

CSC 3210

Computer Organization and Programming

Lab Work 9

Dr. Zulkar Nine

mnine@gsu.edu

Georgia State University

Summer 2021

Learning Objective

- Indirect Addressing & Indexed Operands, Loop instruction, and Logical Instructions

Disclaimer

- The process shown in these slides might not work in every single computer due to Operating system version, Microsoft Visual Studio versions and everything.
- If you find any unusual error, you can inform the instructor.
- Instructor will help you resolve the issue.

Attendance!

Lab Work 9 Instructions

- Lab 9(a): Loop instruction
- Lab 9(b): Nested Loop
- Lab 9(c): Logical Instruction

Plan early ...

- You have one week time to submit the lab
- Start early
- If you have issues
 - Email TA or instructor
 - Stop by during office hours
- Start working **at the last moment** is not a good idea.
- Appendix A shows how to check memory data and Appendix B shows how to create a new project.

Problems in this lab

- Before running the problems in Visual studio,
- You can try to solve them in a paper for practice.
- You might see similar questions in the quizzes and exam.
- During the exam you might need solve similar problems without visual studio.

Directions

- Create a new application for every question.
- Do not use one application with multiple .asm files
- The Appendix has the steps for creating a new project

Lab 9(a)

Loop instruction

Submission

Loop Instruction

Problem 2: Write and run a program to find the values of each **destination operand**:

```
; Summing an Array (SumArray.asm)
```

```
.data
```

```
intarray DWORD 10000h,20000h,30000h,40000h
```

```
.code
```

```
main PROC
```

```
mov edi,OFFSET intarray ; 1: EDI = address of intarray
```

```
mov ecx,LENGTHOF intarray ; 2: initialize loop counter
```

```
mov eax,0 ; 3: sum = 0
```

```
L1: ; 4: mark beginning of loop
```

```
add eax,[edi] ; 5: add an integer
```

```
add edi,TYPE intarray ; 6: point to next element
```

```
loop L1 ; 7: repeat until ECX = 0
```

EAX =?

Submission Instruction – 9(a)

- Debug until you reach “INVOKE ExitProcess, 0”.
 - Take a screenshot of the code and register window at the end
 - Record the content of the EAX register
 - **Then explain the register content.**

Lab 9(b)

Nested loop

Submission

Nested Loop

Problem 3: Write and run a program to find the values of each **destination operand**:

```
.data
temp dword ?

.code
mov eax,0
mov ecx,10      ; outer loop counter
L1:
    mov eax,3
    mov temp,ecx
    mov ecx,5    ; inner loop counter
    L2:
        add eax,5
        loop L2    ; repeat inner loop
    mov ecx, temp
    loop L1        ; repeat outer loop
```

EAX =?

Submission Instruction – 9(b)

- Debug until you reach “INVOKE ExitProcess, 0”.
 - Take a screenshot of the code and register window at the end
 - Record the EAX register.
 - Also add the screenshot
 - **Then explain the EAX register content.**

Lab 9(c)

Logical Instructions

Submission

Logical Instructions

Problem 4: Write and run a program to find the values of each **destination operand**:

```
.code
mov al,01101111b
and al,00101101b    ; a. Al=
mov al,6Dh
and al,4Ah          ; b. Al=
mov al,00001111b
or al,61h           ; c. Al=
mov al,94h
xor al,37h          ; d. Al=
```


Submission Instruction – 9(c)

- Debug through each line of code.
 - Execute the instruction
 - Take a screenshot of the code and register window
 - Record the line number, instruction, Register values in the answer sheet.
 - Also add the screenshot
 - **Then explain the register or memory contents.**

Appendix A

Checking Memory Data

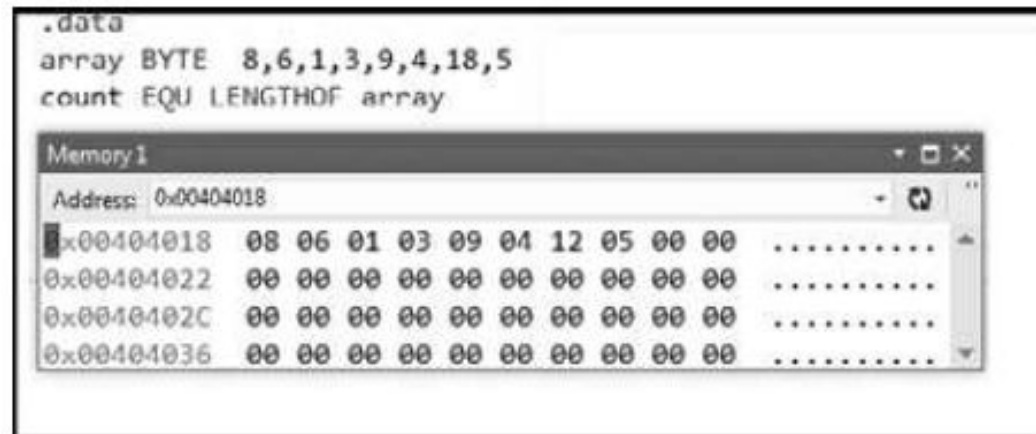
Checking Memory Data

- Use **Memory window** to verify the values of memory locations.
 - **To activate Memory window**, run the debugger, go to debug menu and click on windows, open it, go to **Memory** then choose **Memory1**.
 - When you run your program and step over every line you will see the changed values marked with red color.

You Must be in the Debugging Mode to see the memory or the register window

Checking Memory Data

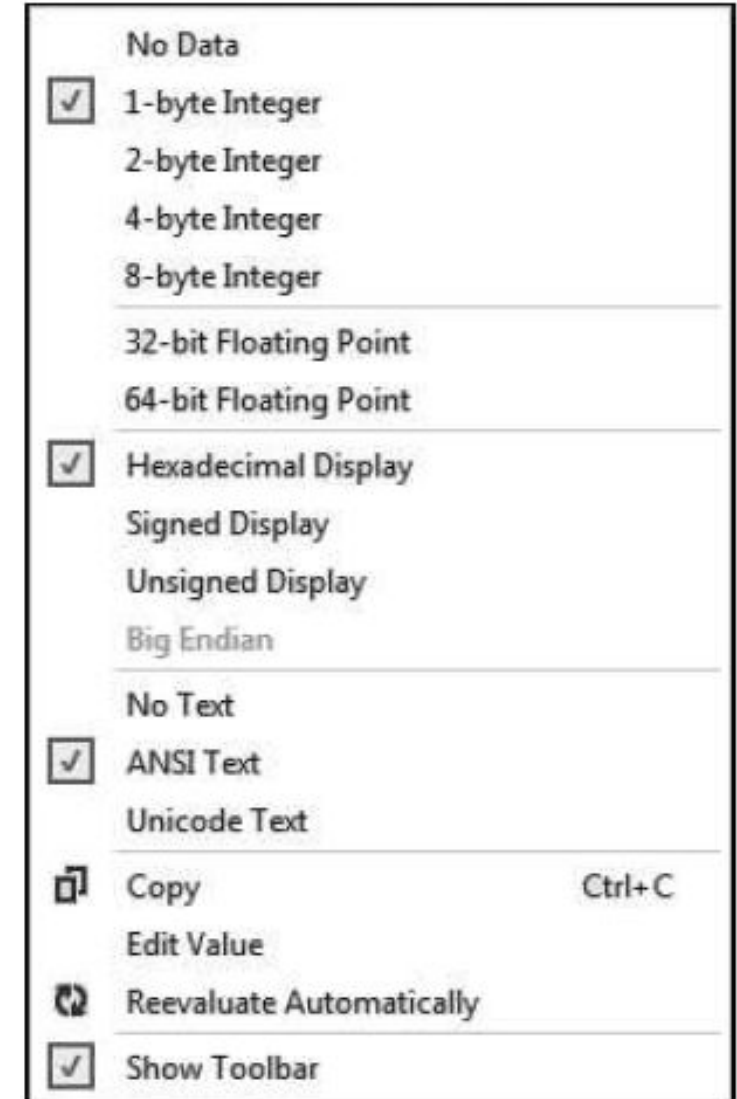
- To activate Memory window,
 - if you want to see the location of your variable in the memory,
 - Memory window search box (on the top of the memory window, Address:)
 - write **&** follow it with the variable name: example: **&myVall**.
 - This will take you to the memory locations of your program (.data section).



Checking Memory Data

- To activate Memory window,

- You can right-click inside the memory window
- You will see **Popup menu for the debugger's memory window**
- You can choose how you want to group your bytes: by 1,2,4, or by 8
- You can also presents data in **hexadecimal, signed, or unsigned** display

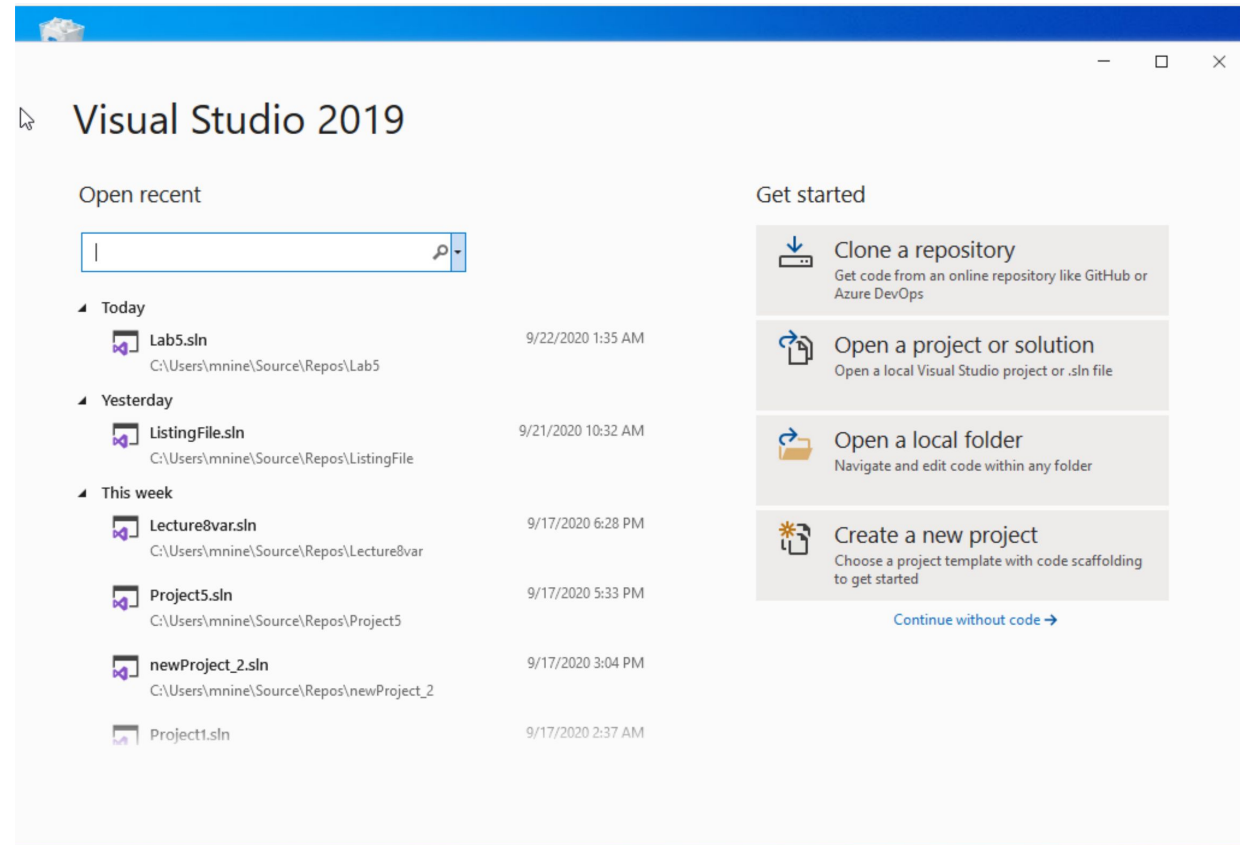


Appendix B

Create a Project

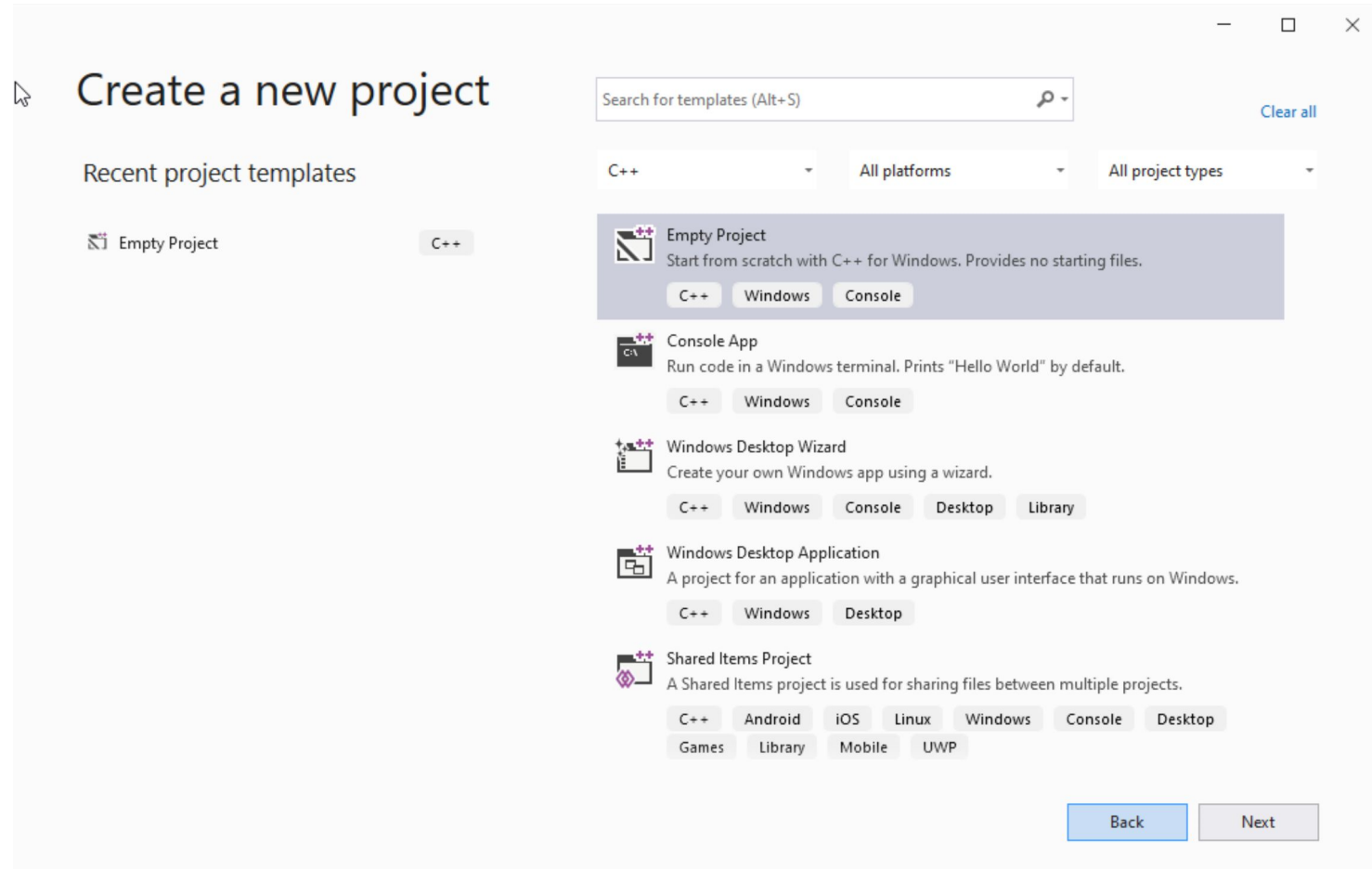
Step 1: Create a project (1)

- (1) Start Visual Studio
- (2) Click Create a new Project



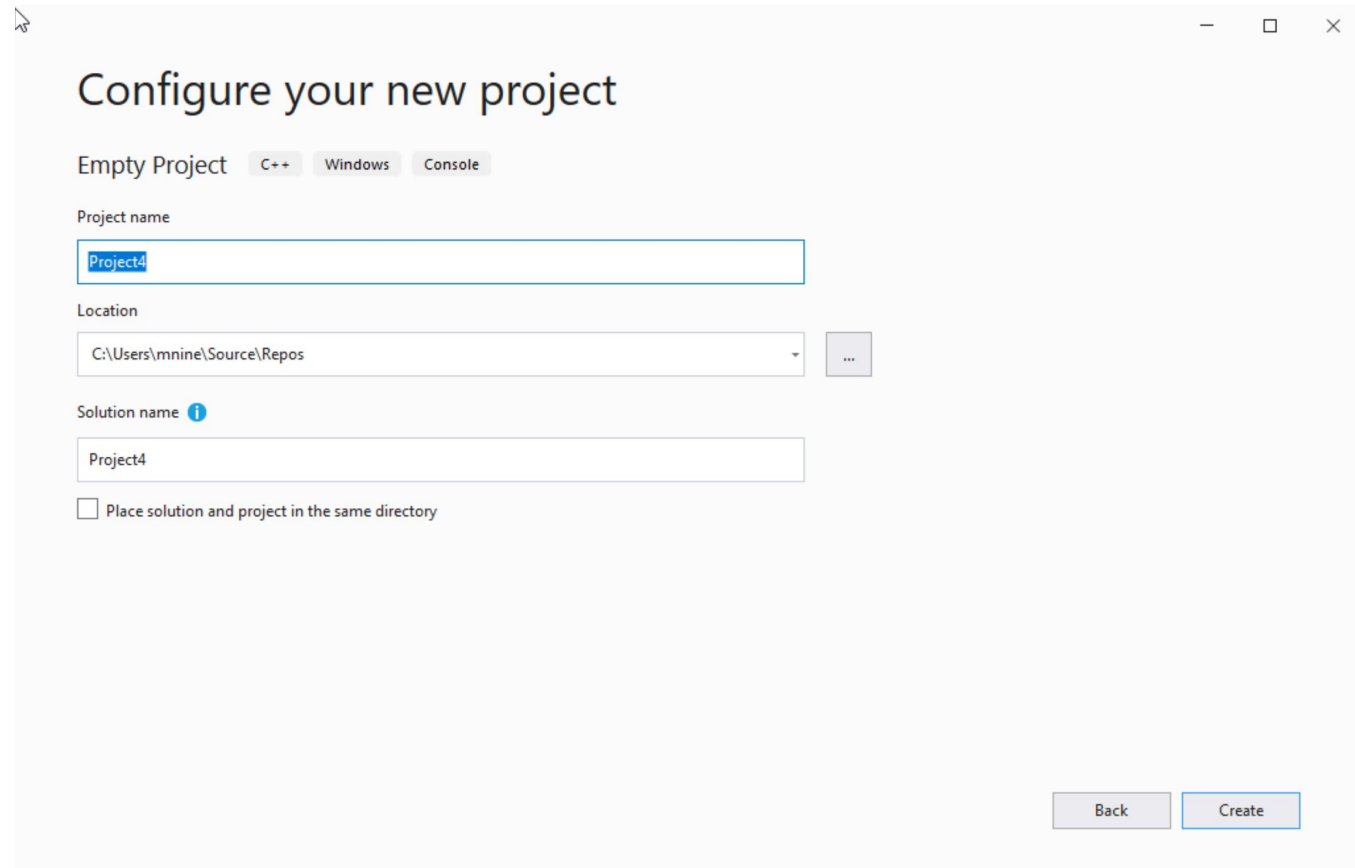
Step 1: Create a project (2)

- (1) Select C++ as language
- (2) Select Empty Project
- (3) Click Next



Step 1: Create a project (3)

- (1) You can change the project name as you like
- (1) Also, you can change the project location
- (2) Click Next



The screenshot shows the 'Configure your new project' dialog box in Visual Studio. The title bar includes standard window controls. The main heading is 'Configure your new project'. Below it, there are tabs for 'Empty Project', 'C++', 'Windows', and 'Console'. The 'Empty Project' tab is selected. The 'Project name' field contains 'Project4'. The 'Location' field shows the path 'C:\Users\mnine\Source\Repos' with a browse button ('...') to its right. The 'Solution name' field, which has an information icon, also contains 'Project4'. At the bottom, there is a checkbox labeled 'Place solution and project in the same directory' which is currently unchecked. In the bottom right corner, there are 'Back' and 'Create' buttons.

Step 1: Create a project (4)

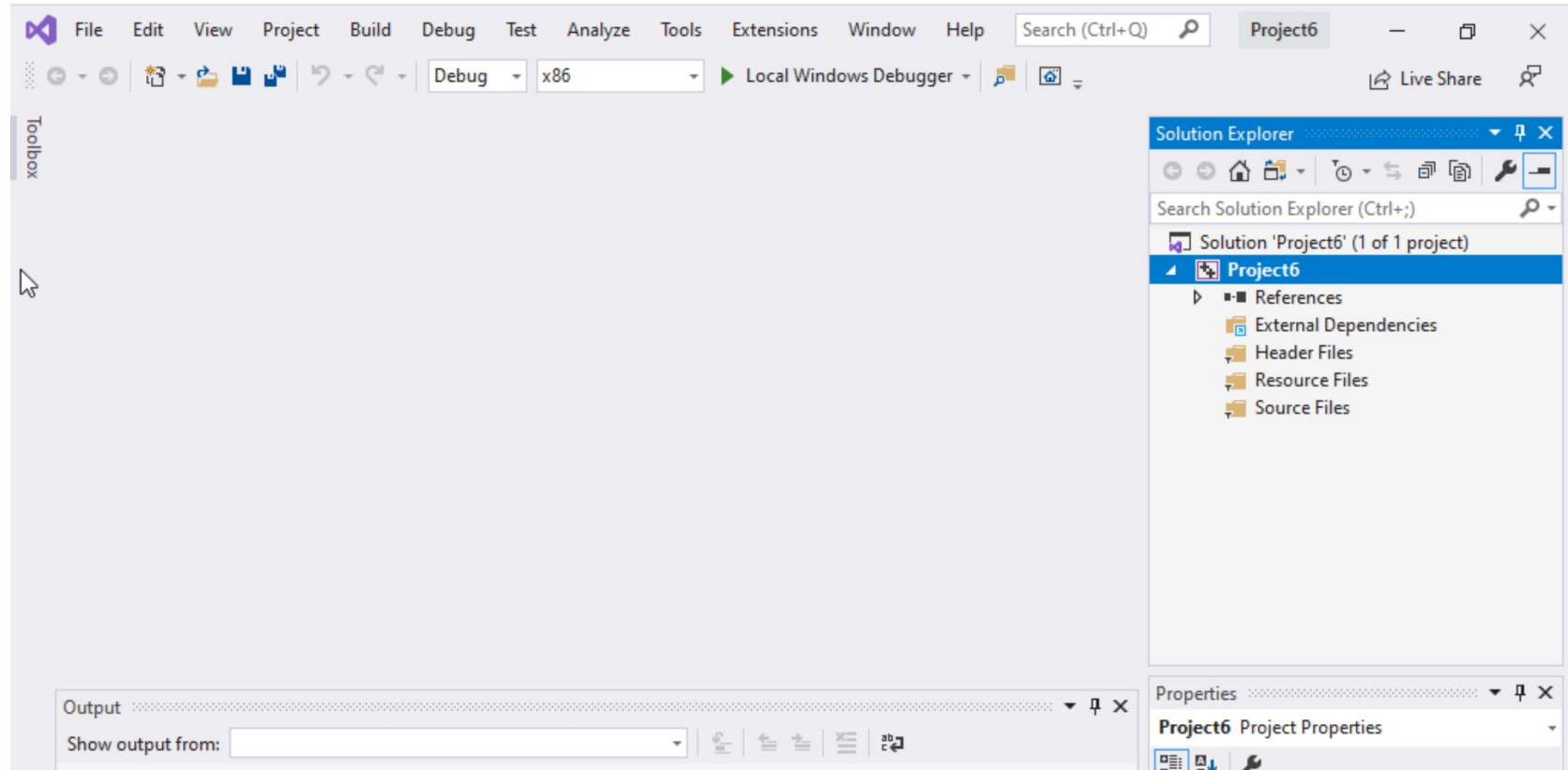
Delete the

Following folders:

- Header files

- Resources Files, and

- Source Files



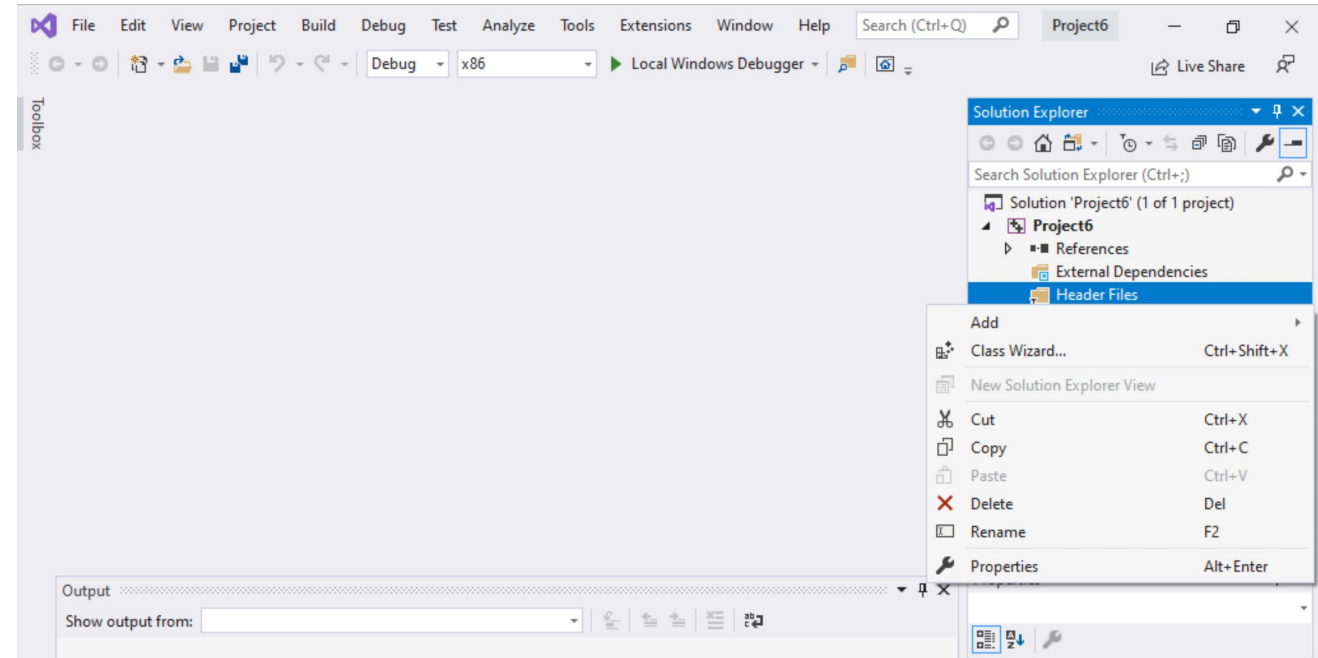
Step 1: Create a project (5)

To delete :

Select the folders

Right click on it

Select delete



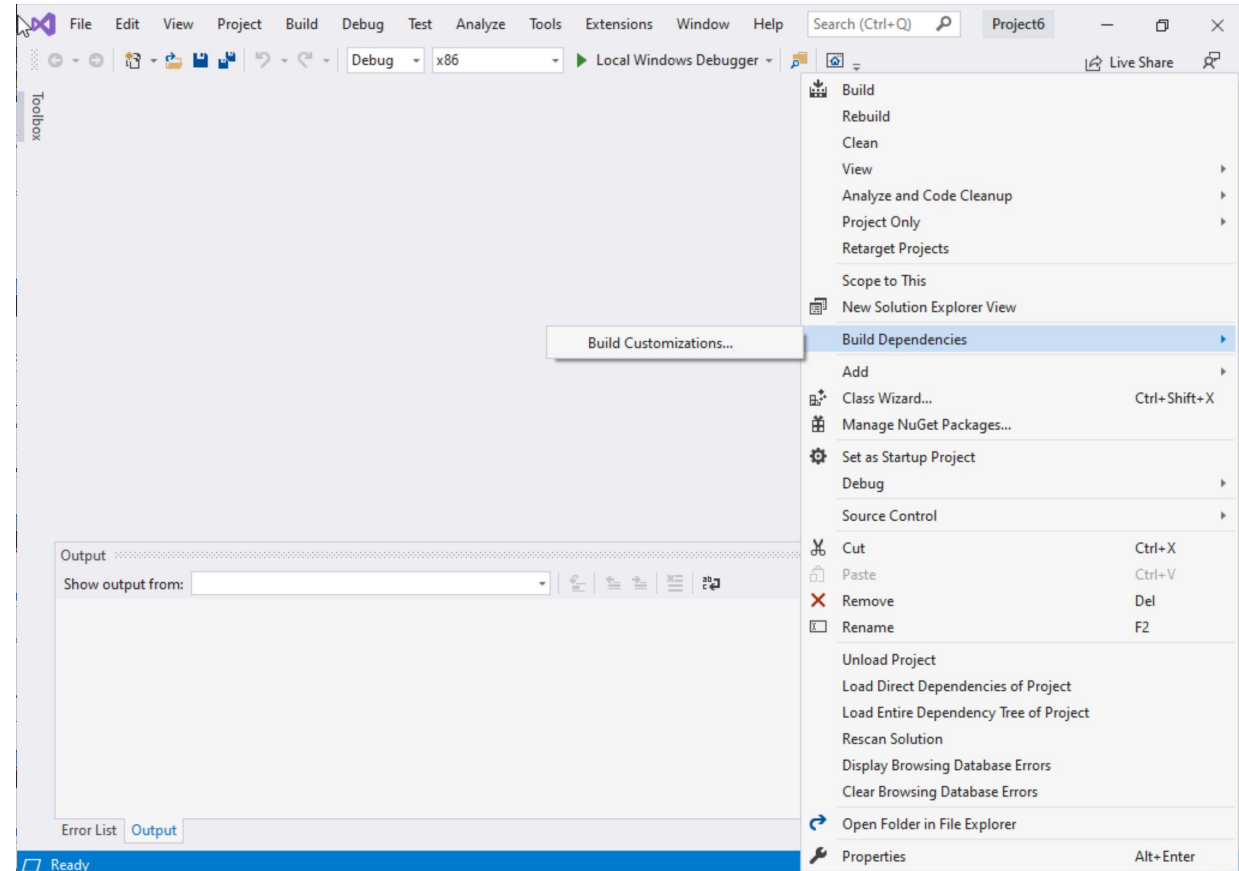
Step 1: Create a project (6)

Select Project Name on solution explorer

Right click on it

Go to Build Dependencies

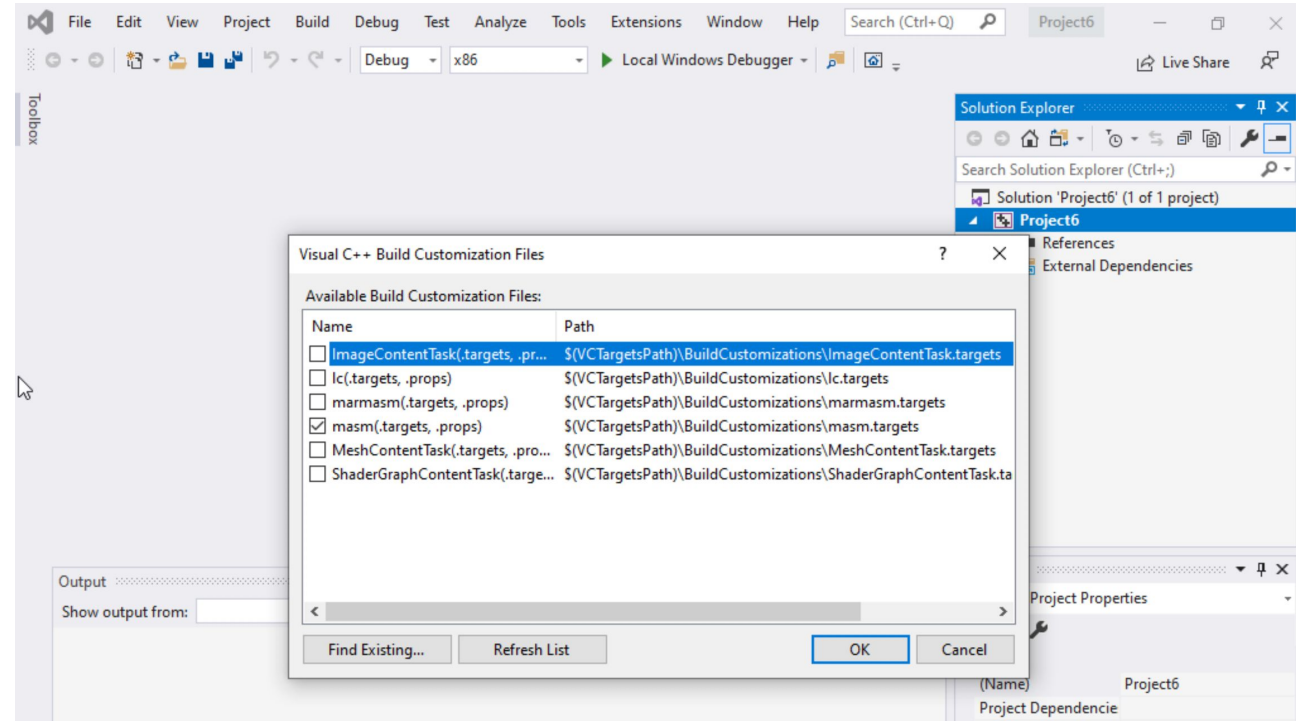
Click on Build Customizations



Step 1: Create a project (7)

Select masm(.target, .props)

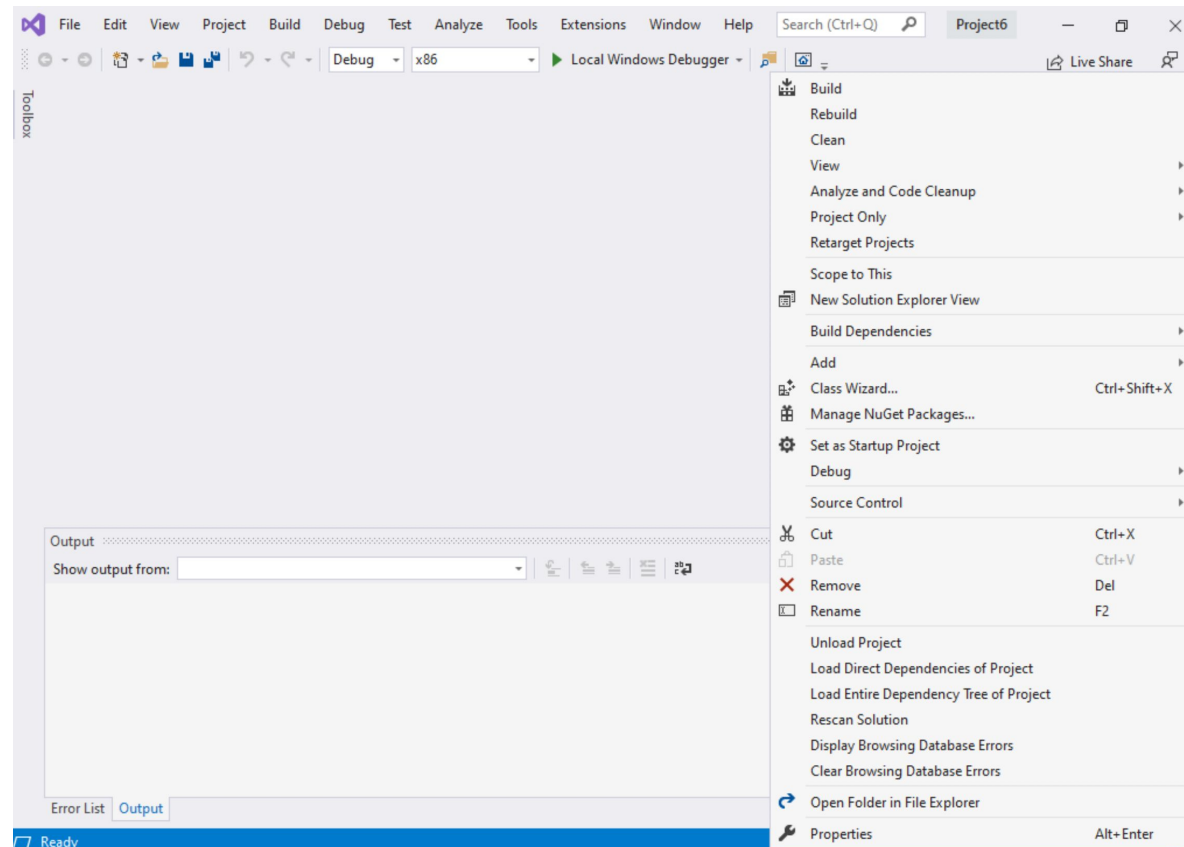
Click ok



Step 1: Create a project (8)

Right click on the Project name in the solution explorer

Click properties



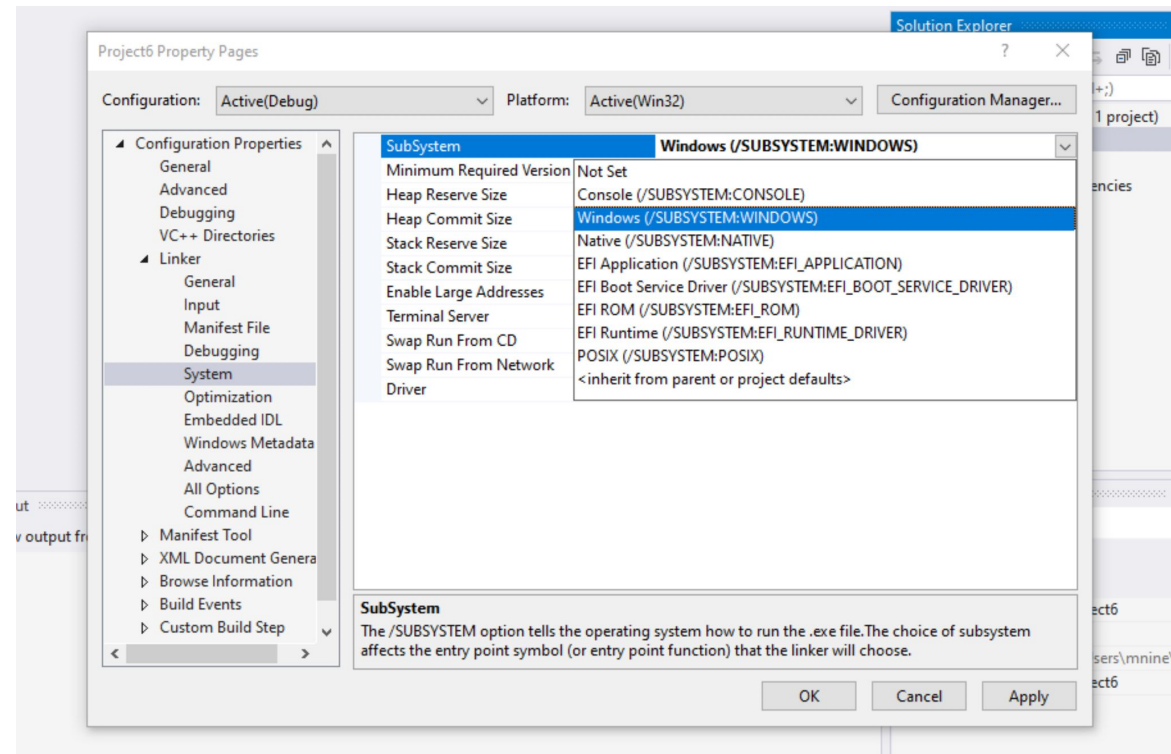
Step 1: Create a project (9)

Expand the 'Linker'

Select 'System'

Select Windows(/SUBSYSTEM:WINDOWS)

Click OK



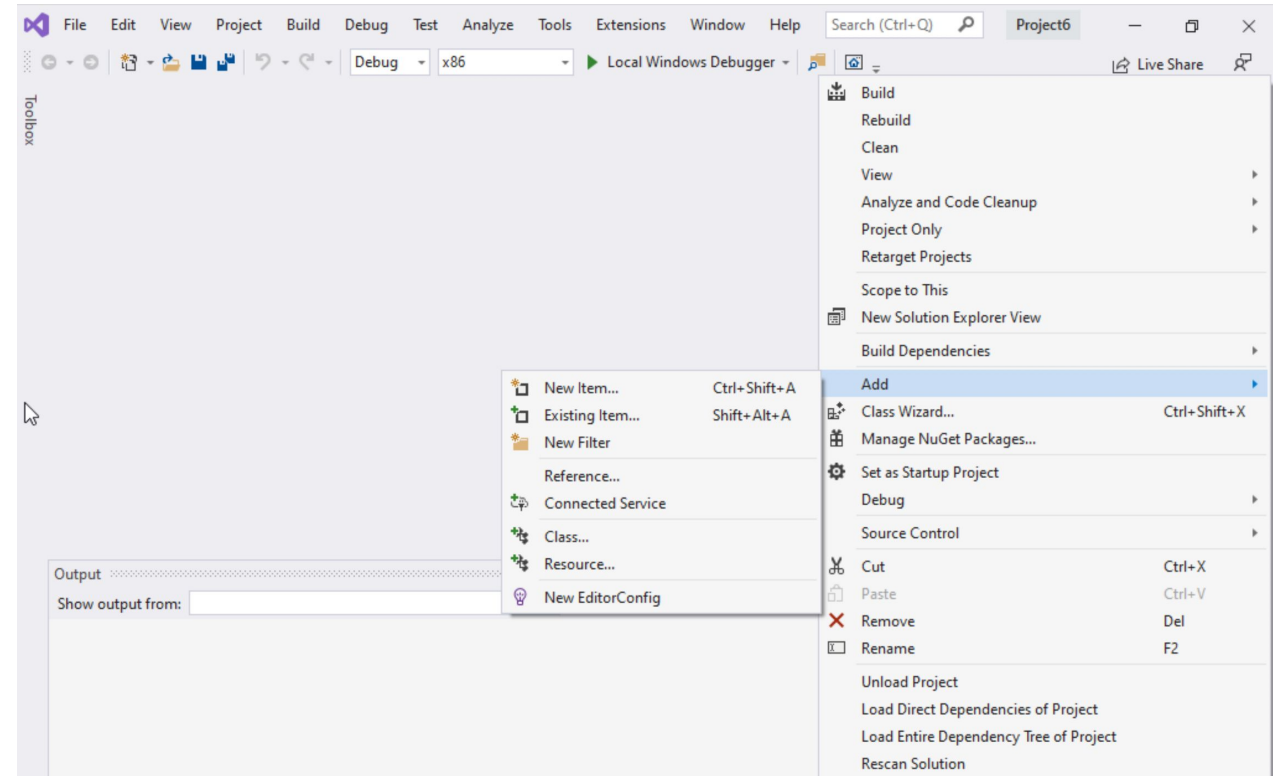
Step 1: Create a project (10)

Select Project name on solution explorer

Right click on it

Expand Add

Choose New Item

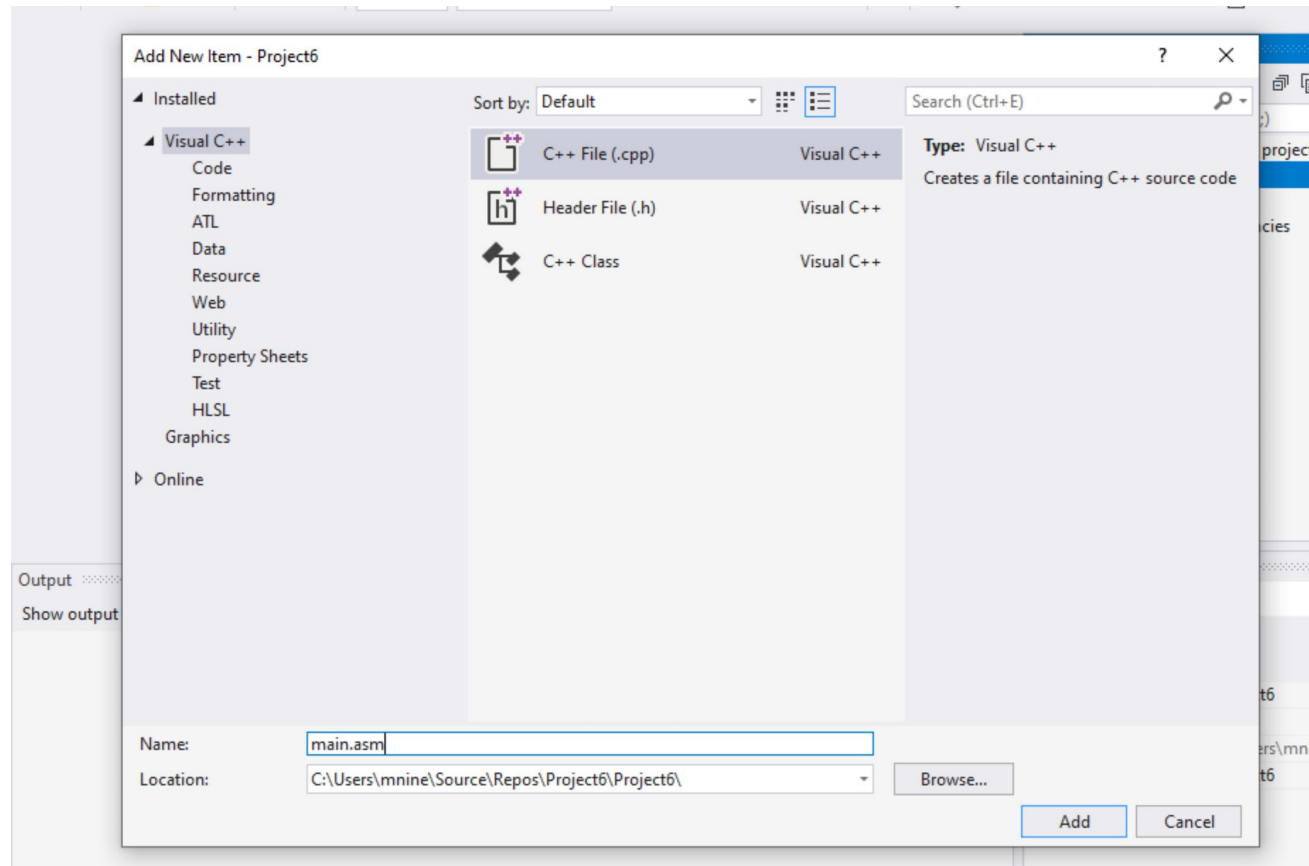


Step 1: Create a project (11)

Select C++ File(.cpp)

Name: main.asm

Click Add



Step 1: Create a project (12)

Select main.asm

Add your code

In the main.asm File.

