# **CS 410 C++ to Assembly Activity**

**Step 1:** Explain the functionality of the C++ code.

## C++ Code Functionality

| **C++ Line of Code** | **Explanation of Functionality** |
| --- | --- |
| #include<iostream> | A standard library that provides input and output services (cout & cin) |
| using namespace std; | Declares that the program is using the std namespace which is a collection of standard functions. |
| Int main(){} | The main function of the program that serves as the entry point for execution. (MUST HAVE) |
| Int width = 10; | Declares a variable called width and stores the value 10 within. |
| Int height = 5; | Declares a variable called height and stores the value 5 within. |
| Int area; | Declares a variable called area. |
| area = width \* height; | Takes the value stored in width and the value stored in height, Multiplies the two together and stores the result in the variable area. |
| cout << endl << area | Uses the standard library to call a function called cout which prints the following values to the console << first a new line is printed to the console << then the value that is stored in area. |
| return 0; | Returns 0 to the console which stops/closes the program. Signifies to the operating system that the program finished without errors. |
|  |  |
|  |  |

**Step 2:** Convert the C++ file into assembly code.

**Step 3:** Align each line of C++ code with the corresponding blocks of assembly code.

## C++ to Assembly Alignment

| **C++ Line of Code** | **Blocks of Assembly Code** |
| --- | --- |
| Int main() | Main:  Pushq %rbp  Movq &rsp, %rbp |
|  |  |
| Int width = 10; | Movl $10, -12(%rbp) |
|  |  |
| Int height = 5; | Movl $5, -8(%rbp) |
|  |  |
| Int area | movl %eax, -4(%rbp) (done after the multiplication) |
|  |  |
| Area = width \* height | movl -12(%rbp), %eax  imull  -8(%rbp), %eax |
|  |  |
| cout << endl << area | movq %rax, %rsi   leaq  \_ZSt4cout(%rip), %rdi  movq %rax, %rdx   movl  -4(%rbp), %eax   movl  %eax, %esi |
|  |  |
| Return 0; | movl $0, %eax  leave  ret |

**Step 4:** Explain how the blocks of assembly code perform the same tasks as the C++ code.

## Assembly Functionality

| **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- |
| Main: | Pushq %rbp  Movq &rsp, %rbp  These two lines of assembly set up the stack for the program a.k.a the start |
|  |  |
| Movl $10, -12(%rbp) | Moves the value 10 into the stack 12 bytes above the registered base pointer. Must allocate for all variables being used (3 in this case each worth 4 bytes) |
|  |  |
| Movl $5, -8(%rbp) | Moves the value 5 into the stack 8 bytes above the registered base pointer. |
|  |  |
| Movl %eax, -4(%rbp) | Moves the value that is stored in the register EAX (extended accumulator) into the stack at 4 bytes above the registered base pointer (This is done after the multiplication has been addressed and so the area value is stored into %eax) |
|  |  |
| movl -12(%rbp), %eax  imull  -8(%rbp), %eax | Moves the value stored at 12 bytes above the rbp into the %eax register,  Then multiplies the value stored at 8 bytes above the rbp with that value that is stored in the %eax register (aka 5 \* 10) and then stores the result in %eax |
|  |  |
| movq %rax, %rsi   leaq  \_ZSt4cout(%rip), %rdi  movq %rax, %rdx   movl  -4(%rbp), %eax   movl  %eax, %esi | movq %rax, %rsi: This moves the value in rax (likely the address of area or the output) to rsi, which is used for function calls.  leaq \_Zst4cout(%rip), %rdi: This loads the effective address of cout into the rdi register.  movq %rax, %rdx: Moves the value in rax to rdx.  movl -4(%rbp), %eax: Loads the area value into the eax register.  movl %eax, %esi: This moves the value from eax into esi, which is the register that gets printed. |
|  |  |
| movl $0, %eax  leave  ret | Moves the return value 0 into the eax register, which is the return value for main().  Leave cleans up the stack frame.  Ret, returns from the function, marking the end of the program. |