CSC 3210
Computer Organization and Programming

Lab Work 11

Dr. Zulkar Nine

mnine@gsu.edu

Georgia State University

Fall 2021

Learning Objective

Logical Instructions, and If & While statements

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 11 Instructions

- Lab 11 : Logical Instructions, and If & While statements

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.

- AppendixA shows how to check memory data and Appendix B shows how to create a new project.

Problems in this lab

- You might see similar questions in the quizzes and exam.
- During the exam you might need solve similar problems without visual studio.

Compound Expression with **AND** **(Example1)**

- When implementing **the logical AND operator**, consider that HLLs use **<u>short-circuit</u> evaluation**
- In the following example, **<u>if the first expression is false</u>**<u>, the</u> **<u>second expression is skipped</u>**:

```
if (al <= bl) AND (bl >= cl)
  X = 1;
else
  Y = 1
```

Compound Expression with AND (Example1)

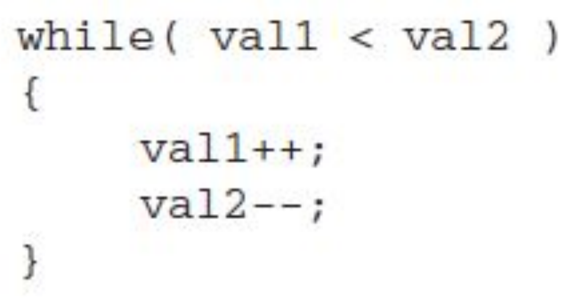
- When implementing **the logical AND operator**, consider that HLLs use **short-circuit evaluation**
- In the following example, **if the first expression is false**, the **second expression is skipped**:

```
if (al <= bl) AND (bl >= cl)
   X = 1;
else
   Y = 1
```

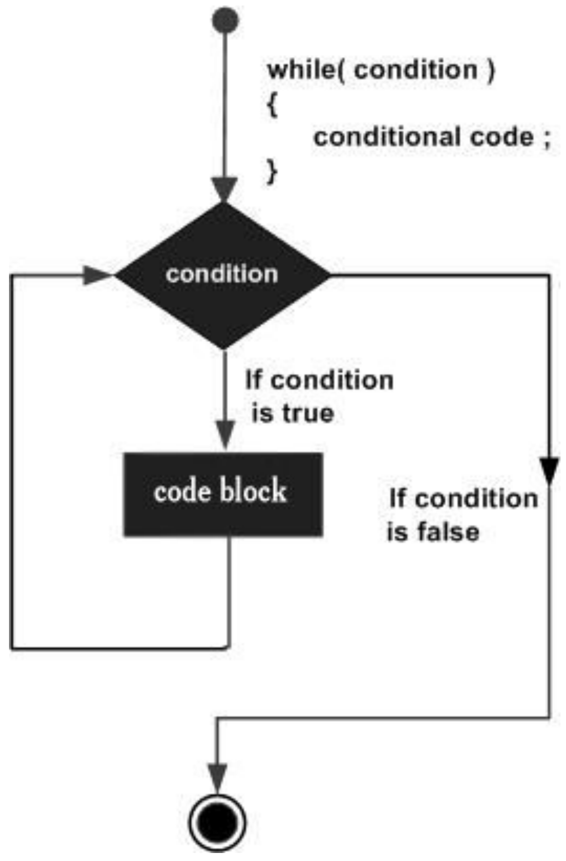
```
    cmp al,bl           ; first expression...
    jbe  L1
    jmp Else_block
L1:
    cmp bl,cl            ; second expression...
    jae  If_block
    jmp Else_block
If_block:               ; both are true
    mov X,1              ; set X to 1
    jmp next
Else_block:
    mov Y, 1
Next:
```

WHILE Loops

- **A WHILE loop tests a condition first** before performing a block of statements.
- **As long as** the loop condition remains true, **the statements are repeated**.

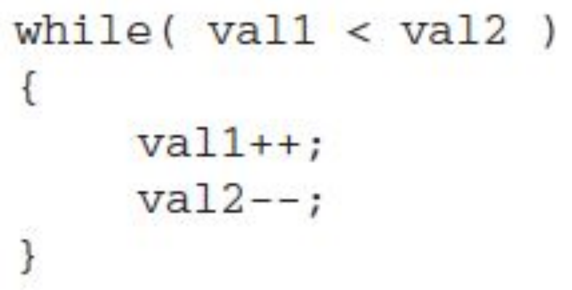


```
while( val1 < val2 )
{
        val1++;
        val2--;
}
```



- When implementing this structure in assembly language,
  - it is convenient to **reverse the loop condition**
  - and **jump** to endwhile **if a condition becomes true**

WHILE Loops: Example1



```
while( val1 < val2 )
{
    val1++;
    val2--;
}
```

Reverse The loop Condition

G and L

```
            mov eax,val1          ; copy variable to EAX?
beginwhile:
            cmp eax,val2          ; if not (val1 < val2)
      jge endwhile                ; exit the loop
      inc eax                     ; val1++;
      dec val2                    ; val2--;
      jmp beginwhile              ; repeat the loop
endwhile:
      mov val1,eax                ; save new value for val1
```

WHILE Loops: Example1

```
While ( (val1 != val2) AND (val1 > 0){
    Val3++
}
```

G and L

WHILE Loops: Example1

```
While ( (val1 != val2) AND (val1 > 0){
    Val3++
}
```

```
                mov eax,val1          ; copy variable to EAX?
beginwhile:
                cmp eax,val2          ; if not (val1 == val2)
        je endwhile
                cmp val1, 0
        jbe endwhile
        inc val3              ; val3++;
        jmp beginwhile        ; repeat the loop
endwhile:
```

G and L

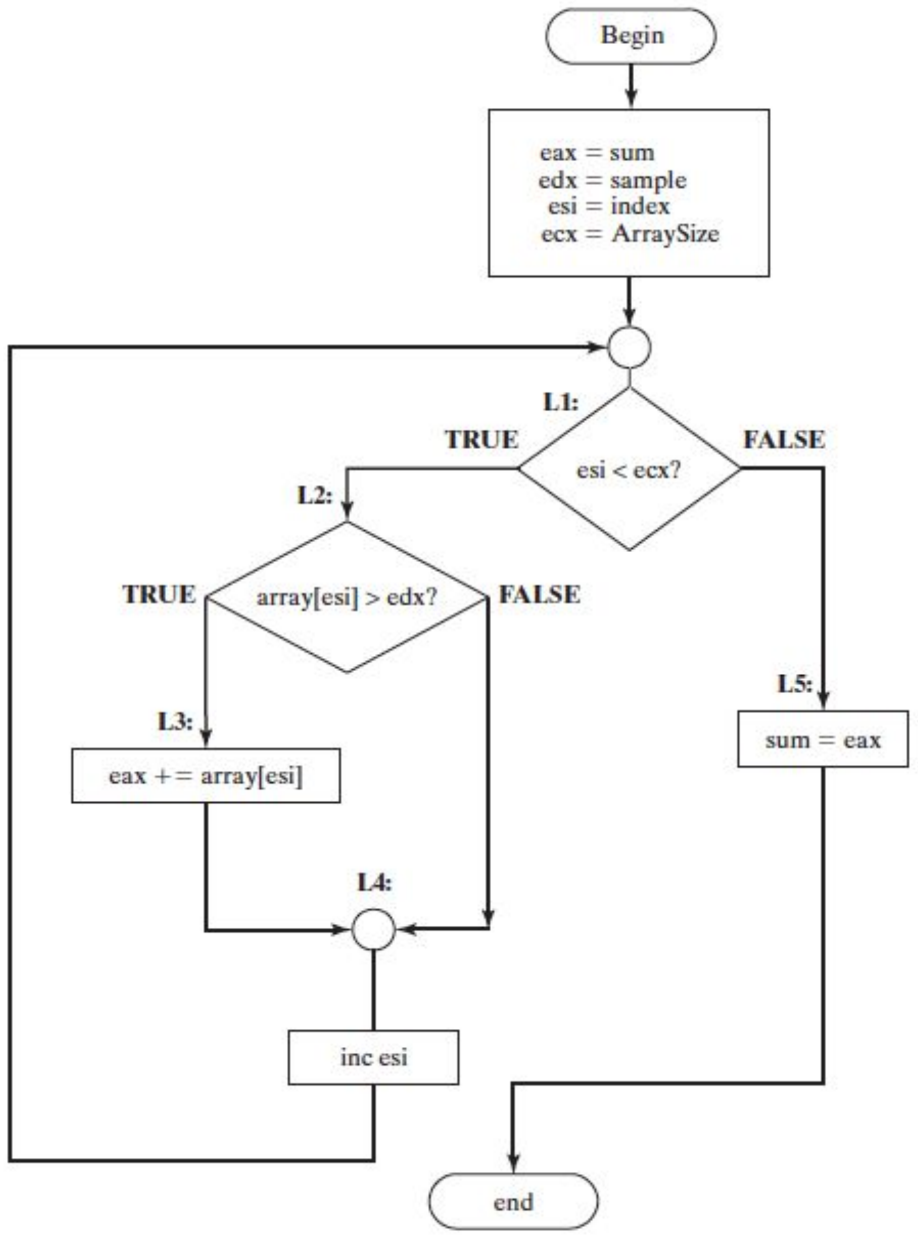
IF and While Statements

Example: Write and run a program to translate the following **while** and **if** statements:

```
int array[] = {10,60,20,33,72,89,45,65,72,18};
int sample = 50;
int ArraySize = sizeof array / sizeof sample;
int index = 0;
int sum = 0;
while( index < ArraySize )
{
    if( array[index] > sample )
    {
       sum += array[index];
    }
    index++;
}
```

WHILE Loops:

- **IF** statement **Nested** in a **Loop**



What will be the value of **SUM?**

```
int array[] = {10,60,20,33,72,89,45,65,72,18};
int sample = 50;
int ArraySize = sizeof array / sizeof sample;
int index = 0;
int sum = 0;
while( index < ArraySize )
{
    if( array[index] > sample )
    {
       sum += array[index];
    }
    index++;
}
```

```
.data
sum DWORD 0
sample DWORD 50
array DWORD 10,60,20,33,72,89,45,65,72,18
ArraySize = ($ - Array) / TYPE array
.code
main PROC
     mov    eax,0               ; sum
     mov    edx,sample
     mov    esi,0               ; index
     mov    ecx,ArraySize
L1:  cmp    esi,ecx             ; if esi < ecx
     jl     L2
     jmp    L5
L2:  cmp    array[esi*4], edx   ; if array[esi] > edx
     jg     L3
     jmp    L4
L3:  add    eax,array[esi*4]
L4:  inc    esi
     jmp    L1
L5:  mov    sum,eax
```

40/4 = 10

Loop

IF

Shift and Rotate Instructions

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

```
mov al,0D4h
shr al,1          ; a.
mov al,0D4h
sar al,1          ; b.
mov al,0D4h
sar al,4          ; c.
mov al,0D4h
rol al,1          ; d.
mov al,0D4h
ror al,3          ; e
```

Shift and Rotate: Instructions Shifting Multiple Doublewords

- **Example:** Write and run a program to find the values of each **destination operand**:

```
.data
ArraySize = 3
array DWORD ArraySize DUP(99999999h)       ; 1001 1001...
.code
mov esi,0
shr array[esi + 8],1    ; high dword
rcr array[esi + 4],1    ; middle dword, include Carry
rcr array[esi],1 ; low dword, include Carry
```

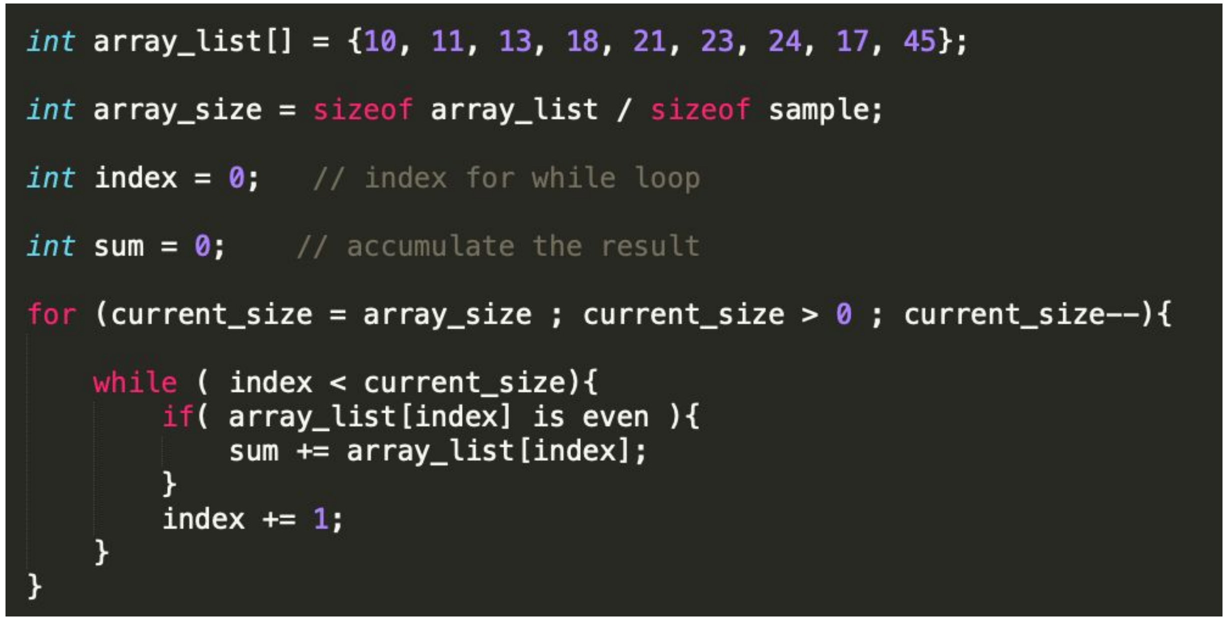
Use the memory window to verify the result

Lab 11

Submission

Submission (1)

- Convert the following pseudo code into assembly code.



```
int array_list[] = {10, 11, 13, 18, 21, 23, 24, 17, 45};

int array_size = sizeof array_list / sizeof sample;

int index = 0;    // index for while loop

int sum = 0;     // accumulate the result

for (current_size = array_size ; current_size > 0 ; current_size--){

    while ( index < current_size){
        if( array_list[index] is even ){
            sum += array_list[index];
        }
        index += 1;
    }
}
```

- Store the result in the variable – sum.

Submission Instruction

- Submit the screenshot of your code.
- Debug your code until you reach INVOKE ExitProcess, 0
- Take a screenshot of the watch window showing variable sum.
  - Submit the screenshot.
- Also, Rename the asm file using your last name as Lastname.asm
  - Submit the ASM file as well.

Appendix A

Checking Memory Data

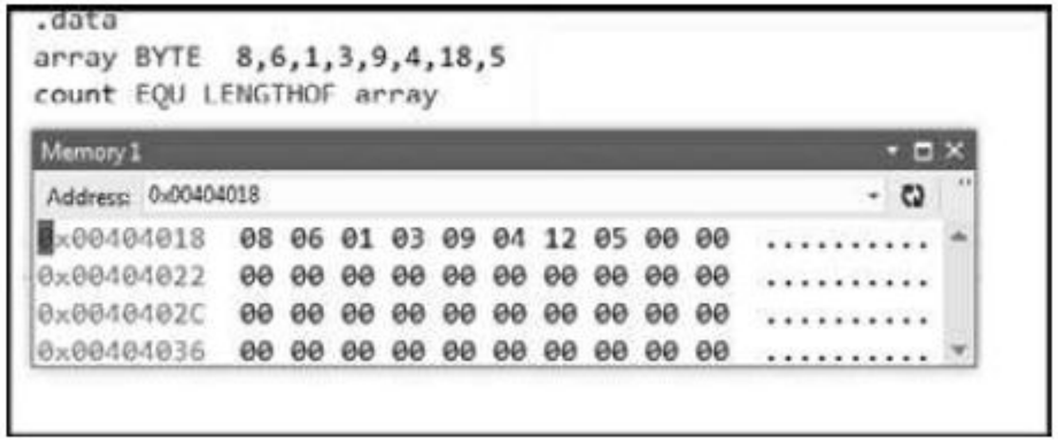
Checking Memory Data

- 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).

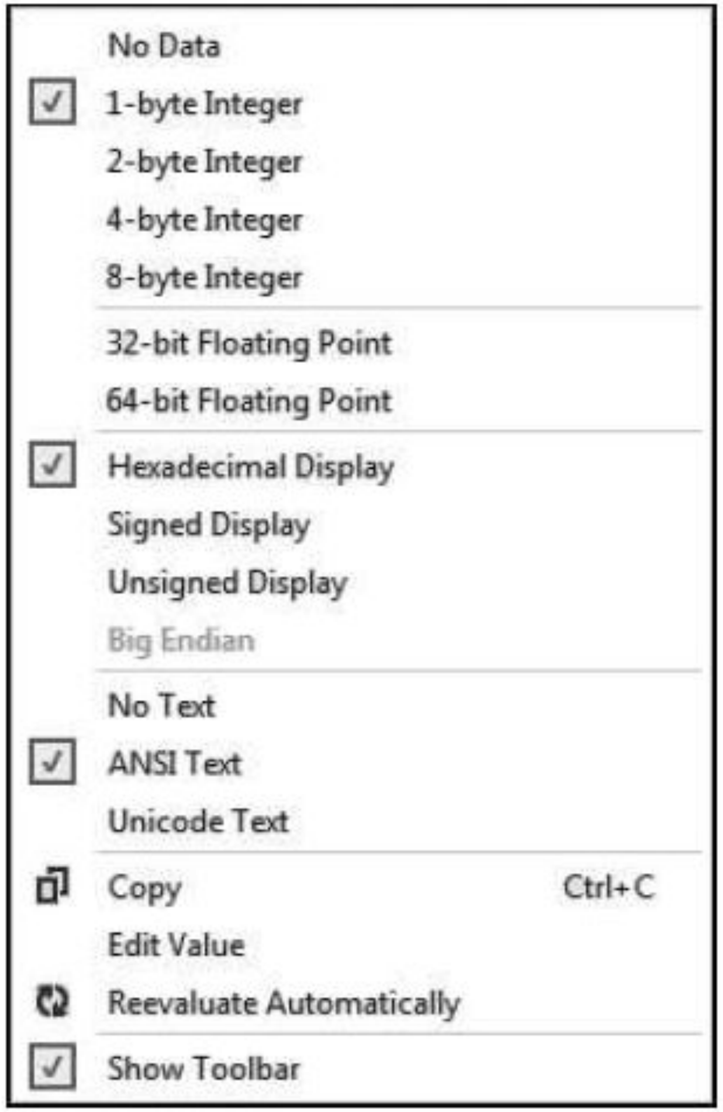


```
.data
array BYTE  8,6,1,3,9,4,18,5
count EQU LENGTHOF array
```

```
0x00404018  08 06 01 03 09 04 12 05 00 00  ..........
0x00404022  00 00 00 00 00 00 00 00 00 00  ..........
0x0040402C  00 00 00 00 00 00 00 00 00 00  ..........
0x00404036  00 00 00 00 00 00 00 00 00 00  ..........
```

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



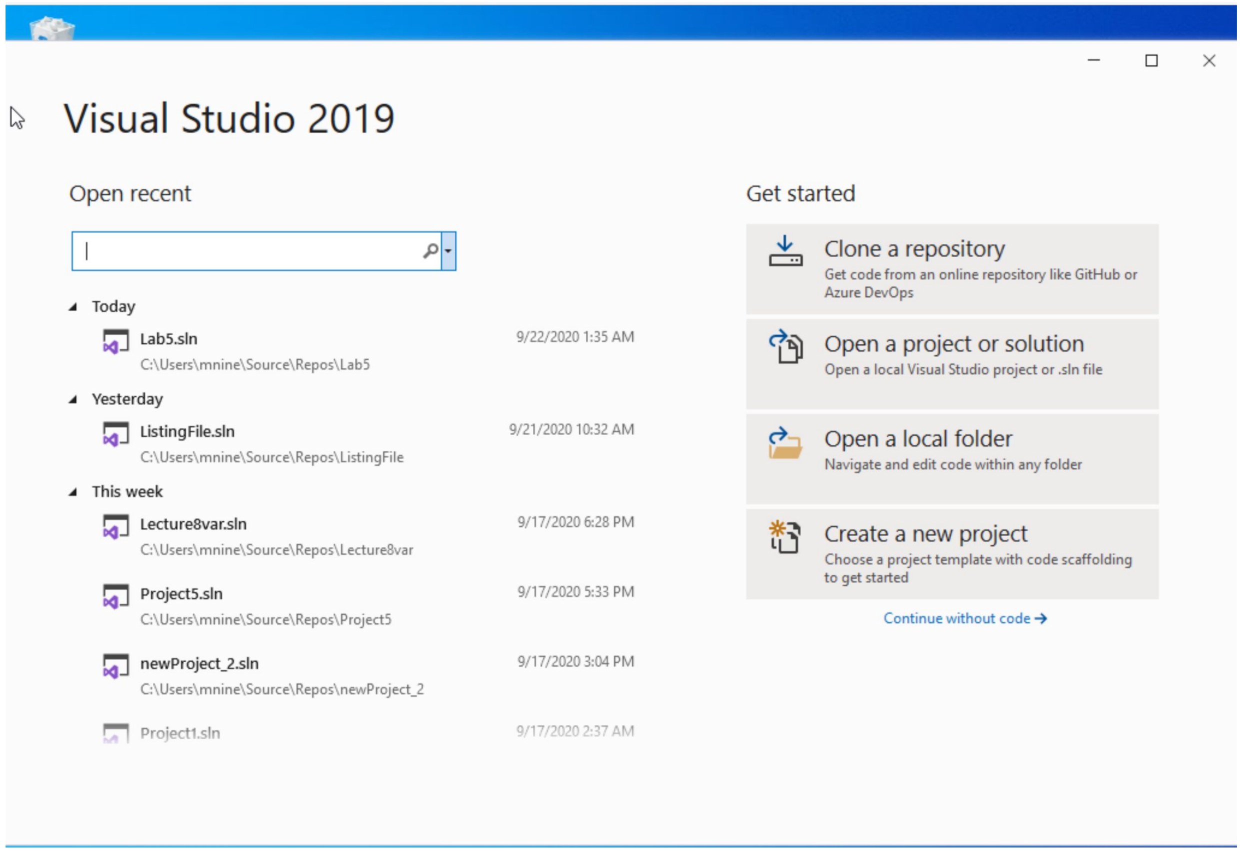
Appendix B

Create a Project

Step 1: Create a project (1)

(1) Start Visual Studio

(2) Click Create a new Project



Visual Studio 2019

Open recent

Today

Lab5.sln — C:\Users\mnine\Source\Repos\Lab5 — 9/22/2020 1:35 AM

Yesterday

ListingFile.sln — C:\Users\mnine\Source\Repos\ListingFile — 9/21/2020 10:32 AM

This week

Lecture8var.sln — C:\Users\mnine\Source\Repos\Lecture8var — 9/17/2020 6:28 PM

Project5.sln — C:\Users\mnine\Source\Repos\Project5 — 9/17/2020 5:33 PM

newProject_2.sln — C:\Users\mnine\Source\Repos\newProject_2 — 9/17/2020 3:04 PM

Project1.sln — 9/17/2020 2:37 AM

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

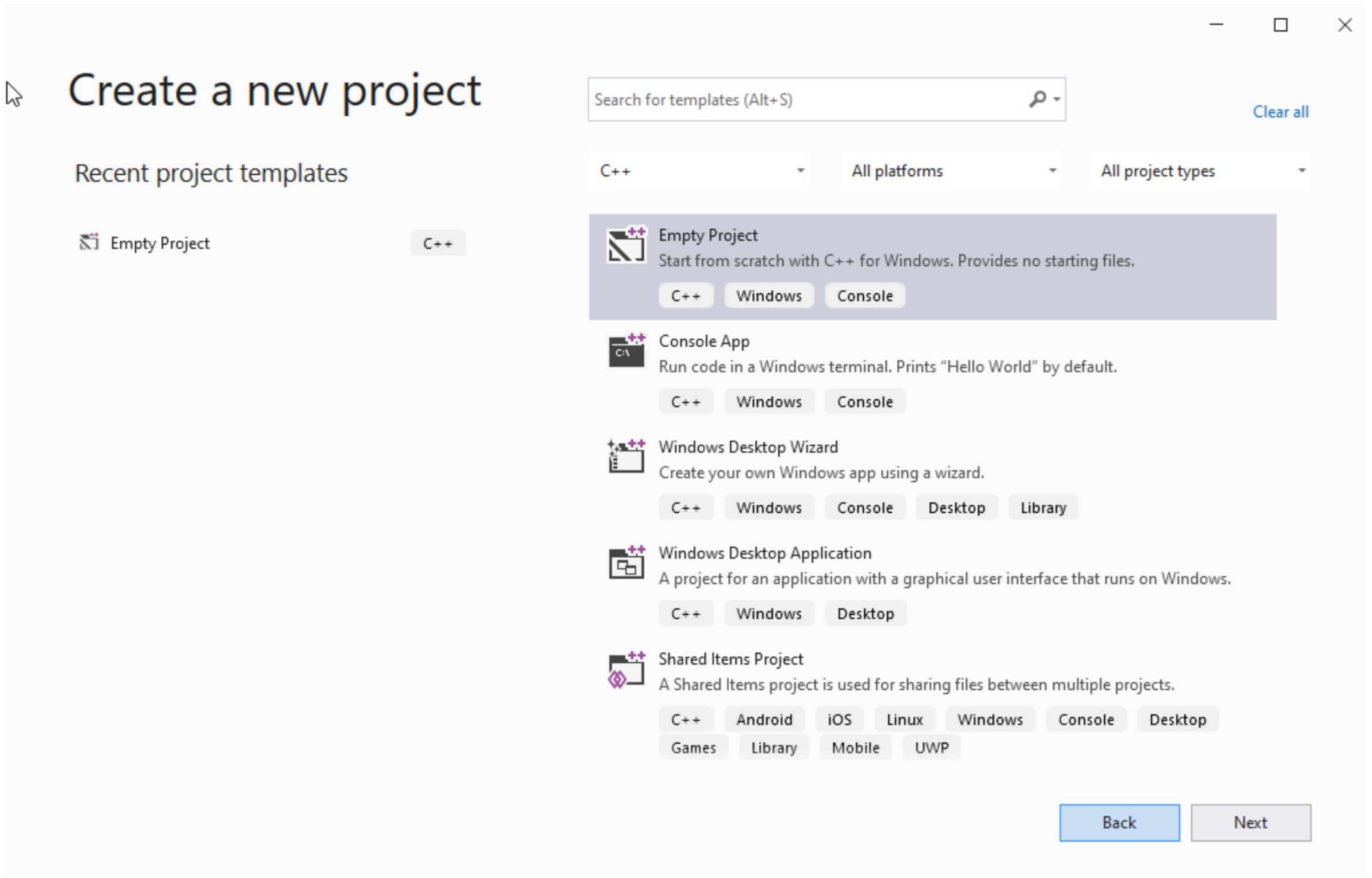
Continue without code →

Step 1: Create a project (2)

(1) Select C++ as language

(2) Select Empty Project

(3) Click Next


Create a new project

Search for templates (Alt+S)

Clear all

Recent project templates

Empty Project C++

C++ | 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

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

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

Shared Items Project
A Shared Items project is used for sharing files between multiple projects.
C++ Android iOS Linux Windows Console Desktop Games Library Mobile UWP

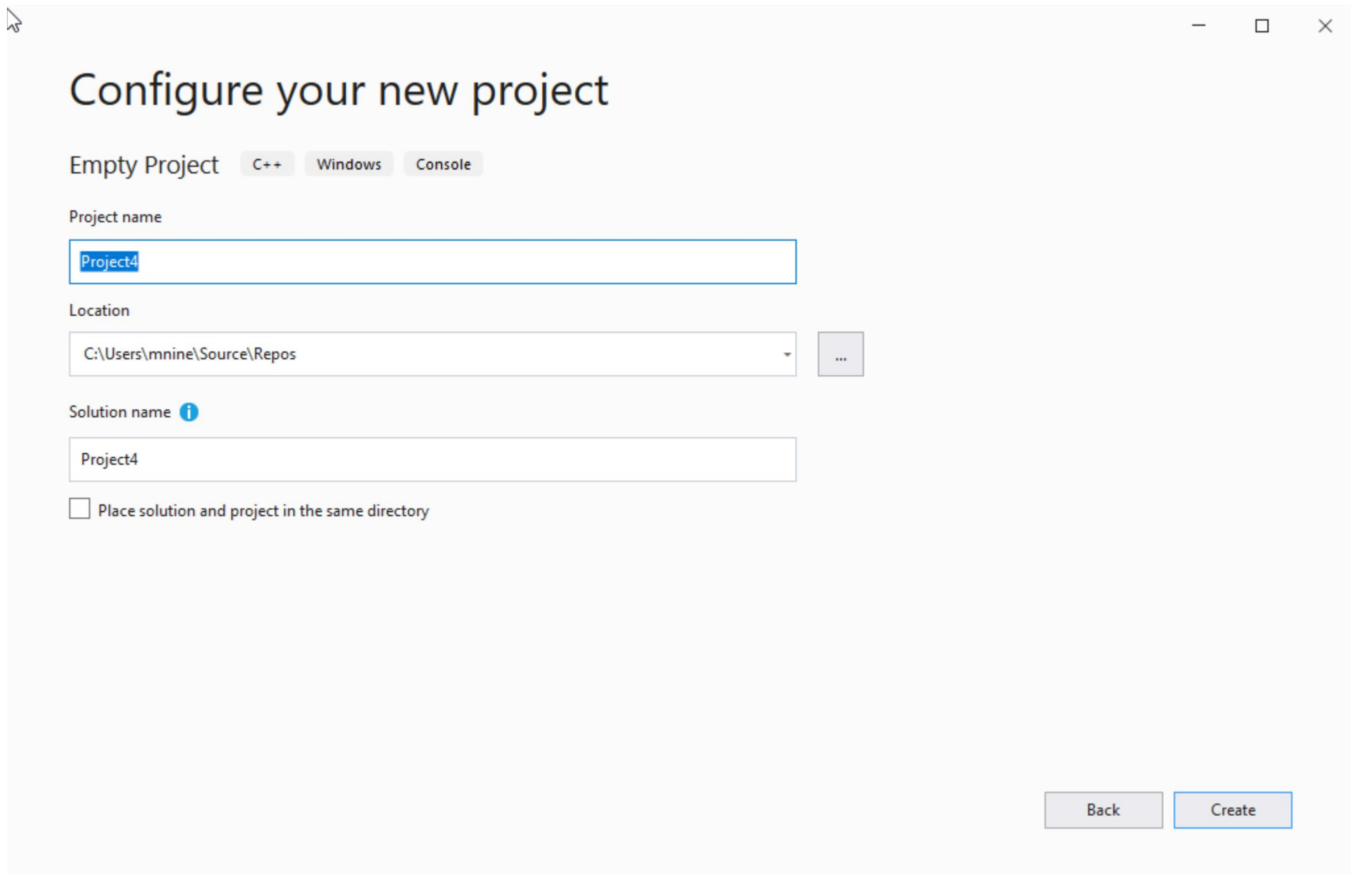
Back Next

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



Configure your new project

Empty Project C++ Windows Console

Project name

Project4

Location

C:\Users\mnine\Source\Repos

Solution name

Project4

☐ Place solution and project in the same directory

Step 1: Create a project (4)

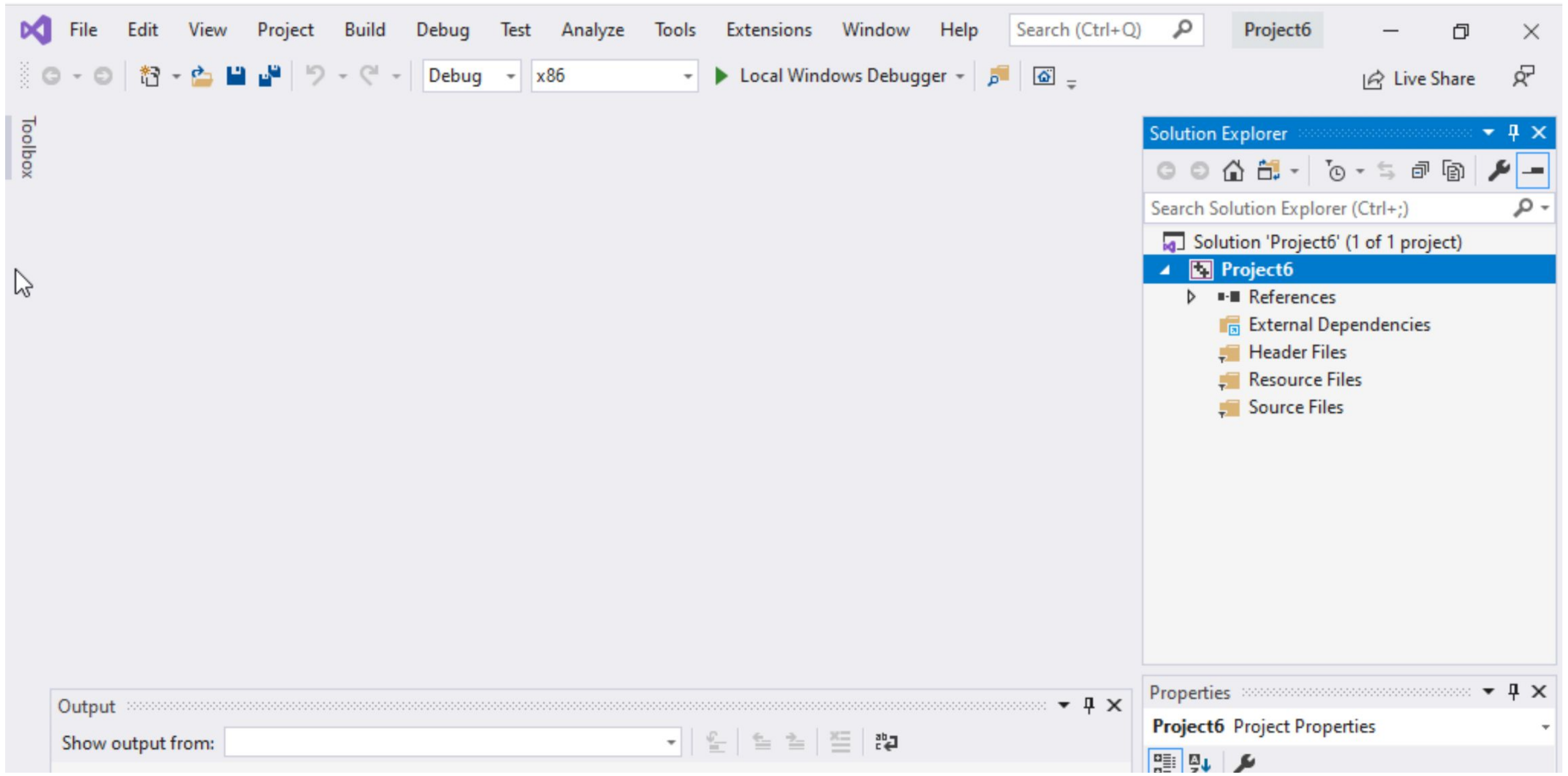
Delete the

Following folders:

Header files

Resources Files, and

Source Files



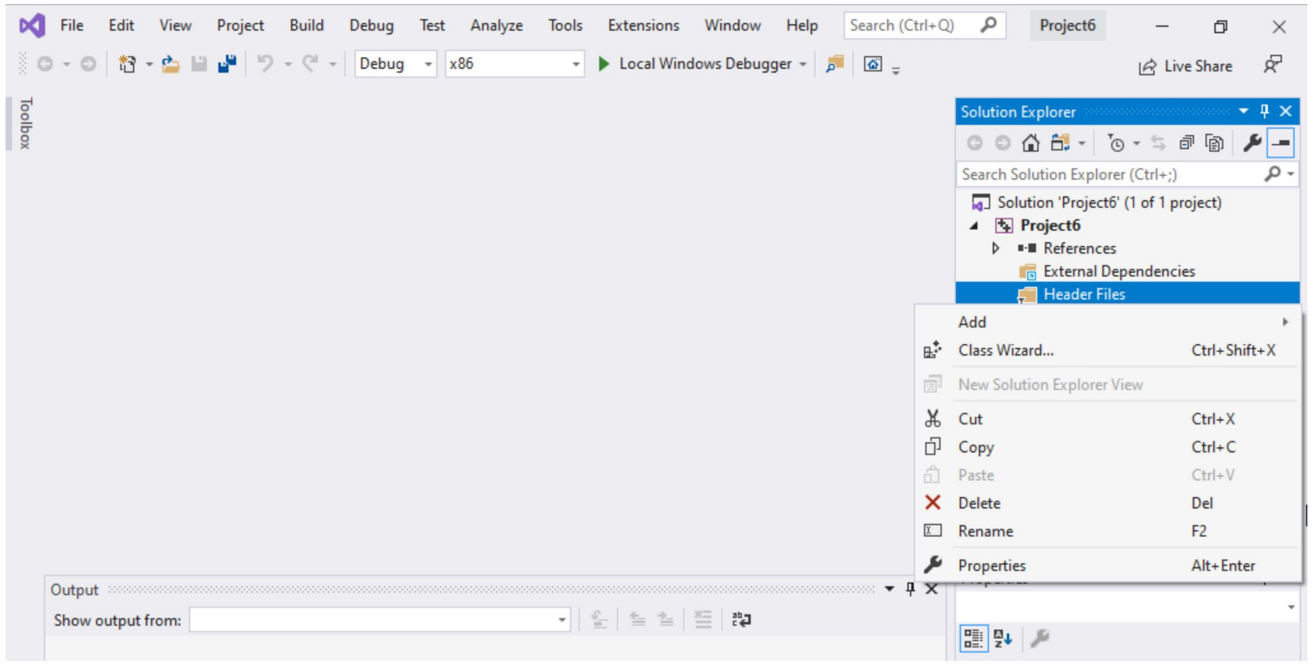
Step 1: Create a project (5)

To delete :

Select the folders

Right click on it

Select delete



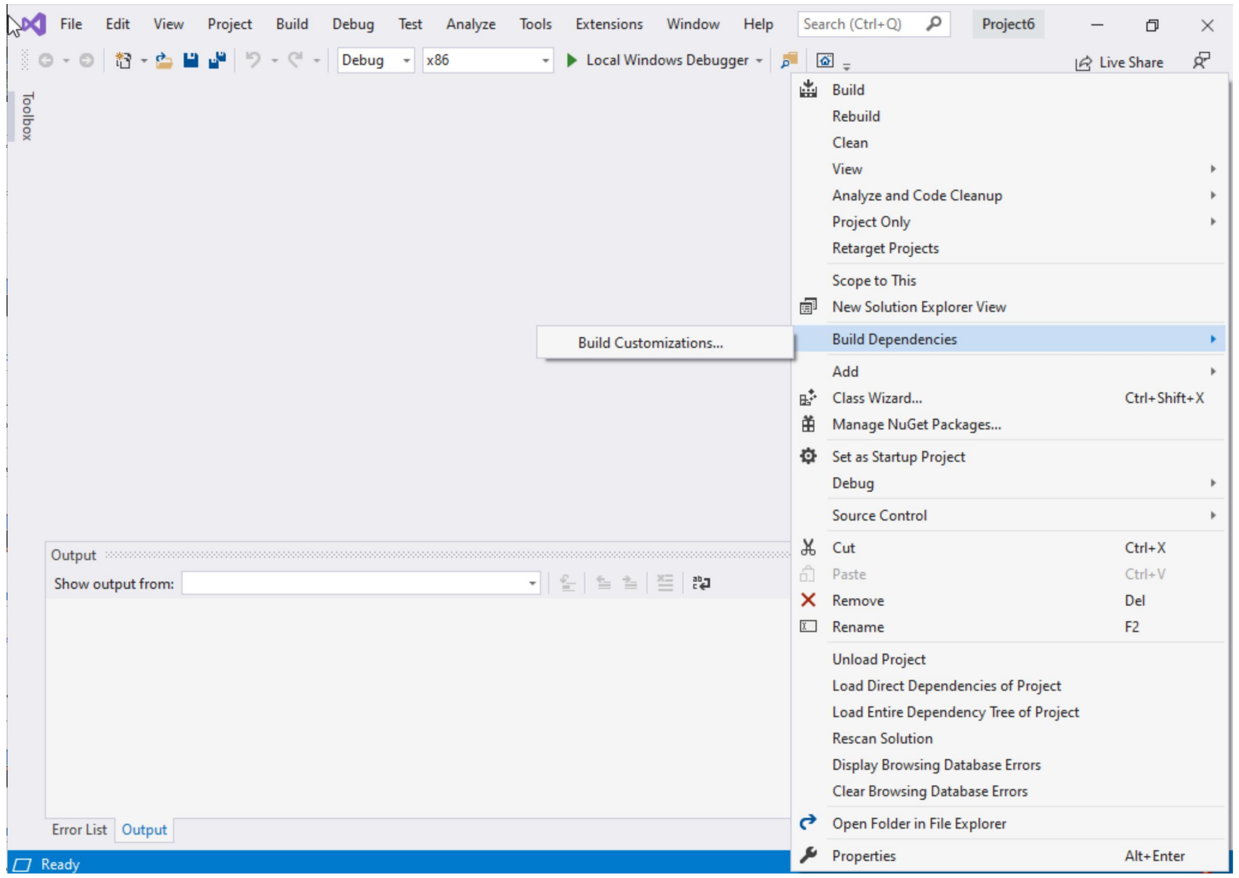
Step 1: Create a project (6)

Select Project Name on solution explorer

Right click on it

Go to Build Dependencies

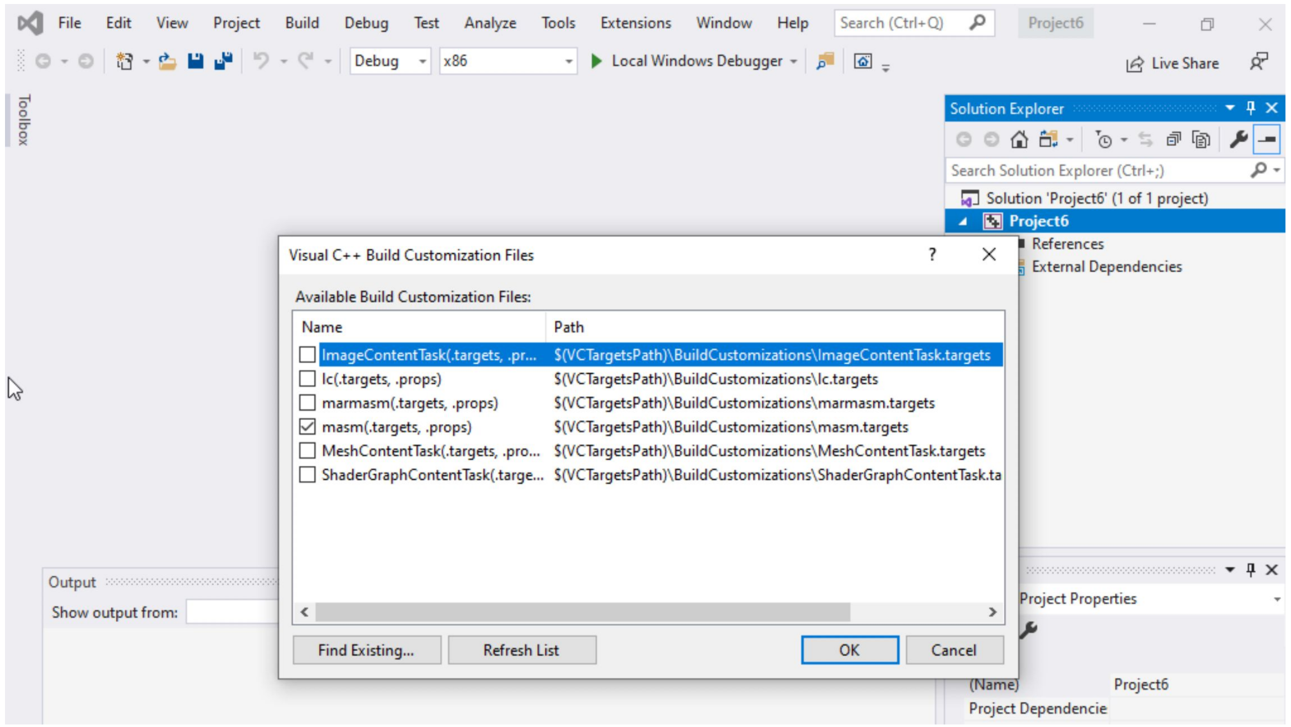
Click on Build Customizations



Step 1: Create a project (7)

Select masm(.target, .props)

Click ok


Visual C++ Build Customization Files

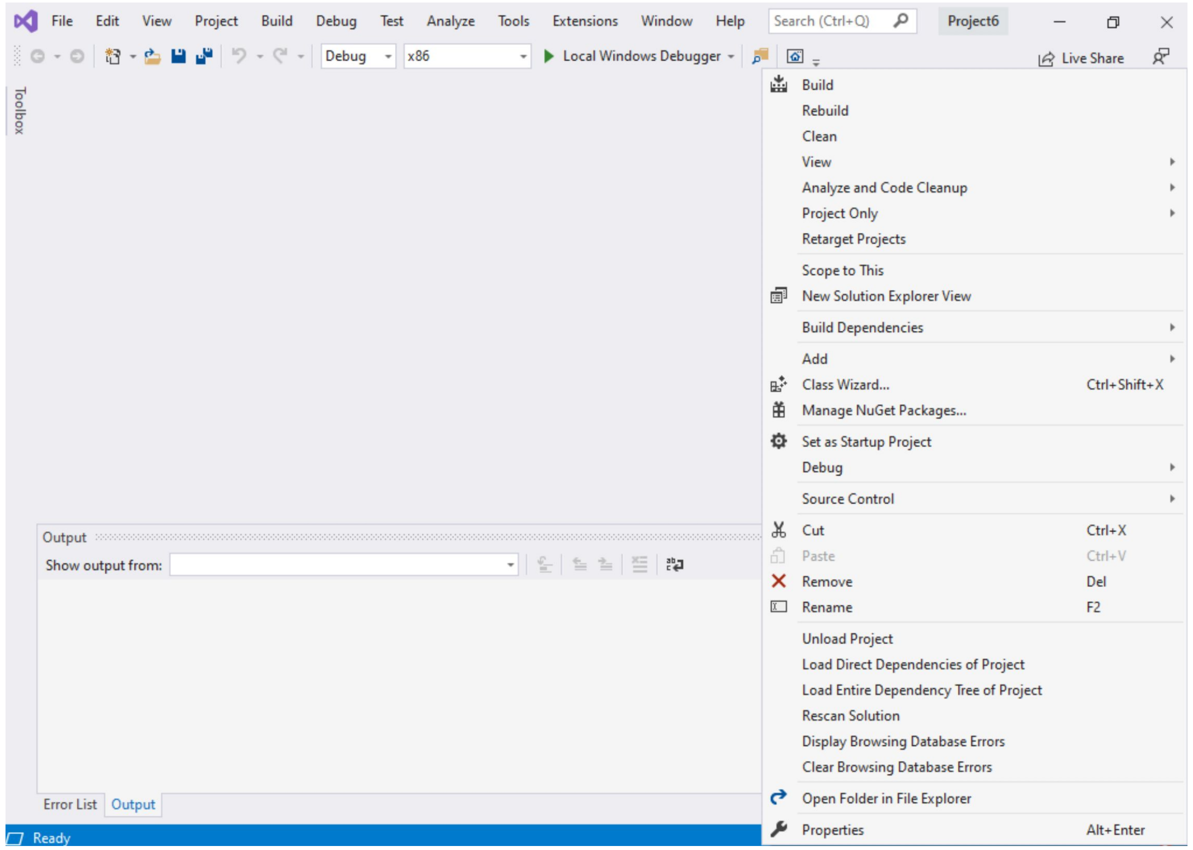
Available Build Customization Files:

	Name	Path
☐	ImageContentTask(.targets, .pr...	$(VCTargetsPath)\BuildCustomizations\ImageContentTask.targets
☐	lc(.targets, .props)	$(VCTargetsPath)\BuildCustomizations\lc.targets
☐	marmasm(.targets, .props)	$(VCTargetsPath)\BuildCustomizations\marmasm.targets
☑	masm(.targets, .props)	$(VCTargetsPath)\BuildCustomizations\masm.targets
☐	MeshContentTask(.targets, .pro...	$(VCTargetsPath)\BuildCustomizations\MeshContentTask.targets
☐	ShaderGraphContentTask(.targe...	$(VCTargetsPath)\BuildCustomizations\ShaderGraphContentTask.ta

Step 1: Create a project (8)

Right click on the Project name in the solution explorer

Click properties



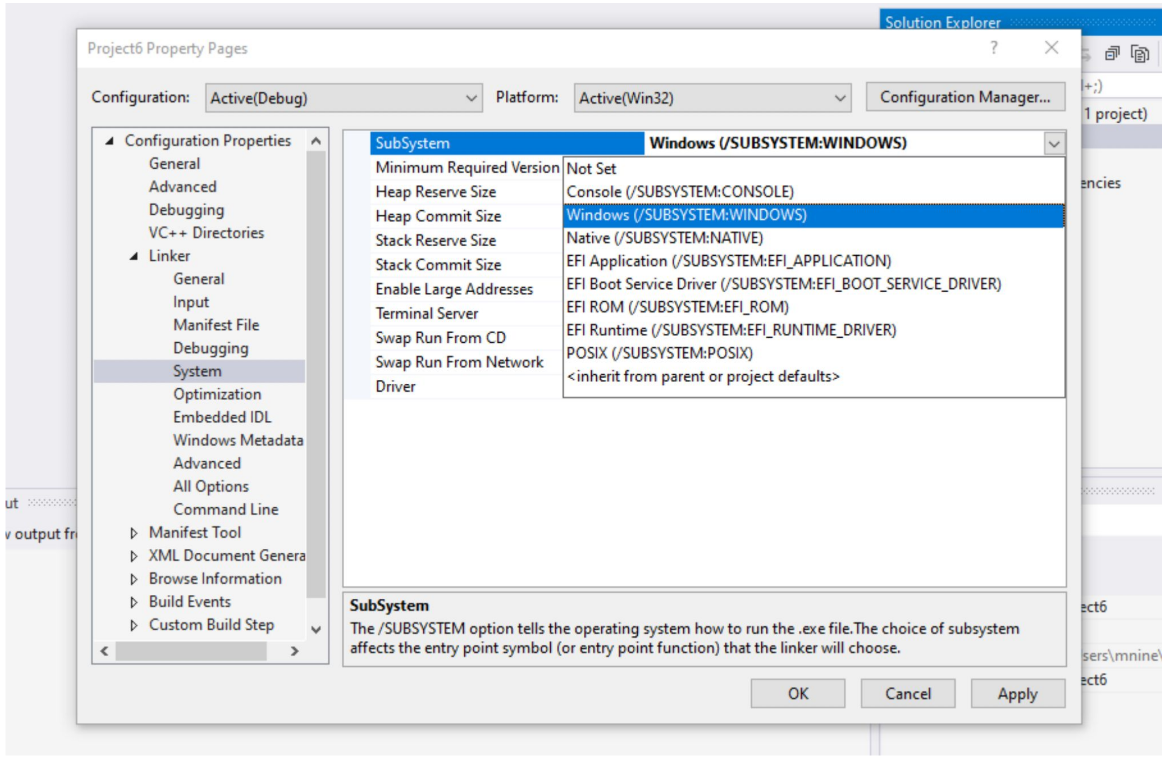
Step 1: Create a project (9)

Expand the 'Linker'

Select 'System'

Select Windows(/SUBSYSTEM:WINDOWS)

Click OK



SubSystem: Windows (/SUBSYSTEM:WINDOWS)

Not Set
Console (/SUBSYSTEM:CONSOLE)
Windows (/SUBSYSTEM:WINDOWS)
Native (/SUBSYSTEM:NATIVE)
EFI Application (/SUBSYSTEM:EFI_APPLICATION)
EFI Boot Service Driver (/SUBSYSTEM:EFI_BOOT_SERVICE_DRIVER)
EFI ROM (/SUBSYSTEM:EFI_ROM)
EFI Runtime (/SUBSYSTEM:EFI_RUNTIME_DRIVER)
POSIX (/SUBSYSTEM:POSIX)
<inherit from parent or project defaults>

SubSystem
The /SUBSYSTEM option tells the operating system how to run the .exe file.The choice of subsystem affects the entry point symbol (or entry point function) that the linker will choose.

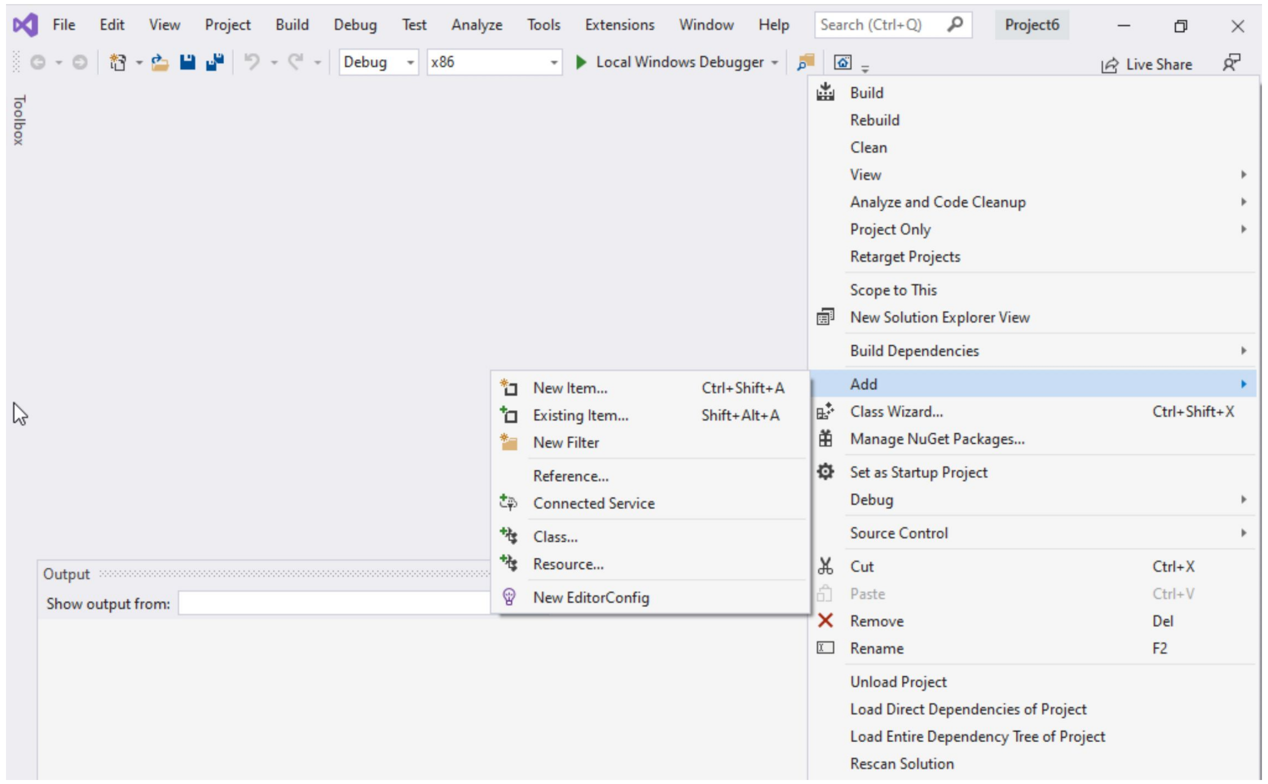
Step 1: Create a project (10)

Select Project name on solution explorer

Right click on it

Expand Add

Choose New Item

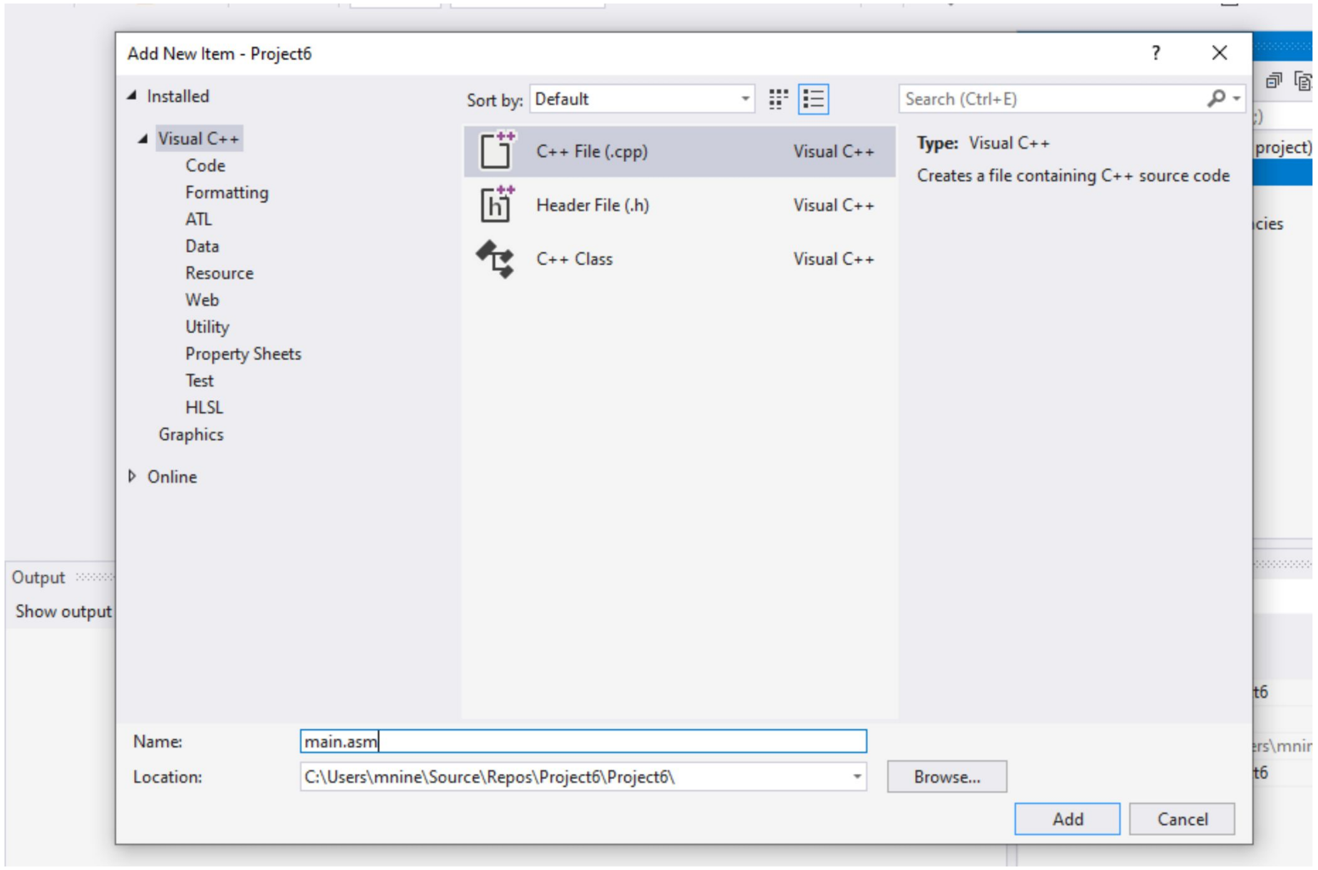


Step 1: Create a project (11)

Select C++ File(.cpp)

Name: main.asm

Click Add



Add New Item - Project6

Sort by: Default

Search (Ctrl+E)

C++ File (.cpp)	Visual C++
Header File (.h)	Visual C++
C++ Class	Visual C++

Type: Visual C++

Creates a file containing C++ source code

Name: main.asm

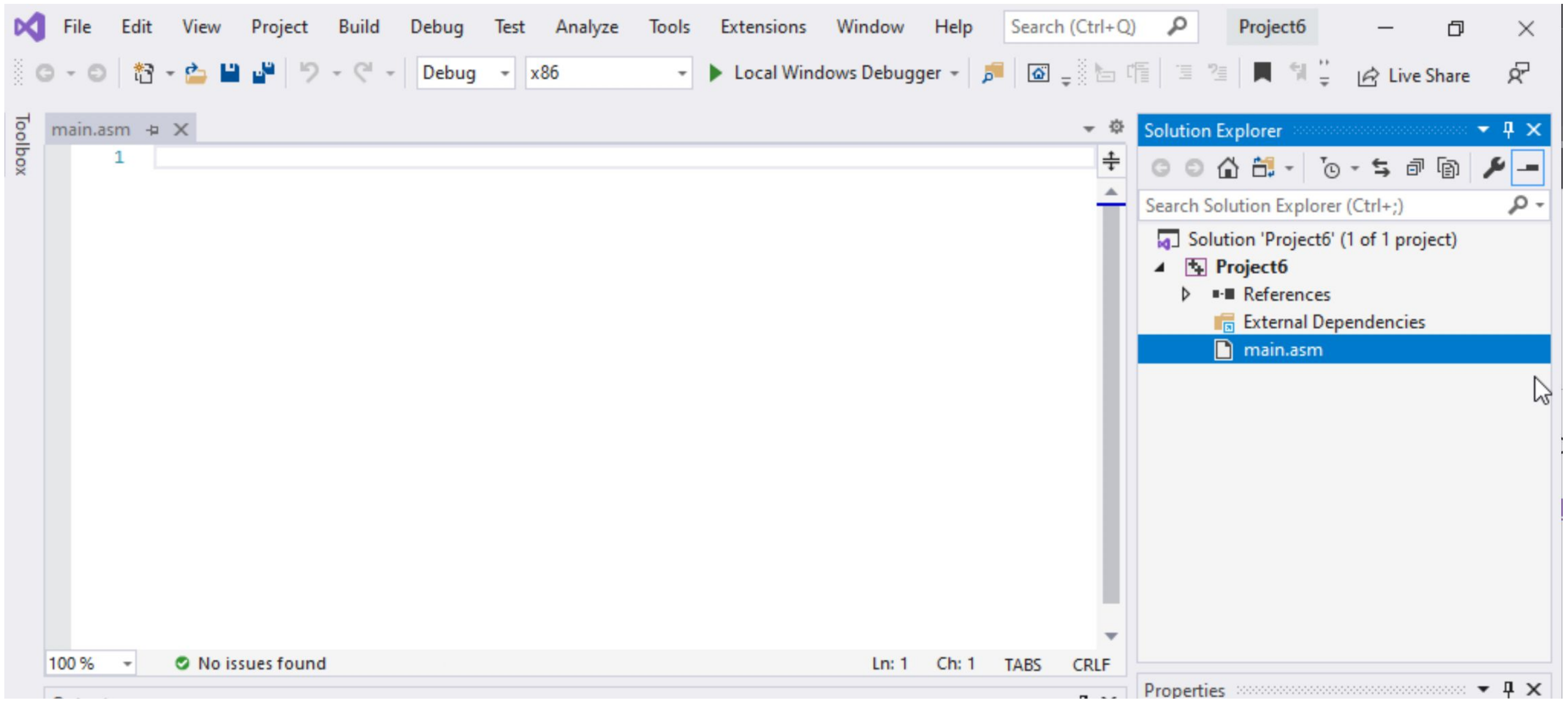
Location: C:\Users\mnine\Source\Repos\Project6\Project6\

Step 1: Create a project (12)

Select main.asm

Add your code

In the main.asm File.


File
Edit
View
Project
Build
Debug
Test
Analyze
Tools
Extensions
Window
Help
Search (Ctrl+Q)
Project6
Debug
x86
Local Windows Debugger
Live Share
Toolbox
main.asm
100 %
No issues found
Ln: 1
Ch: 1
TABS
CRLF
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Project6' (1 of 1 project)
Project6
References
External Dependencies
main.asm
Properties