# **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 x;  int result = (x << 3) - 3; | 1. Move contents of -8(%rbp) to %eax 2. Shift left the value of %eax by 3 positions. 3. Subtract 3 from the value of %eax 4. Move contents of %eax to −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 x;  int y = ((x << 2) – 1) >> 3; | 1. Move contents of -8(%rbp) to %eax 2. Shift left the value of %eax by 2 positions. 3. Subtract 1 from the value of %eax 4. Add the value of 7 into %rax and store into %edx 5. Test logical for %eax to determine if above zero (AND operation) 6. IF previous value is negative, move %edx to %eax 7. Shift right the value of %eax by 3 positions. 8. Move contents of %eax to −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 x;  int y = (x << 3) + (x << 2); | 1. Move contents of -8(%rbp) to %eax 2. Add the value of 7 into %rax and store into %edx 3. Test logical for %eax to determine if above zero (AND operation) 4. IF previous value is negative, move %edx to %eax 5. Shift right the value of %eax by 3 positions. 6. Move contents of −8(%rbp) to %edx 7. Shift left the value of %edx by 2 positions. 8. Add the value of %edx with the value of %eax 9. Move contents of %eax to −4(%rbp) |