.

b. Write a C function that recursively reverses a string. Recall, a recursive function is a function that calls itself. These functions have at least one base or simple case and at least one recursive step. The base case(s) have known solutions. As the function is called, the problem is broken down into simpler parts that are closer to the base case.

c. Write a C function called `myStrTok()` that behaves in the following manner (taken from http://en.cppreference.com/w/c/string/byte/strtok). Note: all references to `strtok()` should be replaced by `myStrTok()` . You will need to declare a *static* pointer inside of `myStrTok()` to track tokens through successive calls!!! For example:

```
static char *pToken; // static will allow for the
                     // pointer contents/direct value to retain
                     // changes through successive calls
```

Finds the next token in a null-terminated byte string pointed to by `str`. The separator characters are identified by null-terminated byte string pointed to by `delim`. This function is designed to be called multiples times to obtain successive tokens from the same string.

- If `str != NULL`, the call is treated as the first call to `strtok` for this particular string. The function searches for the first character which is *not* contained in `delim`.

- If no such character was found, there are no tokens in `str` at all, and the function returns a null pointer.
- If such character was found, it is the *beginning of the token*. The function then searches from that point on for the first character that *is* contained in `delim`.
- If no such character was found, `str` has only one token, and future calls to `strtok` will return a null pointer
- If such character was found, it is *replaced* by the null character `'\0'` and the pointer to the following character is stored in a *static* location for subsequent invocations
- The function then returns the pointer to the beginning of the token
- If `str == NULL`, the call is treated as a subsequent calls to `strtok`: the function continues from where it left in previous invocation. The behavior is the same as if the previously stored pointer is passed as `str`. The behavior is undefined if either `str` or `delim` is not a pointer to a null-terminated byte string.

d. Write a C function that integrates or merges two unsorted strings into sorted order. You will need to provide multiple solutions.
   i. Solution 1: Merge items into a third, fixed-sized array.
   ii. Solution 2: Merge items into a third, dynamically allocated array, which grows as each item is inserted. Consider using `realloc()`. You will find more information about `realloc()` at: http://en.cppreference.com/w/c/memory/realloc. We are NOT using a linked list to solve this problem!!!
   iii. Solution 3: Merge items without the use of a third array.

IV. Submitting Labs:

- You are not required to submit your lab solutions.

V. Grading Guidelines:

- This lab is worth 10 points. Your lab grade is assigned based on completeness and effort. To receive full credit for the lab you must show up on time, work with a team, continue to work on the problems until the TA has dismissed you, and complete at least 2/3 of the problems. If you do not work with a team, then you will not earn any points for the lab.