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

Step 1: Convert the assembly code into C++ code.

Step 2: Explain the function of the converted C++ code.

| **Assembly Code** | **C++ Code** | **Explanation of Functionality** |
| --- | --- | --- |
| movl −8(%rbp), %eax sall $3, %eax subl $3, %eax movl %eax, −4(%rbp) | Int A;  Int B;  A = A \*8;  A = A -3;  B = A; | Move the contents of -8(%rbp), to %eax, it is then multiplied by 8 this is done to shift the A integer to the left 3 bits, it is then subtracted by 3, finally it is stored in the B integer. |
| movl −8(%rbp), %eax sall $2, %eax subl $1, %eax leal 7(%rax), %edx testl %eax, %eax cmovs %edx, %eax sarl $3, %eax  movl %eax, −4(%rbp) | Int A;  Int B;  A = (A \* 4) -1;  Int C = A + 7;  If( A == 0){  A = C;  }  A = A/8;  B = A; | The variable is read using -8(%rbp), and moved into %eax, it is then multiplied by 4 to shift A to the left 2 bits, then subtracted by 1, A new variable is made and stores the result of adding 7 to A, A is then checked to see if it is equal to 0 if it is A takes the value of 7, A is then divided by 8 to shift the bits 3 to the right, finally the value for A is stored in the B variable |
| movl −8(%rbp), %eax leal 7(%rax), %edx testl %eax, %eax cmovs %edx, %eax sarl $3, %eax movl −8(%rbp), %edx sall $2, %edx addl %edx, %eax  movl %eax, −4(%rbp) | Int A;  Int B;  Int C = A + 7;  If(A == 0){  A = C;  }  A = A / 8;  Int D = A;  Int E;  E = E \* 4;  A = D + E;  B = A; | Just like the code above it calculates C which is just A +7, and then checks if A is equal to 0 if it is it assumes the value we have stored in C, A is then divided by 8 in order to shift the bits 3 to the right, we then initialize a new variable in this case D and assign it the same value as A, then we have E (which is the same as our value that was read from -8(%rbp)) multiplied by 4 to shift the bits to the left by 2, following that A is given the new value of D + E, and then finally the value B takes on the value of A. |