# **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) | I = n;  i = i \* (2^3);  i = i – 3; | I = eax  n = -8(%rbp)  The first line fetches i from rbp. Sall shifts the bits 3 places which is equal to \* (2^2). Subl then subtracks 3 from i. The value is then placed above the original value on the rbp stack. |
| 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) | I = n;  I = i \* (2^2);  i = i – 1;  j = 7 + k;  if (i == 0){  i = j;  }  i = i/(2^3); | I = eax  j = edx  n = -8(%rbp)  k = rax  Movl initializes variable i by equaling it to a value on the rbp stack. Sall shifts the bits 2 spaces which is equal to \* (2^2). Subl subtracks 1 from i. Leal add 7 to the value from rax to edx. Testl checks for a flag that eax = eax which equates to if i is equal to 0. If that is the case then i is set to equal j. Sarl moves the bits 3 places to the opposite of sall which equates to / (2^3). Lastly i is then moved back onto the rbp stack above the original value. |
| 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) | I = n;  j = 7 + k;  if (i == 0){  i = j;  }  i = i/(2^3);  j = n;  j = j \* (2^ 2)  i = i + j; | I = eax  j = edx  n = -8(rbp)  k = rax  See paragraph below. Grid cuts off part of it. |

Movl initializes i from the rbp stack. Leal adds 7 + the value from rax to j. Testl checks if eax is equal to eax or eax is equal to 0. If that is true then i is set to equal j. Sarl divides i by 2^3. Movl sets j to the value at rbp. Sall multiplies j by (2^2). Addl adds j to i. Lastly i is moved to the rbp stack 4 bytes higher than the original value.