**Lab Session 12**



**High Level Language Interface**



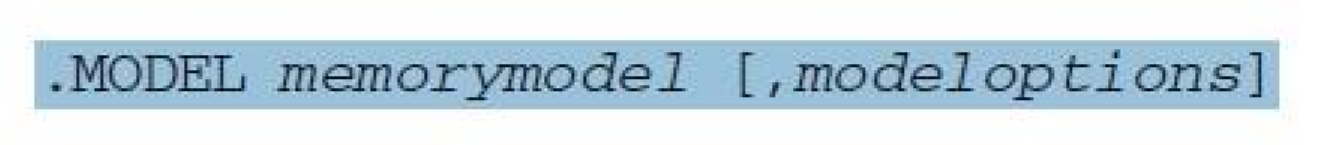
# Objectives

* General Conventions, Model Directive
* Implementing Inline Assembly Code

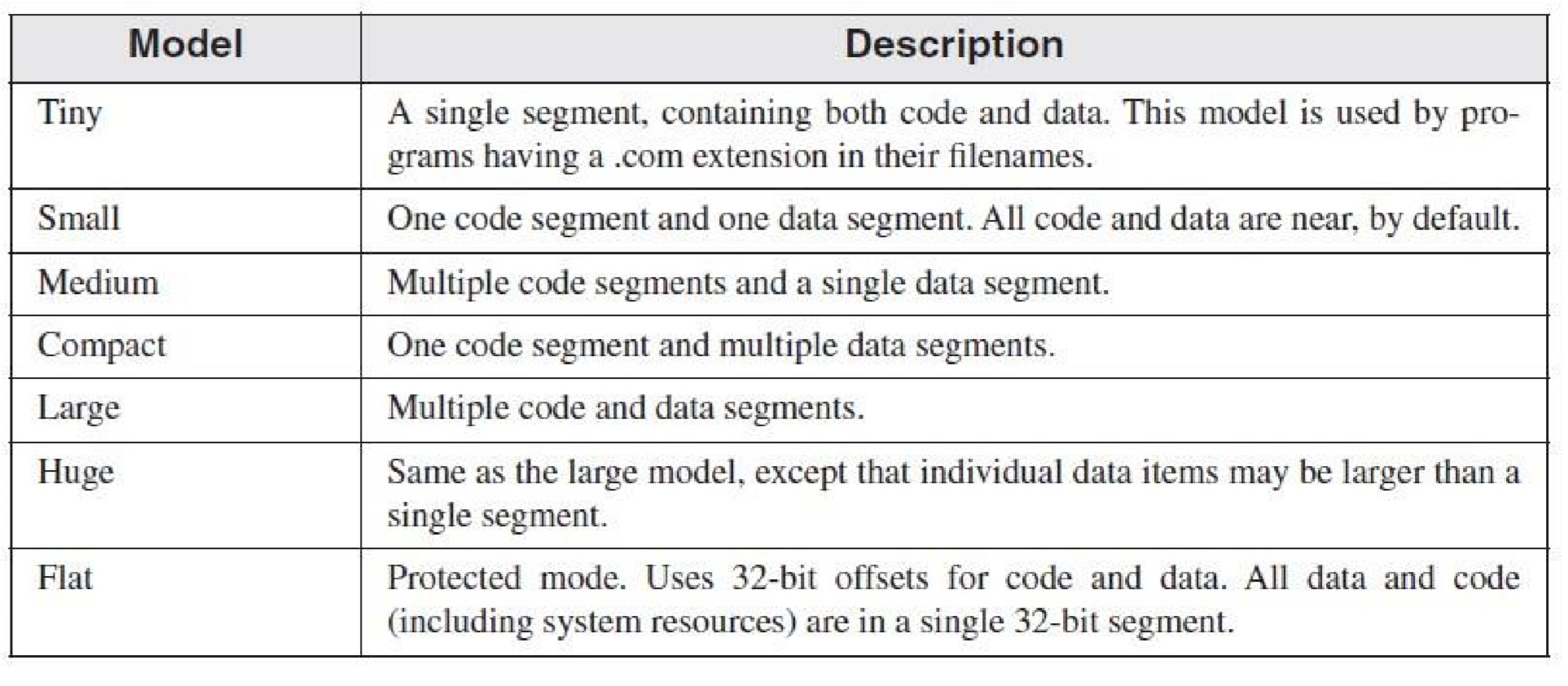
# .Model Directive

.MODEL directive determines

* memory model type
* procedure naming scheme
* parameter passing convention



**Memory Model**



**Memory Options**

Language specifier -> determines calling and naming conventions for procedures and public symbols

Stack distance -> can be NEARSTACK (the default) or FARSTACK

We mostly uses *.model flat, STDCALL*

**STDCALL** is the language specifier used when calling MS-Windows functions.

# Steps to follow

1.

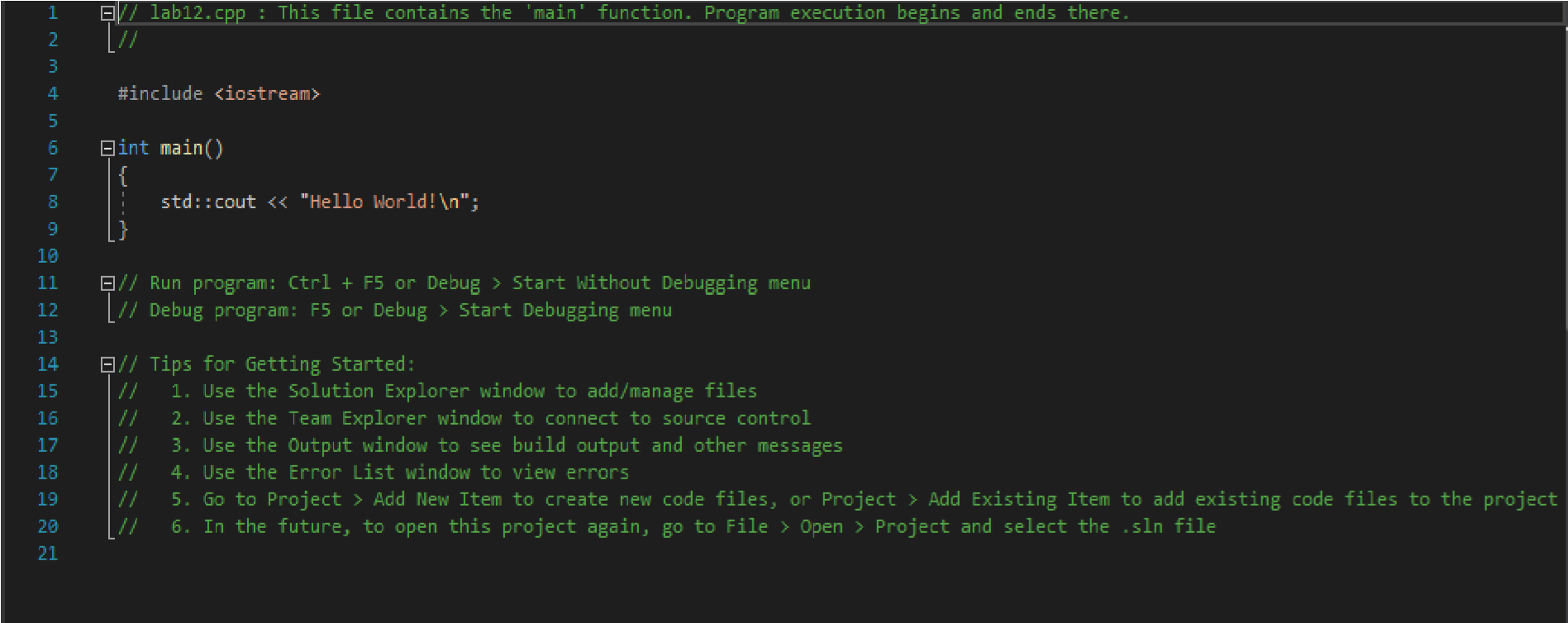
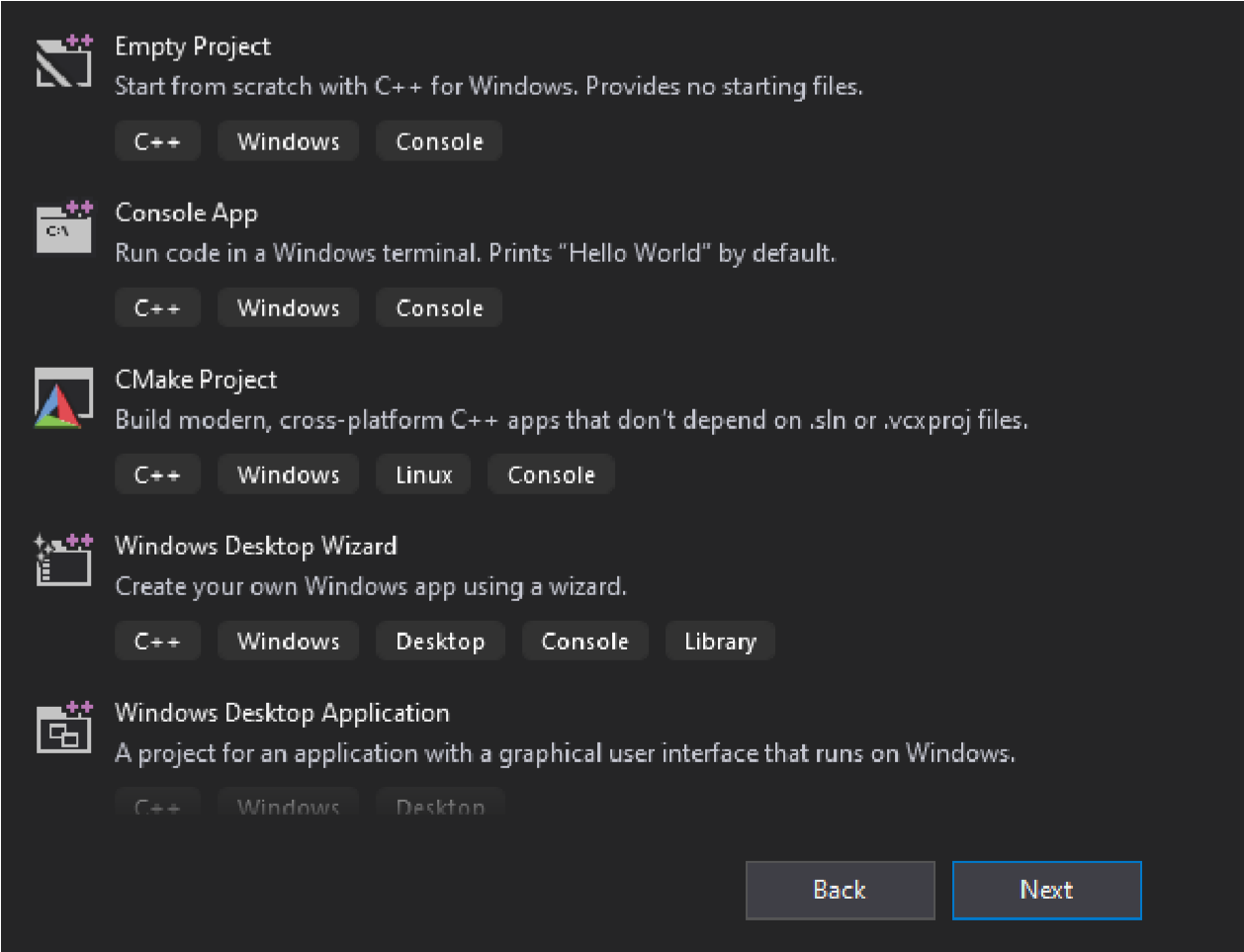
Select

**New Project >**

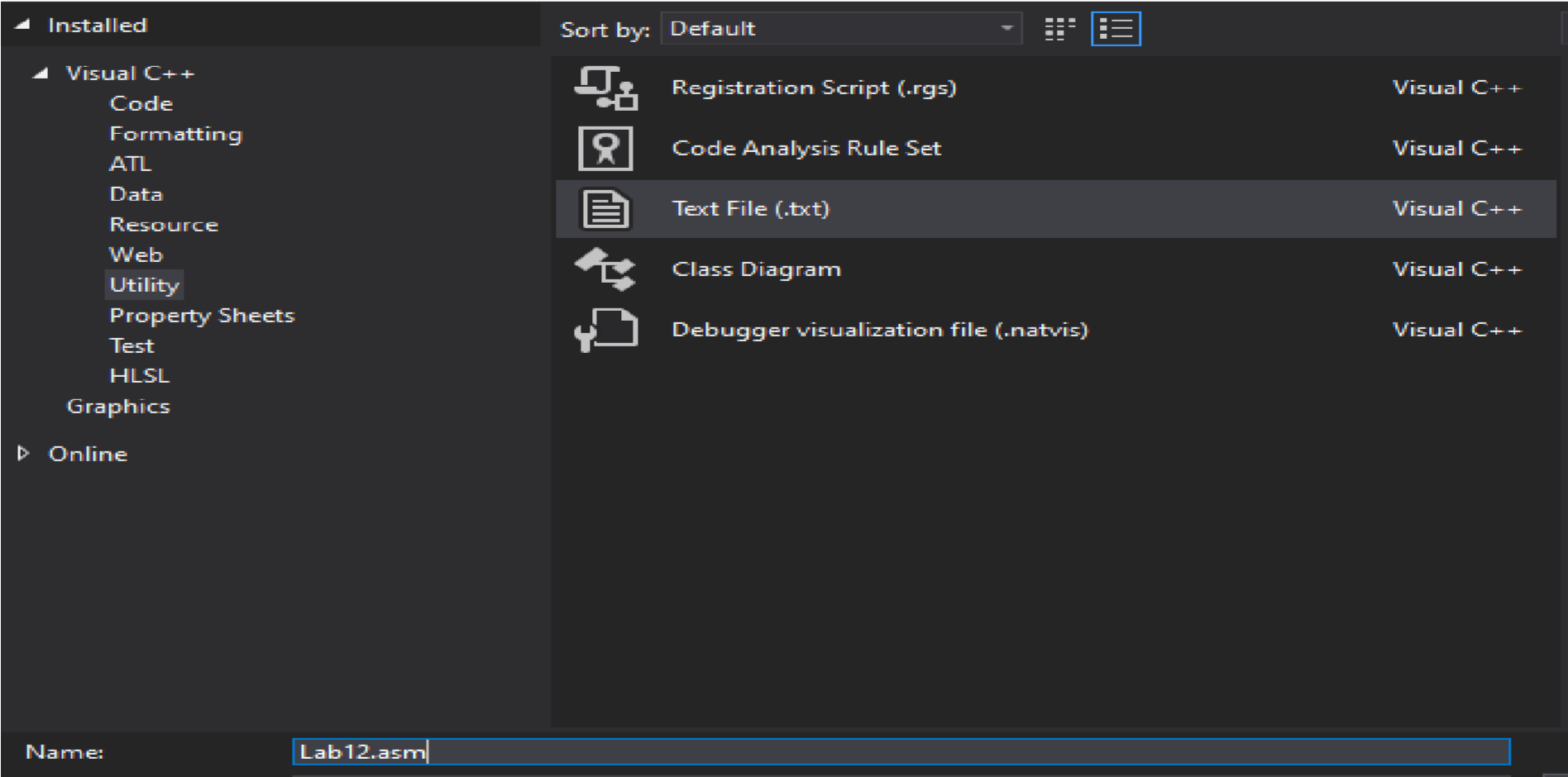
**Console Application**

2

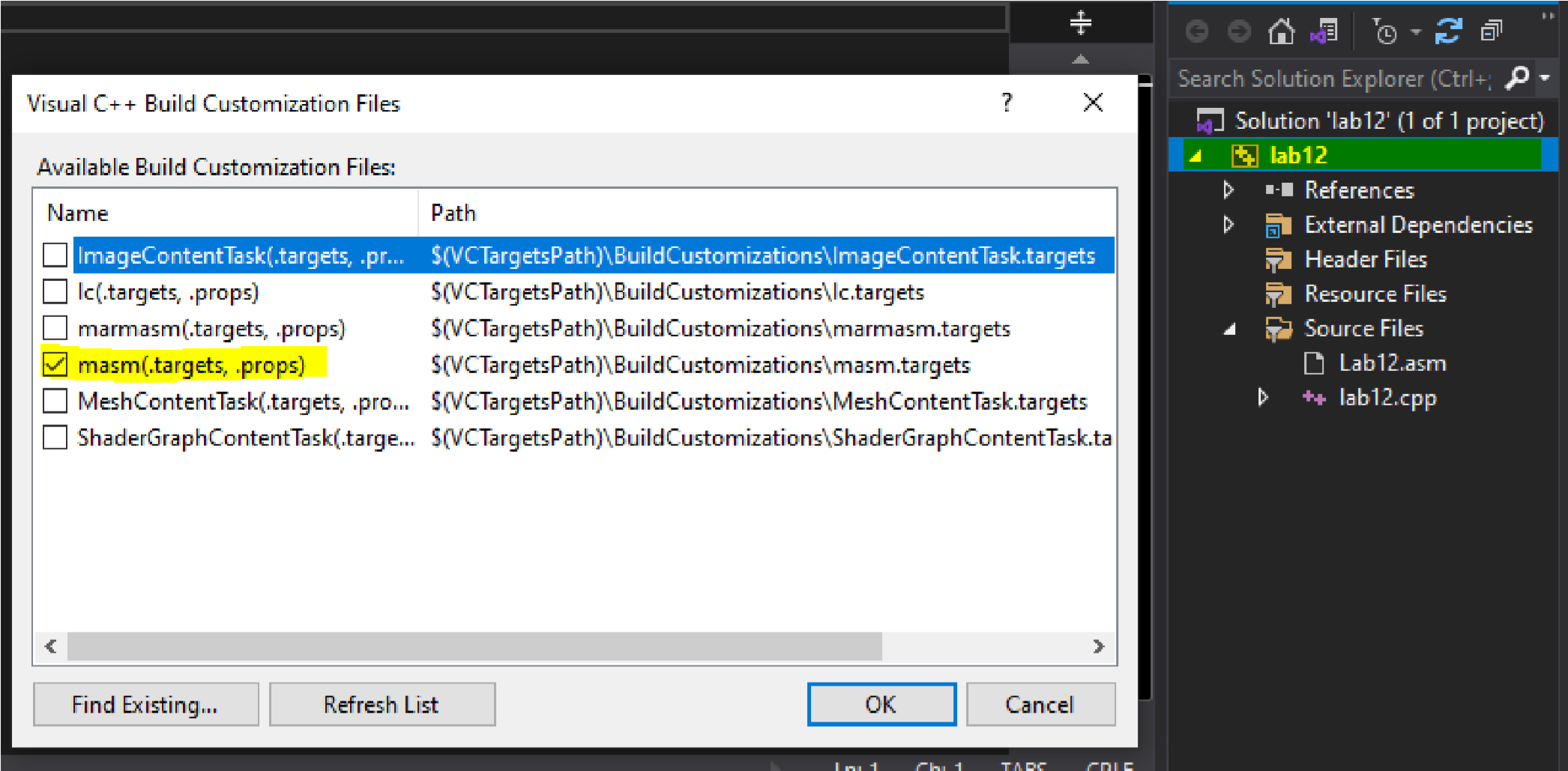
. Build the example Code.



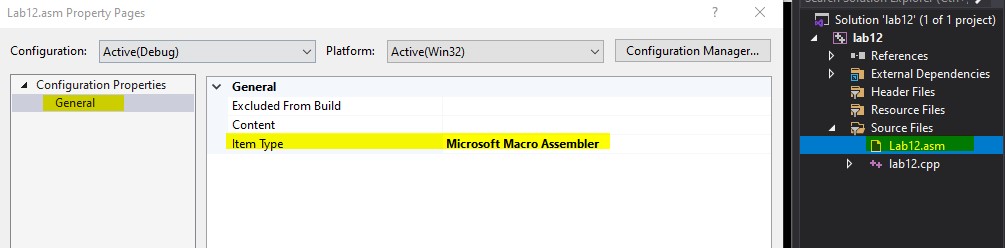
1. Add Source File . Utility >> .txt file. Change the name and extension to .asm.



1. Right Click Project >> Build Dependencies >> Build Customization >> Select MASM.



1. Right Click on .asm File.>> Properties >> General >> Item Type >> MASM.



1. Copy C++ code in .cpp file.
2. Copy Assembly Code in .asm file.
3. Build solution and observe result.

# Example:

# C++ Code:

#include <stdio.h>

// extern "C" instruct the compiler to use C calling conventions extern "C" void clear(); int main()

{

clear();

//define variables

unsigned short src\_opnd, dst\_opnd, src\_rslt, dst\_rslt; printf("Enter two 4-digit hex numbers - src, dst: \n");

scanf\_s("%hX %hX", &src\_opnd, &dst\_opnd); // in scanf\_s it is necessary to

//specifiy length

//switch to assembly

\_asm

{

MOV AX, src\_opnd

MOV BX, dst\_opnd

SHRD BX, AX, 10; shift AX : BX right 10 bits

MOV src\_rslt, AX

MOV dst\_rslt, BX

}

printf("\nSource result = %X\n Destination result = %X\n\n", src\_rslt, dst\_rslt); return 0;

}

# Assembly Code:

.686 ;Target processor. Use instructions for Pentium class machines

.MODEL FLAT, C ;Use the flat memory model. Use C calling conventions

.STACK 2048 ;Define a stack segment of 1KB (Not required for this example)

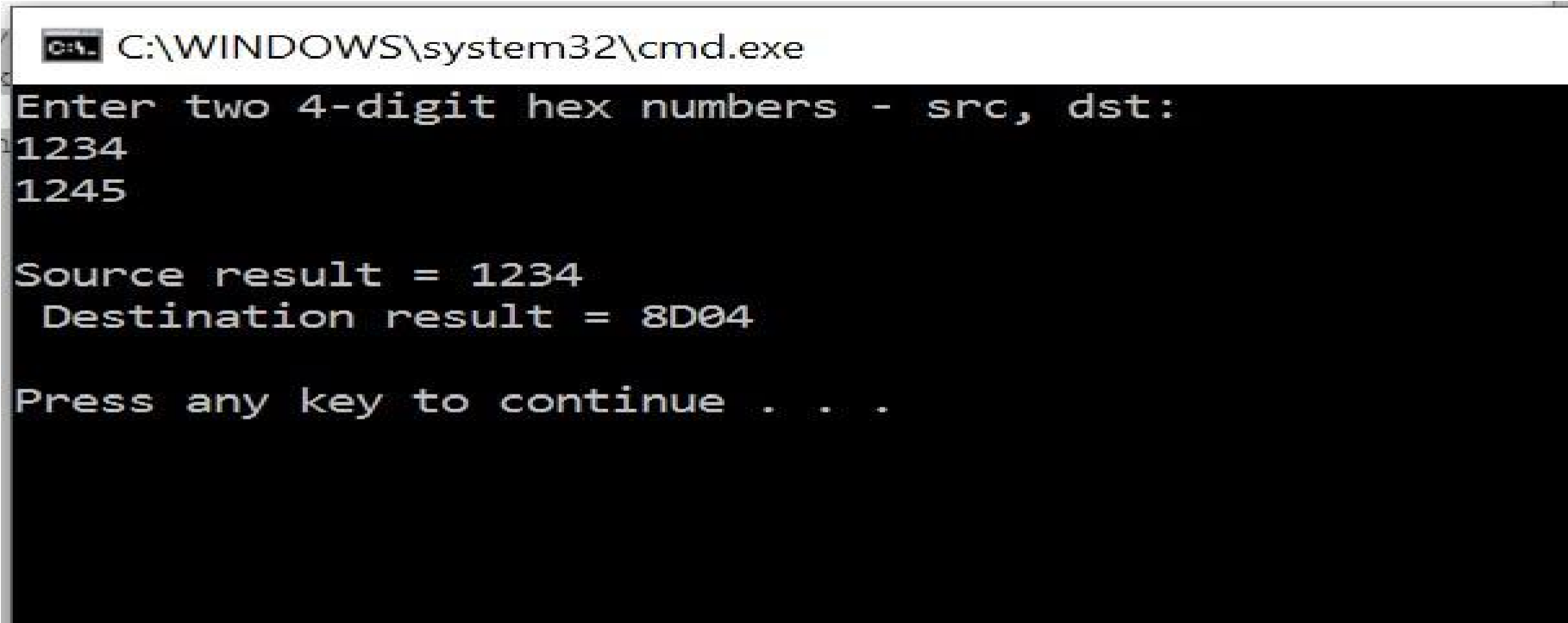
.DATA ;Create a near data segment. Local variables are declared after

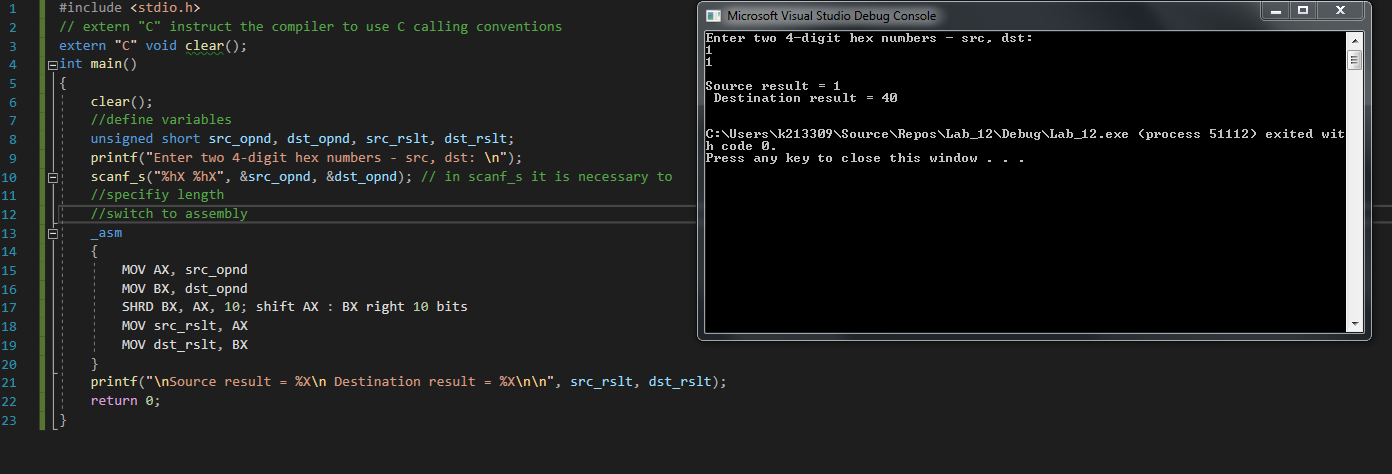
;this directive (Not required for this example) var\_1 dword 10

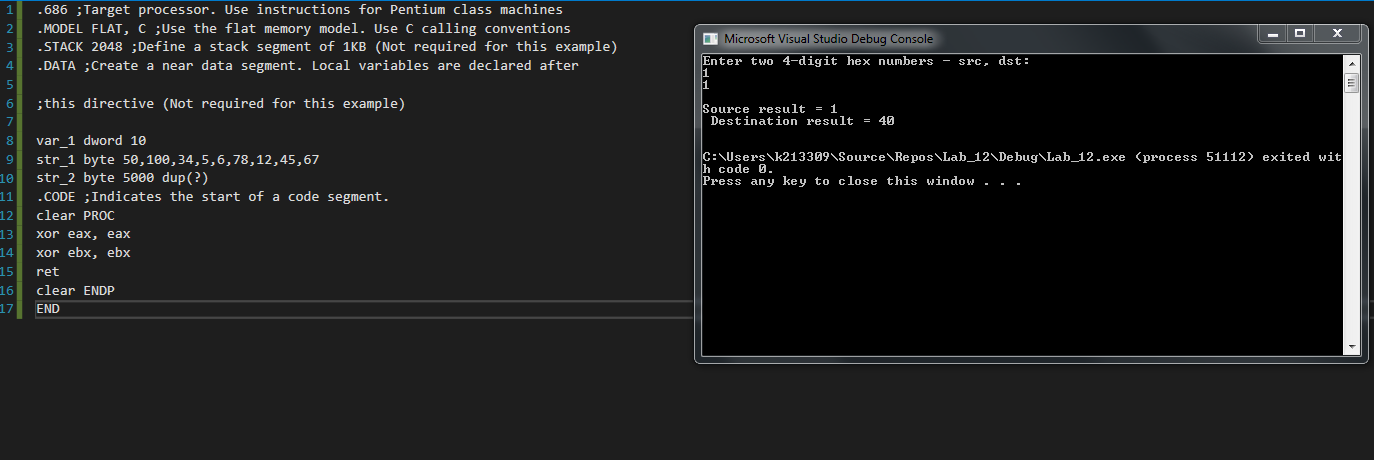
str\_1 byte 50,100,34,5,6,78,12,45,67 str\_2 byte 5000 dup(?)

.CODE ;Indicates the start of a code segment. clear PROC xor eax, eax xor ebx, ebx ret clear ENDP END

# Output:

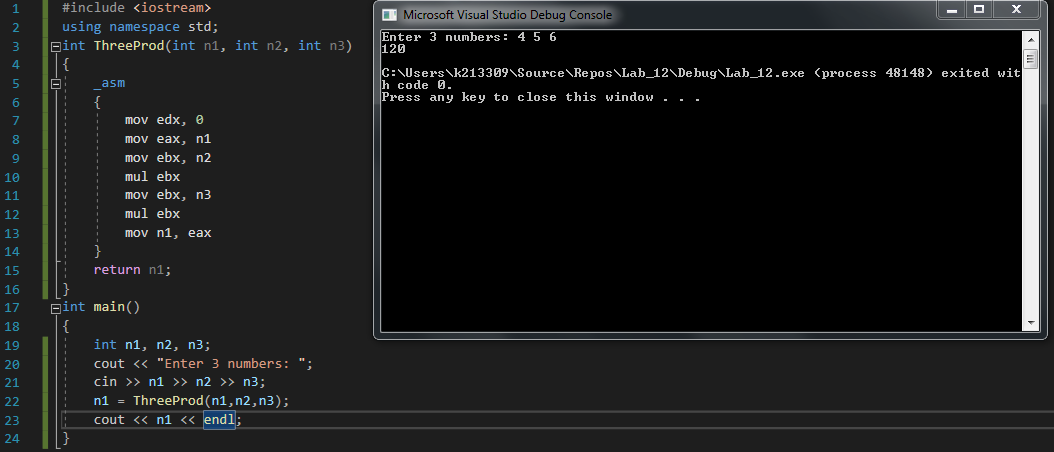




****

**ACTIVITIES:**

1. Write a program in C++ which takes input from user and contains a procedure in assembly named **ThreeProd** that displays the product of three numeric parameters passed in a function argument.



1. Write a program in C++ which takes input from user and contains a procedure in assembly named **GCD(**Greatest common divisor)which calculates their GCD.

