User Manual

Table of Contents

Introduction

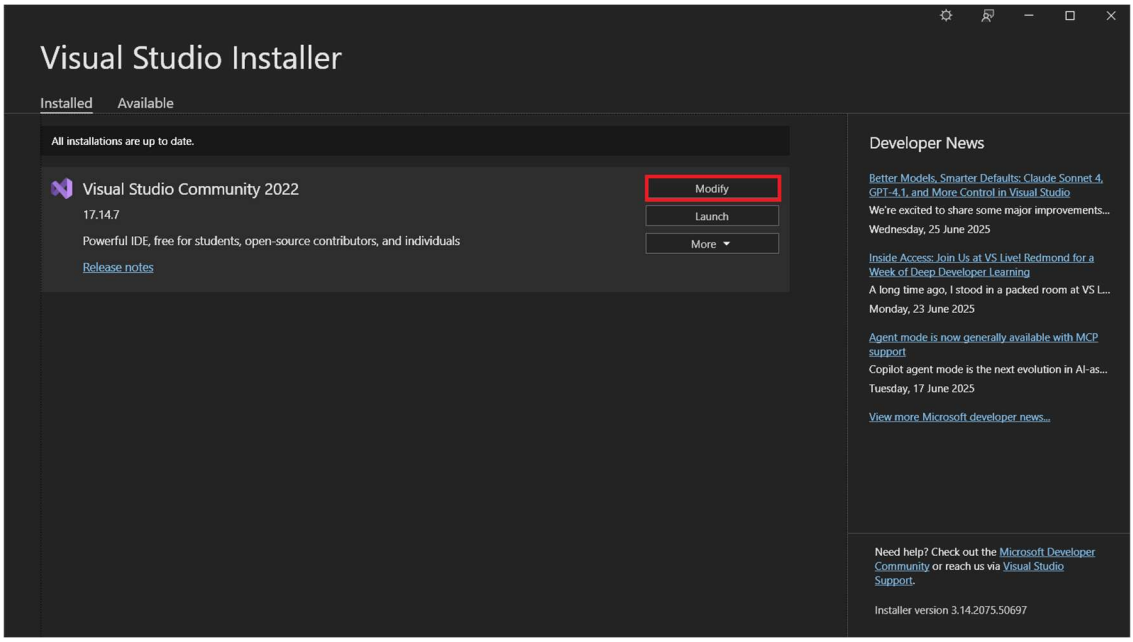
After getting all the necessary files and library, this guide will show you how to use the app from installing the required **C++ components** in **Visual Studio** to performing basic mathematical operations.

Installing C++ Components in Visual Studio

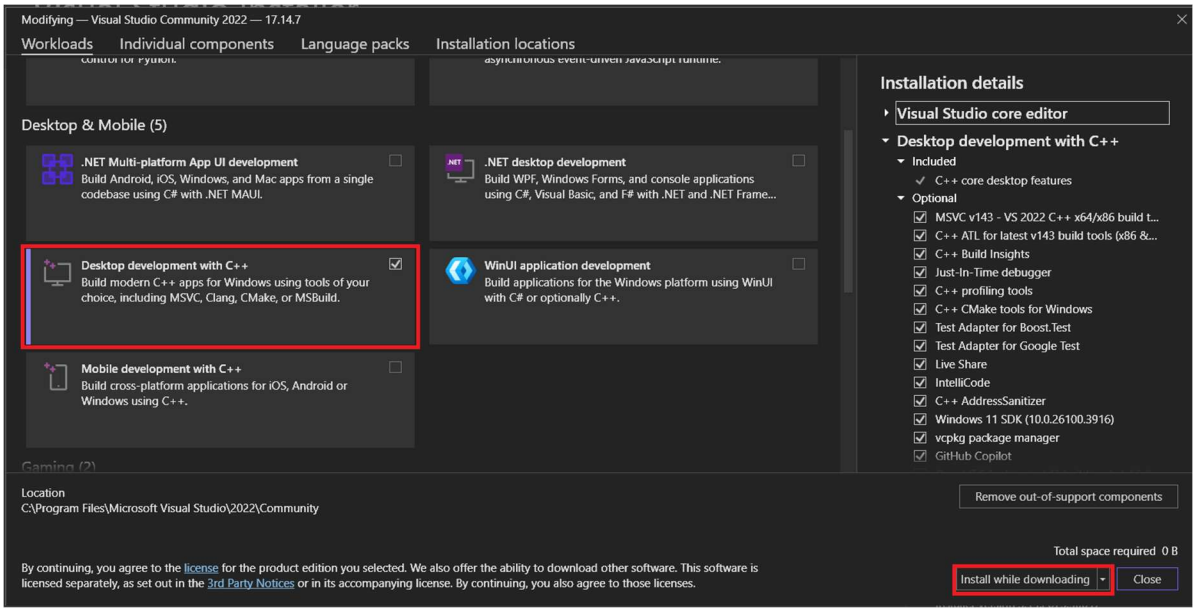
The software of choice here is **Visual Studio**.

Steps:

1. Open **Visual Studio installer**
2. Click **Modify**



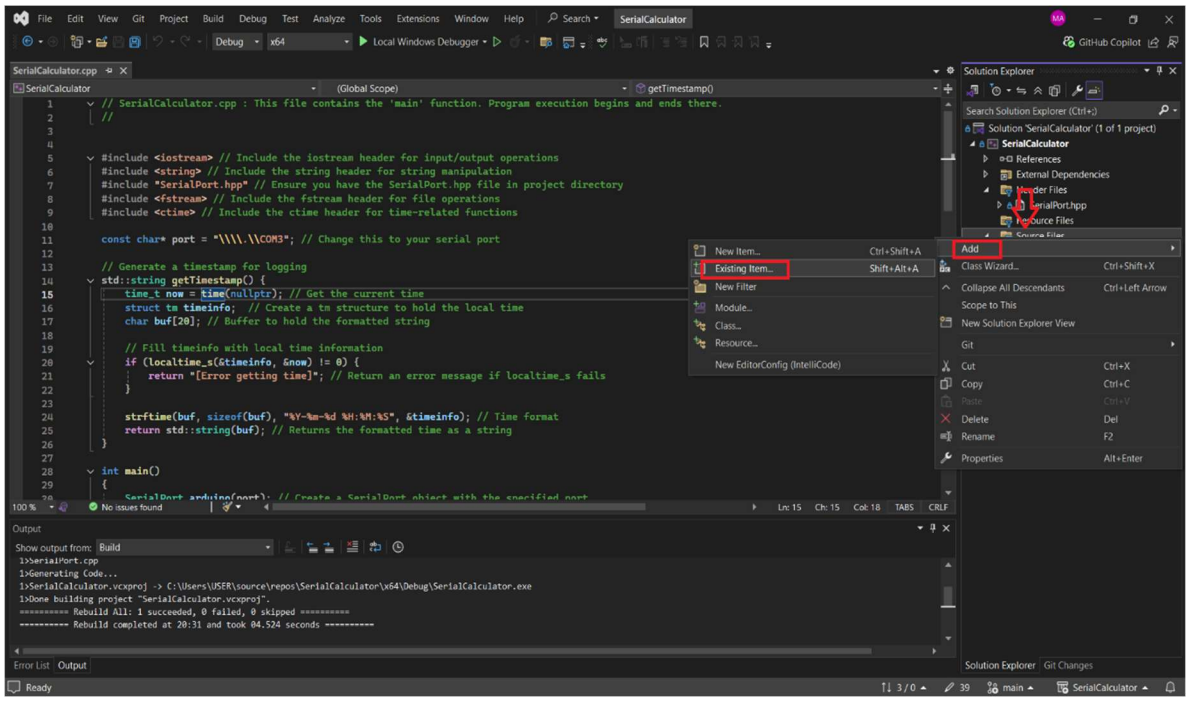
   - Click **Desktop development with C++** as this will install the necessary components and click **Install while downloading**



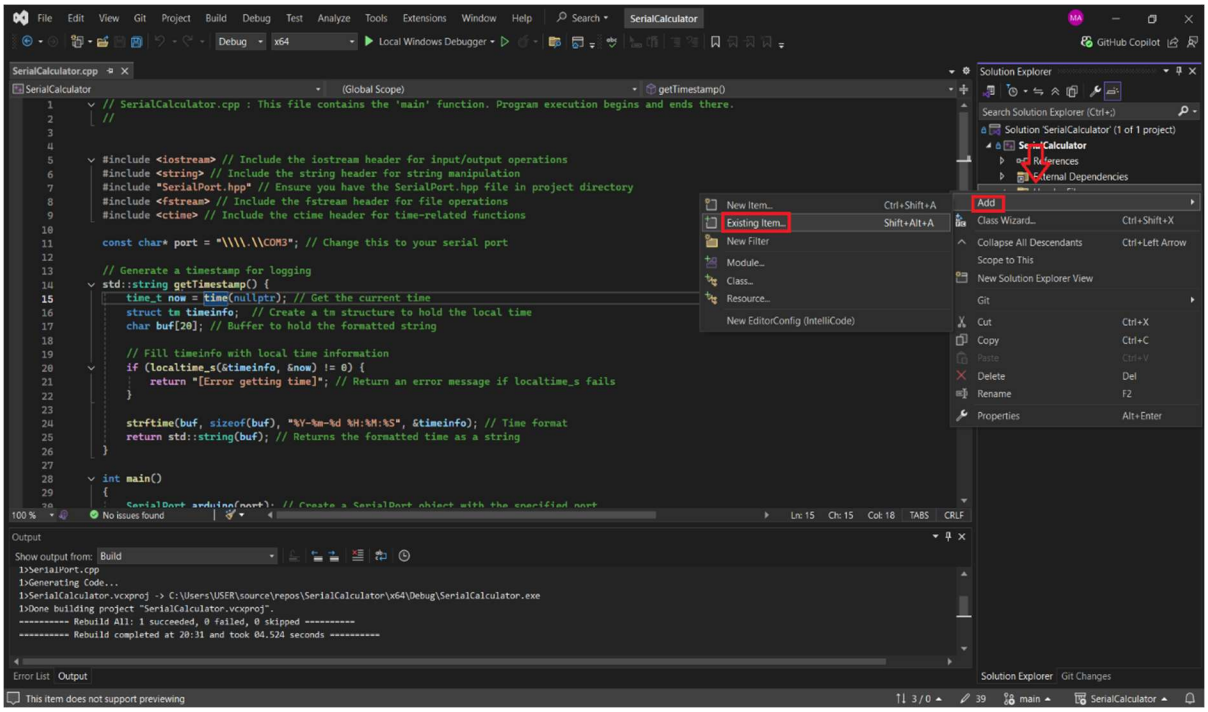
Importing SerialPort Library (Console or Visual Studio)

This can be done in two ways:

1. Import in **Visual Studio**
    - Importing **SerialPort.cpp**



    - Importing **header file: SerialPort.hpp**

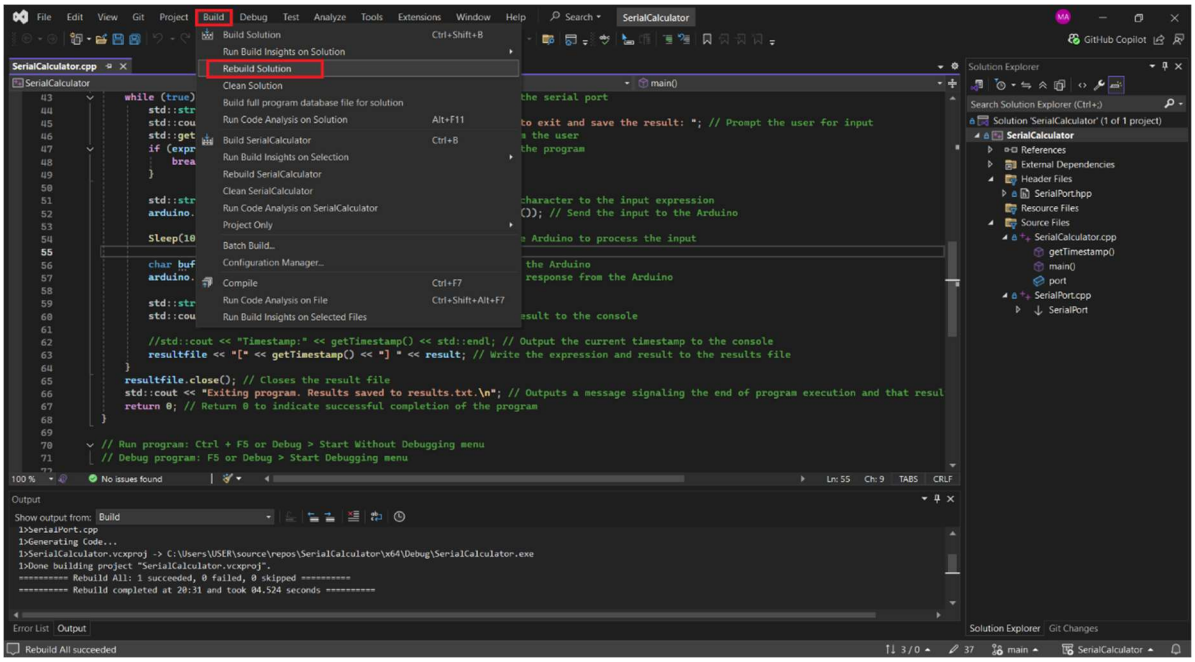


2. Extracting the **SerialPort.zip**, copy and paste the ***SerialPort.cpp*** and ***SerialPort.hpp*** directly into the **project folder**

Building the Calculator

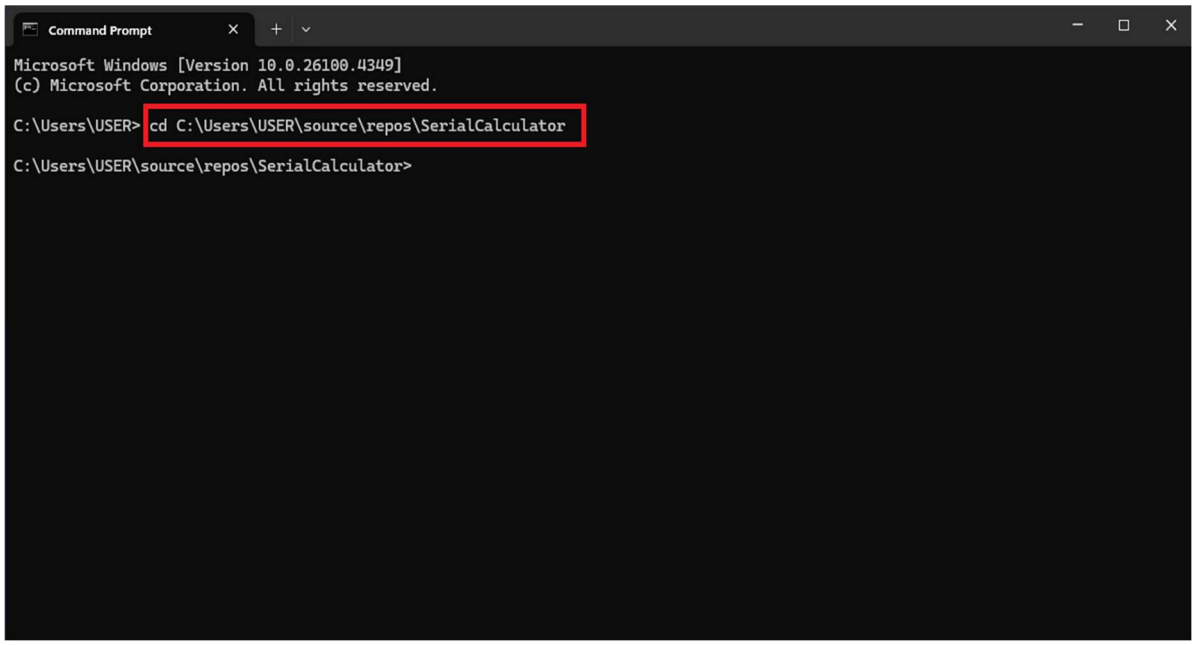
Two methods can be used to do this, either through Visual Studio or the Console App:

1. Visual Studio

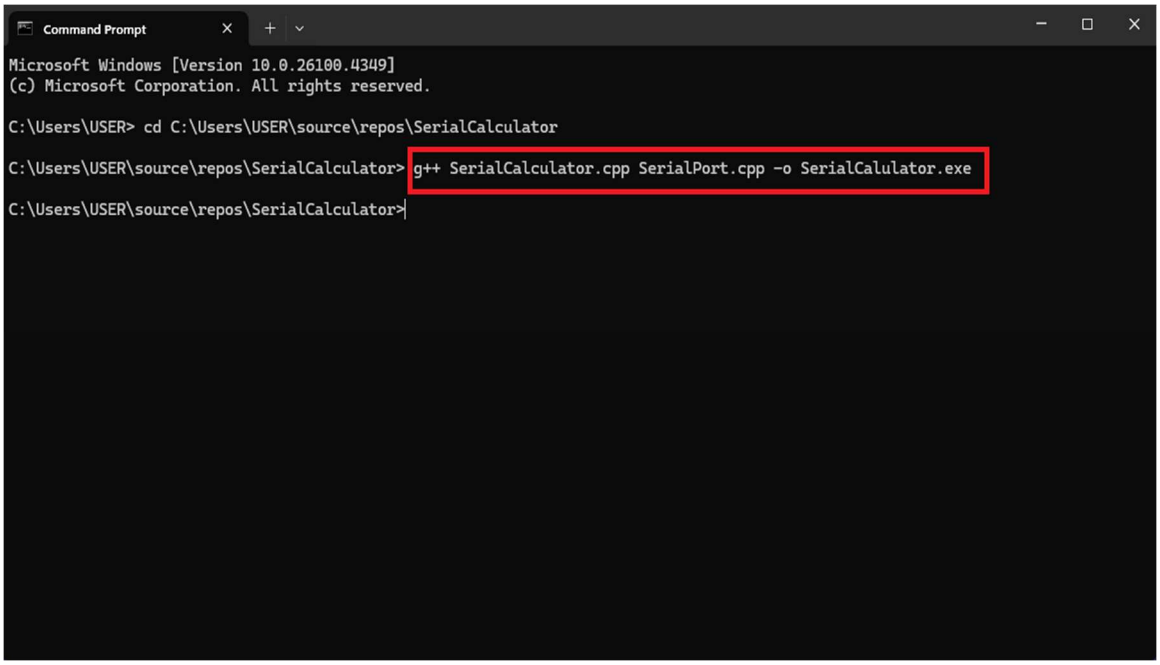


2. Console App

- Change **directory** to the **project directory**

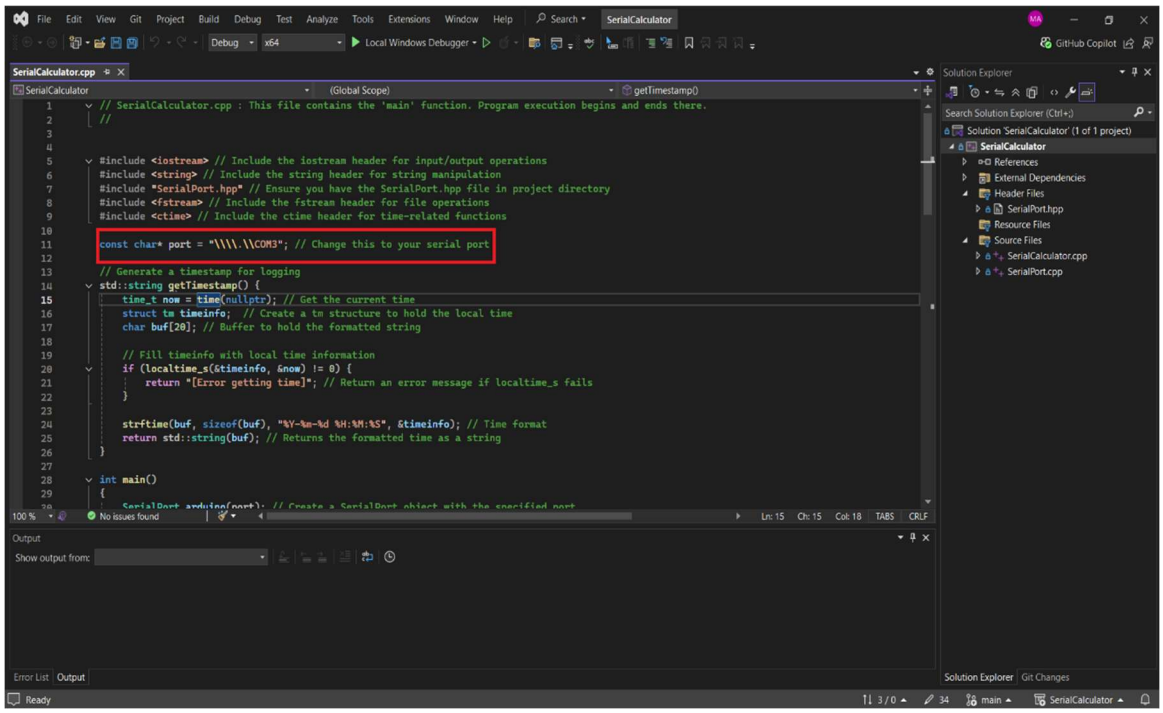


- Creating **executable(.exe)** file

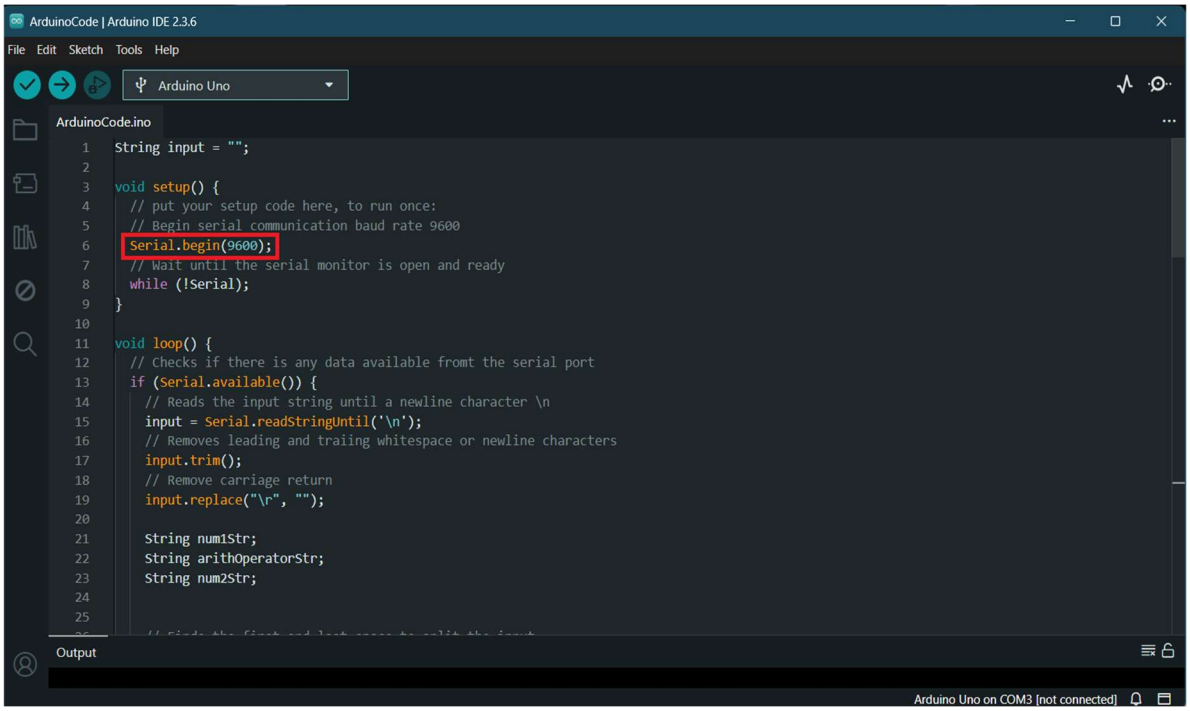


Setting Up the Connection — COM port, baud rate

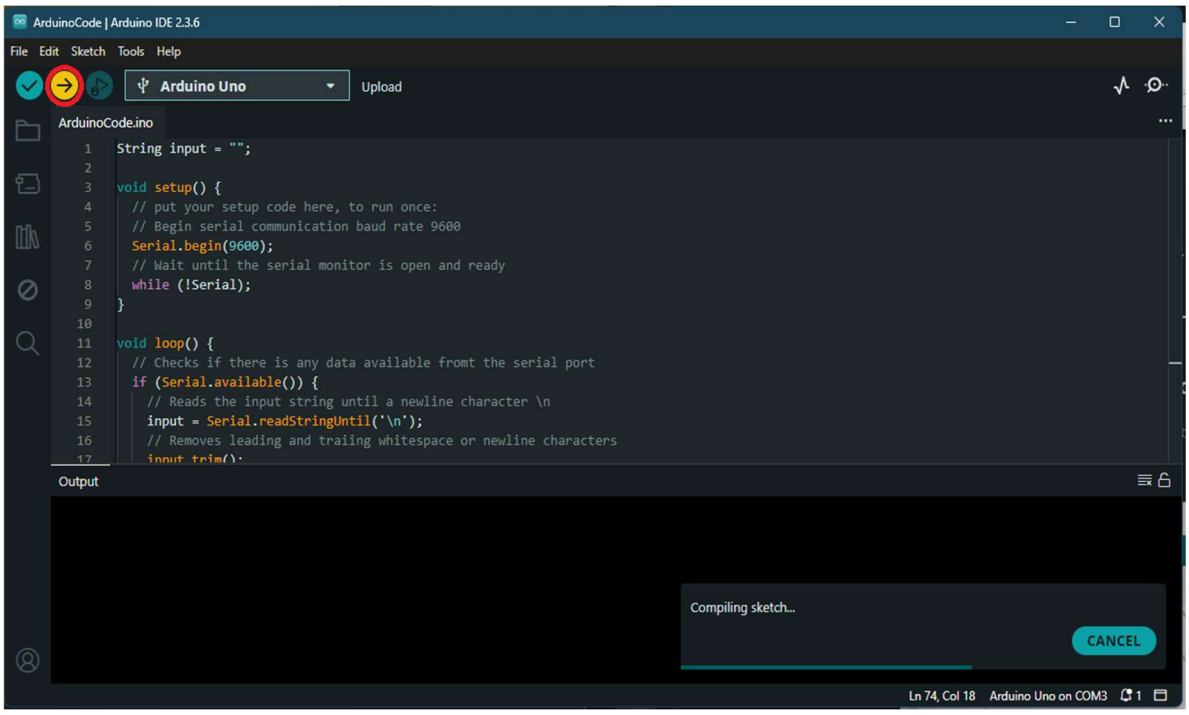
1. Selecting the COM port — Type the COM port where the microcontroller is connected (Open Arduino IDE to check if unsure)



2. Entering **baud rate**



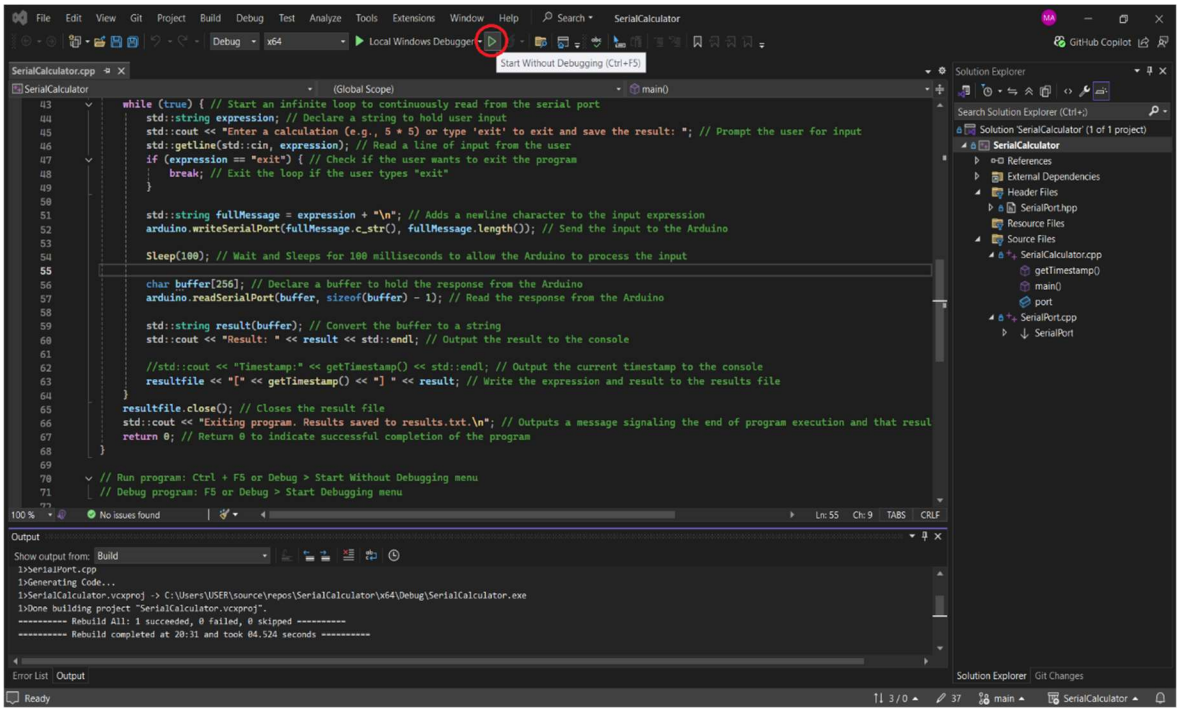
Uploading *ArduinoCode.ino* to Microcontroller



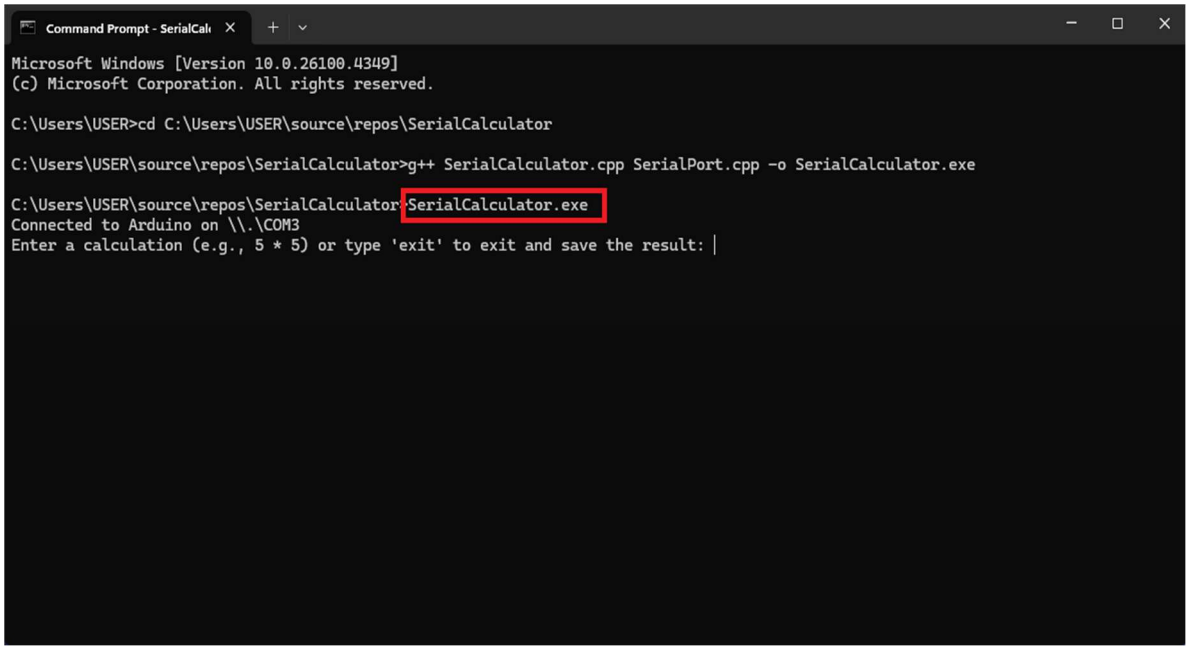
Running the App — via Visual Studio | Console | .exe file

The app can be run through three different ways:

1. Visual Studio



2. Console app

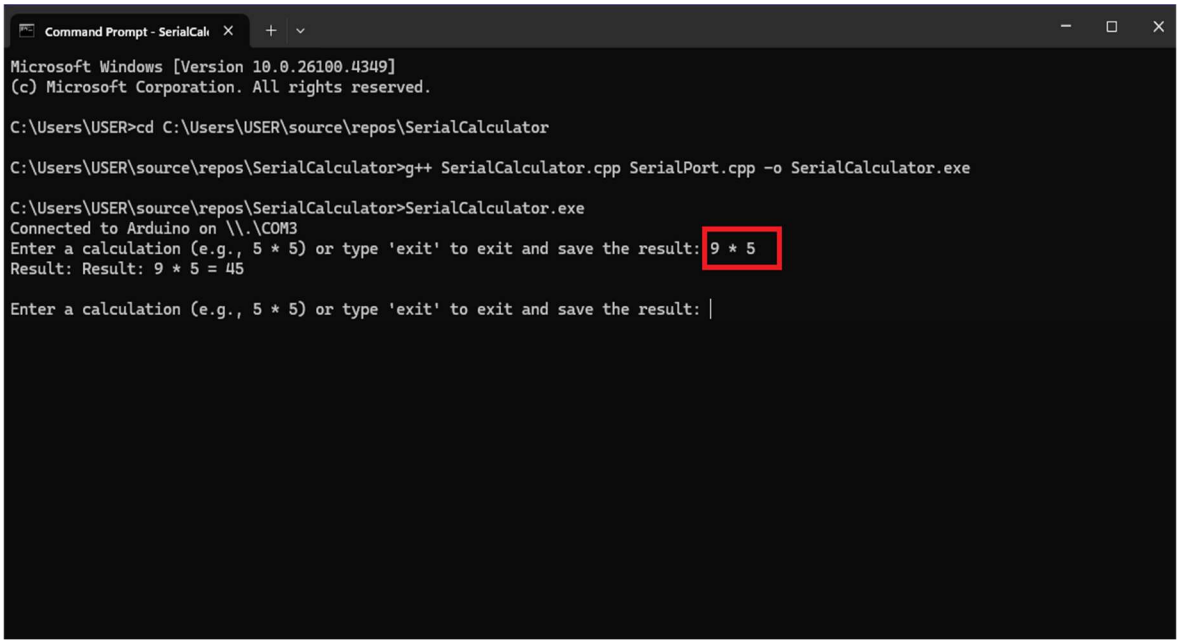


3. Project folder

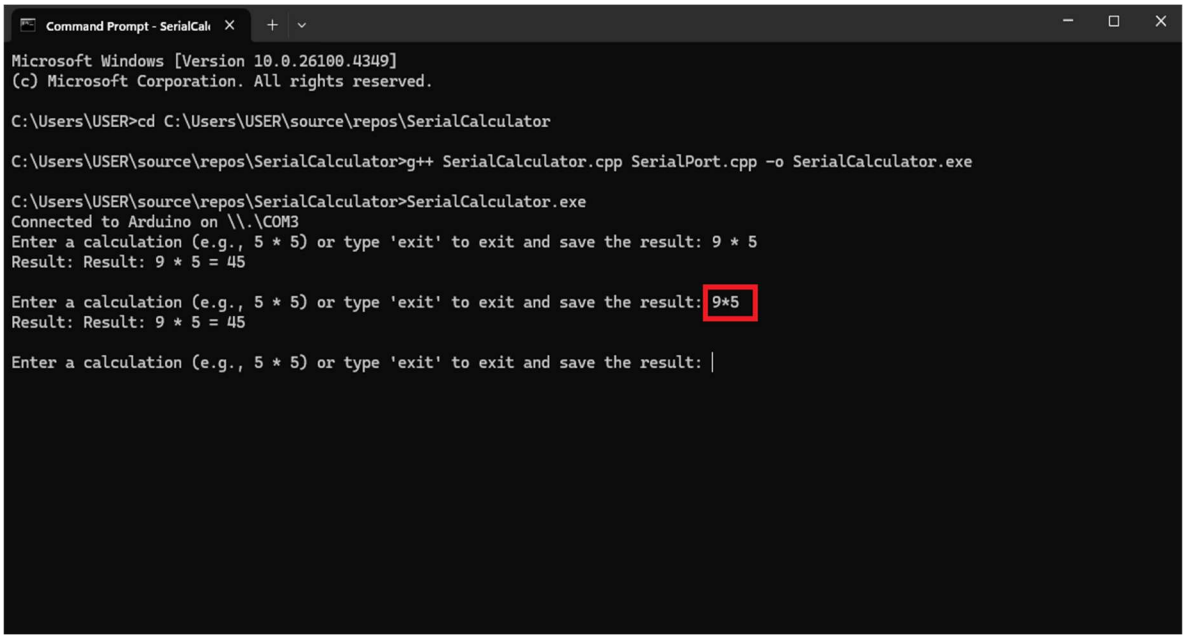
Entering a Mathematical Expression

Typically, there are two ways for the expression to be entered that would be considered valid by the microcontroller and it is of this format: `<number1> <operator> <number2>`

1. Sample entry 1



2. Sample entry 2

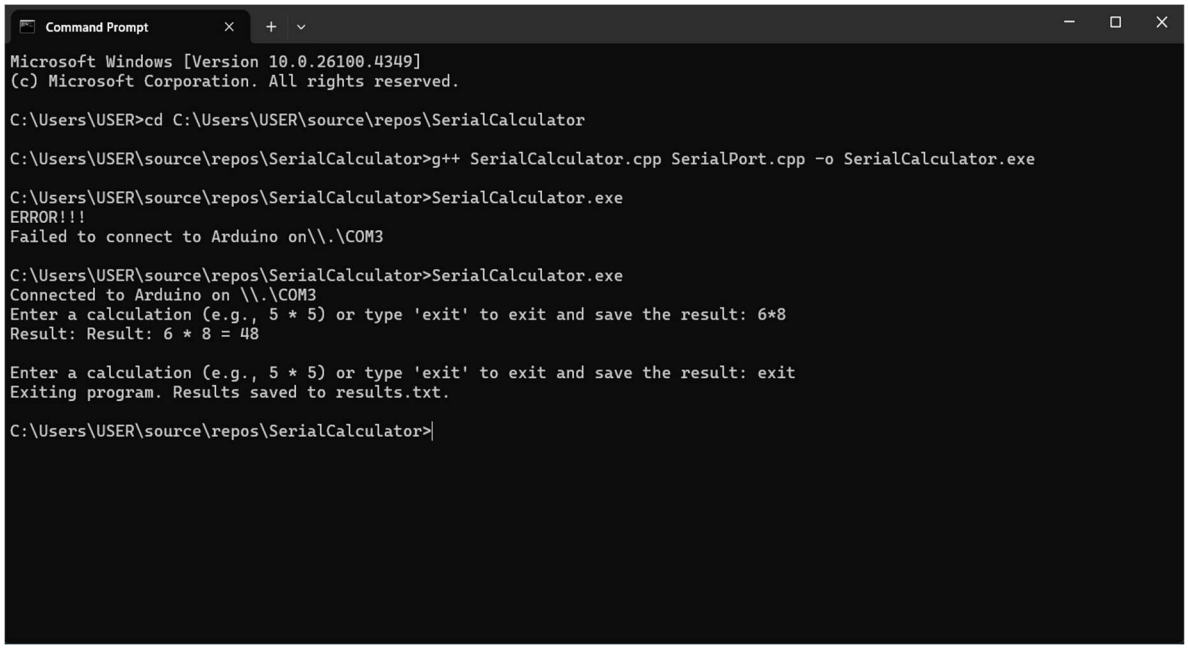


Saving the Result

Once the user has completed the desired operations, to save the **results** from the calculations, the user must type the command **'exit'** to indicate to the program that the set of operation is complete. It then saves the operations to a ***results.txt*** files in **the project folder**.

Note: Abruptly **closing** the **command window/Console App** would result in **failure** of the program in **saving** the previous operations to the **file**.

All **previous operations** are shown in the file.



Checking the *result.txt* File Location

To check the **history** of operations previously performed, go to the **project folder** and find the **results.txt** file. You can open the file with any compatible software including the native **notebook** software that comes pre-installed with **Windows**.

Error Messages

- *"Error: Division by zero"* – Division by zero attempted
- *"Error: Invalid operator"* – Unsupported operator used
- *"Error: No valid operator found"* – No operator detected in input
- *"Error: Invalid input format"* – Incorrect input string

Troubleshooting

Common Issues

"Failed to connect to Arduino"

- **Cause:** Incorrect **COM** port, Arduino not connected or Serial monitor still open in Arduino IDE
- **Solution:**
    - Close the serial monitor in Arduino IDE
    - Ensure corrected port is selected by updating port variable in code (this can be verified via the Arduino IDE)
    - Ensure Arduino is connected
    - Ensure the proper drivers are installed, if not, check Device Manager

“Could not open results file”

- **Cause:** File permissions or disk space issues
- **Solution:**
    - Check available disk space
    - Ensure directory is writable
    - Run application as administrator

Arduino not responding

- **Cause:** Arduino not programmed or serial connection issues
- **Solution:**
    - Verify Arduino firmware is uploaded
    - Check serial monitor in Arduino IDE
    - Try different USB cable or port

Incorrect Calculation

- **Cause:** Input parsing issues or precision limitations
- **Solution:**
    - Verify input format matches supported patterns (Format: <number1> <operator> <number2>)
    - Check for extra spaces or characters
    - Understand floating-point precision limits