# **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 input;  Int output = (input \* 8) -3; | 1. Move contents of -8(rbp) to %eax 2. Take value of %eax and multiply it by 8 (shifts 3 bits to the left) 3. Subtract 3 from the contents of %eax 4. Move contents of %eax into -4(rbp) |
| 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 input;  Int output = (((input \* 4) -1)/8); | 1. Move contents of -8(rbp) to %eax 2. Take value of %eax and multiply it by 4 (shifts 2 bits to the left) 3. Subtract 1 from the contents of %eax 4. Load effective address – put memory address of 7(%rax) into %edx 5. Test %eax to see if it’s above zero (AND) 6. Conditional move if negative for %edx to %eda 7. Take value of %eax and divide it by 8 (shifts 3 bits to the right) 8. Move contents of %eax into -4(rbp) |
| 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 Input;  Int output = (input/8) \* 4;  Input – input + output; | 1. Move contents of -8(rbp) to %eax 2. Load effective address – put memory address of 7(%rax) into %edx 3. Test %eax to see if it’s above zero (AND) 4. Test %eax to see if it’s above zero (AND) 5. Take value of %eax and divide it by 8 (shifts 3 bits to the right) 6. Move contents of -8(%rbp) to %edx 7. Take value of %eax and multiply it by 4 (shifts 2 bits to the left) 8. Add %edx from the contents of %eax 9. Move contents of %eax into -4(%rbp) |