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

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;  b = (b\*8);  b = b – 3; | * Move value of %rbp(-8) to %eax * Shift the stored value in %eax 3 bits to the left * Subtract 3 from %eax value * Move value of %eax to %rbp(-4) |
| 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;  b = b \* 4;  b = b – 1;  b = b/8;  a = b; | * Move %rbp value to %eax. * Multiply %eax value by 4. * Subtract 1 from %eax value * Load address of %rax into %edx * Test to ensure %eax value is above zero * Move value of %edx to $eda if negative for two values listed. * Take %eax value and divide it by 8 * Pass contents of %eax into %rpb register |
| 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;  b = b/8;  b = b \* 4;  a = b; | * Move %rbp(-8) value to %eax. * Add %rax value and store in %edx * Test %eax to determine value type: +, -, 0 * Move value of %edx to %eax if negative * Shift stored value of %eax 3 bits to the right * Move %rbp(-8) value to %edx * Shift stored value of %eax 2 bits to the left * Add value of %edx to %eax * Move value of %eax to %rbp(-4) |