# **CS 410 C++ to Assembly Activity – Elliot Putnam**

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

## C++ Code Functionality

| **C++ Line of Code** | **Explanation of Functionality** |
| --- | --- |
| #include<iostream> | Calls to include input output stream functions into the program that allows the use of peripherals |
| using namespace std; | Calls to use the standard library with many functions within it. So by calling this standard, one does not need to type std::cout, and just types cout. Careful as not to write duplicate functions of the standard namespace. |
| int main() | Calls the main function which is the main loop of the program |
| int width=10; | Creates an integer named width, with a value of 10. |
| int height=5; | Creates an integer named height, with a value of 5 |
| int area; | Creates an integer named area, without any value assigned to it |
| area = width \* height; | Fills area with the sum of width and height |
| cout<<endl<< area; | Console out (cout) means the following can be written to the console, endl ends the line, starting a new one, and the value of area is written to the console. |
| return 0; | Return 0 tells the main function loop that is has ended, 0 being off, 1 being on. |
|  |  |
|  |  |

**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() | | .globl main | | --- | | .type main, @function | | .LFB1493 | | .cfi\_startproc | | pushq %rbp | | .cfi\_def\_cfa\_offset 16 | | .cfi\_offset 6, -16 | | movq %rsp, %rbp | | .cfi\_def\_cfa\_register 6 | | subq $16, %rsp | | movl $10, -12(%rbp) | | movl $5, -8(%rbp) | |
| Int width = 10; | movl $10, -12(%rbp) |
| Int height = 5; | movl $5, -8(%rbp) |
| Int area; | Defined in the multiplication below |
| Area = width \* height; | | movl -12(%rbp), %eax | | --- | | imull -8(%rbp), %eax | | movl %eax, -4(%rbp) | | movq \_ZSt4endl...%rax | | movq %rax, %rsi | | leaq \_ZSt4cout(%rip), %rdi | | call \_ZNSolsEPFRSoS\_E@PLT | | movq %rax, %rdx | | movl -4(%rbp), %eax | |
| cout << endl << area; | movq \_ZSt4endl...%rax |
|  | movq %rax, %rsi |
|  | leaq \_ZSt4cout(%rip), %rdi |
|  | call \_ZNSolsEPFRSoS\_E@PLT |
|  | movq %rax, %rdx |

|  | movl -4(%rbp), %eax |
| --- | --- |
|  | movl %eax, %esi |
|  | movq %rdx, %rdi |
|  | call \_ZNSolsEi@PLT |
|  | movl -4(%rbp), %eax |
|  | movq %rdx, %rdi |
|  | call \_ZNSolsEi@PLT |
| 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** |
| --- | --- |
| .file “assignment1\_1.cpp” | Marks the start of the file with name of the source file |
| .text | Begins a section of executable code instructions will be |
| .section .rodata | Marks a section and that the data will be read only .rodata |
| .type \_ZStL19piecewise\_construct, @object | Defines the type, specifically declares “\_ZStL19piecewise\_construct” to be an object |
| .size \_ZStL19piecewise\_construct, 1 | Defines the size using and (object, size in bytes) |
| \_ZStL19piecewise\_construct: | Defines the start of the object that was declared |
| .zero 1 | Allocates a single byte at zero (since it’s a 1 byte object) |
| .local \_ZStL8\_ioinit | Declares a local symbol |
| .comm \_ZStL8\_ioinit, 1, 1 | Declares the symbol to be a common block with the size of 1 byte, and alignment of 1 byte |
| .text | Ends the section of executable code instructions |
| .globl main | Declares main as a global symbol |
| .type main, @function | Declares main as a function |
| .LFB1493 | An internal label used for debugging |
| .cfi\_startproc | Start of the call frame information |
| Pushq %rbp | Pushes base pointer on to the stack |
| .cfi\_def\_cfa\_offset 16 | Defines the canonical frame address (CFA) offset |
| .cfi\_offset 6, -16 | CFA offset for the %rbp |
| Movq %rsp, %rbp | Moves the stack pointer to the base pointer to set up the new stack frame |
| .cfi\_def\_cfa\_register 6 | Sets CFA to the new base pointer |
| subq $16, %rsp | Allocates 16 bytes on the stack for local variables |
| movl $10, -12(%rbp) | Moves the integer 10 into the local variable at an offset -12 |
| Movl $5, -8(%rbp) | Moves the integer 5 into the local varable at an offset -8 |
| movl -12(%rbp), %eax | Loads the value at offset -12 into the eax register |
| imull -8(%rbp), %eax | Multiplies eax by the value at -8 (10,5 = 50) |
| movl %eax, -4(%rbp) | Stores eax into the local variable at offset -4 |
| movq \_ZSt4endl...%rax | Loads addresses for the cout and endl std functions |
| movq %rax, %rsi |  |
| leaq \_ZSt4cout(%rip), %rdi |  |
| call \_ZNSolsEPFRSoS\_E@PLT | Calls the << function for streams |
| movq %rax, %rdx | Moves the stack pointer to the base pointer |
| movl -4(%rbp), %eax | Loads the value at offset -4 to the eax register |
| movl %eax, %esi | Moves pointer |
| movq %rdx, %rdi | Moves pointer |
| call \_ZNSolsEi@PLT | Calls the stream output function for integers |
| Movl $0, %eax | Sets the return value to zero |
| Leave | Cleans up stack frame |
| Ret | Returns from the function |
| .cfi\_endproc | End of call frame information |
| .LFE1493 | Label for debugging |
| .size main, .-main | Calculates the size of the main function |
| .type  \_Z41\_\_static\_initialization\_and\_destruction\_0ii, @function | Declares the function type for the listed function |
| \_Z41\_\_static\_initialization\_and\_destruction\_0ii: | Label that marks the start of the function |
| .LFB1979 | Label for debugging |
| .cfi\_startproc | Start of the call frame |
| Pushq %rbp | Saves old base pointer on the stack |
| .cfi\_def\_cfa\_offset 16 | Defines CFA offset by 16 |
| .cfi\_offset 6, -16 | Records the offset of the saved register |
| movq %rsp, %rbp | Setup new base pointer for the stack frame |
| .cfi\_def\_cfa\_register 6 | Updates the CFA to use %rbp |
| subq $16, %rsp | Allocates 16 bytes on the stack for local variables |
| movl %edi, -4(%rbp) | Stores the first parameter into the local variable at -4 offset of rbp |
| movl %esi, -8(%rbp) | Stores the second parameter into the local variable |
| cmpl $1, -4(%rbp) | Compares the local variable with 1 |
| jne .L5 | If not equal, jump to label .L5 |
| cmpl $65535, -8(%rbp) | Compares the local variable with 65535 |
| jne .L5 | If not equal, jump to label .L5 |
| leaq \_ZStL8\_\_ioinit(%rip), %rdi | Loads the address listed into rdi |
| call \_ZNSt8ios\_base4InitC1Ev@PLT | Calls the constructor for ios\_base::Init |
| ios\_base::Init | Initializes the IO system for cpp std |
| leaq \_\_dso\_handle(%rip), %rdx | Loads the address into rdx |
| leaq \_ZStL8\_\_ioinit(%rip), %rsi | Loads the address into rsi |
| movq \_ZNSt8ios\_base4InitD1Ev@GOTPCREL(%rip), %rax | Loads the address of the ios\_base::Init destructor into rax |
| movq %rax, %rdi | Moves the destructors address from rax to rdi |
| call \_\_cxa\_atexit@PLT | Registers the destructor to be called at the program exit |
| .L5 | Label the end of conditionally executed code |
| nop | No operation |
| leave | Restores old base pointer and stack frame |
| .cfi\_def\_cfa 7, 8 | Updates CFA to reflect the leave instruction |
| ret | Returns from the function |
| . cfi\_endproc | End of call frame info |
| . LFE1979 | Label for the end of the function |
| .size \_Z41\_\_static\_initialization\_and\_destruction\_0ii, .-\_Z41\_\_static\_initialization\_and\_destruction\_0ii | Calculates the size of the function |
| .type \_GLOBAL\_\_sub\_I\_main, @function | Declares the function |
| \_GLOBAL\_\_sub\_I\_main: | Label that marks the start of the function |
| .LFB1980: | Label for debugging |
| .cfi\_startproc | Start the CFI |
| pushq %rbp | Saves base pointer |
| .cfi\_def\_cfa\_offset 16 | Defines the CFA offset |
| .cfi\_offset 6, -16 | Records the offset of the saved rbp |
| movq %rsp, %rbp | Setup of a new base pointer |
| .cfi\_def\_cfa\_register 6 | Updates CFA to rbp |
| movl $65535, %esi | Moves 65535 into the register esi |
| movl $1, %edi | Moves 1 into the register edi |
| call \_Z41\_\_static\_initialization\_and\_destruction\_0ii | Calls the static init function |
| popq %rbp | Restores the base pointer |
| .cfi\_def\_cfa 7, 8 | Updates CFA after restoring rbp |
| ret | Returns from the function |
| .cfi\_endproc | End of frame call info |
| .LFE1980: | Label for the end of the function |
| .size \_GLOBAL\_\_sub\_I\_main, .-\_GLOBAL\_\_sub\_I\_main | Calculates the size of the function |
| .section .init\_array,"aw" | Defines a section for an array of function pointers to global constructors |
| .align 8 | Aligns the following data to 8 byte boundary |
| .quad \_GLOBAL\_\_sub\_I\_main | Adds \_GLOBAL\_\_sub\_I\_main to the array |
| .hidden \_\_dso\_handle | Marks \_\_dso\_handle as hidden |
| .ident "GCC: (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0" | Identifies the GCC compiler version that compiled the code |
| .section .note.GNU-stack,"",@progbits | Marks the section attributes for the stack, also indicates that no executable stack is required |