# **CS 410 C++ to Assembly With Loops Activity Template**

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

## C++ Code Functionality

| **C++ Line of Code** | **Explanation of Functionality** |
| --- | --- |
| #include<iostream> | Includes iostream |
| Using namespace std; | Used for the standard namespace |
| Int main() | Creates a function that returns an integer |
| Int num, I;  Int product = 1; | Declares the variables that will be used in the program |
| Cout<< “Enter a number:\n”<<endl; | Prints out “Enter a number:” for user to enter their desired number |
| Cin>>num; | Gets the input of the user |
| For(i=num; i>0; i--)  Product = product \* I; | Makes a loop until I is no longer greater than 0  Makes the multiplication of “product” and “I” to the product variable |
| Cout<<”The factorial for” << num<< “is: \n” << product; | Prints out the factorial |
| Return 1; | Exits the program |

**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 num, I;  Int product = 1; | .LFB1493:    .cfi\_startproc    pushq %rbp    .cfi\_def\_cfa\_offset 16    .cfi\_offset 6, -16    movq  %rsp, %rbp    .cfi\_def\_cfa\_register 6    subq  $32, %rsp    movq  %fs:40, %rax    movq  %rax, -8(%rbp)    xorl  %eax, %eax    movl  $1, -12(%rbp)    leaq  .LC0(%rip), %rsi    leaq  \_ZSt4cout(%rip), %rdi |
| Cout<< “Enter a number:\n”<<endl; | .LC0:    .string "Enter a number:\n" |
| Cin>> num |  |
| For(i=num; i>0; i--)  Product = product \* I; | movl -12(%rbp), %eax    movl  %eax, %esi    movq  %rdx, %rdi    call  \_ZNSolsEi@PLT    movl  $1, %eax    movq  -8(%rbp), %rcx    xorq  %fs:40, %rcx |
| Cout<<”The factorial for” << num<< “is: \n” << product; | .LC1:    .string "The factorial for "  .LC2:    .string "is: \n"    .text |
| Return 1; | leave    .cfi\_def\_cfa 7, 8    ret    .cfi\_endproc |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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** |
| --- | --- |
| .LC0:    .string "Enter a number:\n"   .LC1:    .string "The factorial for "  .LC2:    .string "is: \n"    .text | These are the variables that are used to display any messages to the user. Where LC0 is used to prompt the user to enter their number, LC1 is used to display the first part of the factorial message and, LC2 is the second part of the factorial message |
| movl $1, -12(%rbp) | Initializes a variable to the value of 1, this is accomplished by placing 1 in-12(%rbp). |
| movl -20(%rbp), %eax    movl  %eax, -16(%rbp)  .L3:    cmpl  $0, -16(%rbp)    jle .L2    movl  -12(%rbp), %eax    imull -16(%rbp), %eax    movl  %eax, -12(%rbp)    subl  $1, -16(%rbp)    jmp .L3 | Loops for everytime “I” is greater than 0. This is done by reading the variable, moving it to %eax, then compiles and subtracts 1 before continuing the loop |
| movl -12(%rbp), %eax | Assigns the multiplication of “product” and “I” to “product” |
| .L5:    leave    .cfi\_def\_cfa 7, 8    ret    .cfi\_endproc | Exits the program |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |