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

## **File One**

**Step 2:** Explain the functionality of the blocks of assembly code.

| **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- |
| 0x000000000000090a <+0>: push %rbp  0x000000000000090b <+1>: mov %rsp,%rbp  0x000000000000090e <+4>: sub $0x10,%rsp | Begins the program |
| 0x0000000000000912 <+8>: movl $0x1,-0x8(%rbp)  0x0000000000000919 <+15>: cmpl $0x9,-0x8(%rbp)  0x000000000000091d <+19>: jg 0x9ad <main+163>  0x0000000000000923 <+25>: movl $0x1,-0xc(%rbp)  0x000000000000092a <+32>: cmpl $0x9,-0xc(%rbp)  0x000000000000092e <+36>: jg 0x9a4 <main+154>  0x0000000000000930 <+38>: mov -0x8(%rbp),%eax  0x0000000000000933 <+41>: imul -0xc(%rbp),%eax  0x0000000000000937 <+45>: mov %eax,-0x4(%rbp)  0x000000000000093a <+48>: mov -0x8(%rbp),%eax  0x000000000000093d <+51>: mov %eax,%esi | Gets a value and checks if greater than stored value. If greater, jumps. If not moves to next value and checks for greater value. |
| 0x000000000000093f <+53>: lea 0x2006da(%rip),%rdi # 0x201020 <\_ZSt4cout@@GLIBCXX\_3.4>  0x0000000000000946 <+60>: callq 0x7e0 <\_ZNSolsEi@plt>  0x000000000000094b <+65>: lea 0x153(%rip),%rsi # 0xaa5  0x0000000000000952 <+72>: mov %rax,%rdi  0x0000000000000955 <+75>: callq 0x7b0 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  0x000000000000095a <+80>: mov %rax,%rdx  0x000000000000095d <+83>: mov -0xc(%rbp),%eax  0x0000000000000960 <+86>: mov %eax,%esi  0x0000000000000962 <+88>: mov %rdx,%rdi  0x0000000000000965 <+91>: callq 0x7e0 <\_ZNSolsEi@plt>  0x000000000000096a <+96>: lea 0x138(%rip),%rsi # 0xaa9  0x0000000000000971 <+103>: mov %rax,%rdi  0x0000000000000974 <+106>: callq 0x7b0 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  0x0000000000000979 <+111>: mov %rax,%rdx  0x000000000000097c <+114>: mov -0x4(%rbp),%eax  0x000000000000097f <+117>: mov %eax,%esi  0x0000000000000981 <+119>: mov %rdx,%rdi  0x0000000000000984 <+122>: callq 0x7e0 <\_ZNSolsEi@plt>  0x0000000000000989 <+127>: mov %rax,%rdx  0x000000000000098c <+130>: mov 0x20063d(%rip),%rax # 0x200fd0  0x0000000000000993 <+137>: mov %rax,%rsi  0x0000000000000996 <+140>: mov %rdx,%rdi  0x0000000000000999 <+143>: callq 0x7c0 <\_ZNSolsEPFRSoS\_E@plt> | Moves values to registers and performs print functions |
| 0x000000000000099e <+148>: addl $0x1,-0xc(%rbp)  0x00000000000009a2 <+152>: jmp 0x92a <main+32>  0x00000000000009a4 <+154>: addl $0x1,-0x8(%rbp)  0x00000000000009a8 <+158>: jmpq 0x919 <main+15> | Adds to stored values and jumps to compare the value for greater |
| 0x00000000000009ad <+163>: mov $0x0,%eax  0x00000000000009b2 <+168>: leaveq  0x00000000000009b3 <+169>: retq | Exits program |

**Step 4:** Convert the assembly code to C++ code.

**Step 5:** Explain how the C++ code performs the same tasks as the blocks of assembly code.

| **Blocks of Assembly Code** | **C++ Code** | **Explanation of Functionality** |
| --- | --- | --- |
| 0x0000000000000912 <+8>: movl $0x1,-0x8(%rbp)  0x0000000000000919 <+15>: cmpl $0x9,-0x8(%rbp)  0x000000000000091d <+19>:  cmpl $0x9,-0xc(%rbp)  0x0000000000000923 <+25>: movl $0x1,-0xc(%rbp)  0x000000000000092e <+36>:  imul -0xc(%rbp),%eax | Int number,I,a,x  a=1  a<=9  i=1  i <=9  X= a \* i | Declares variable  Begins loop and checks that a < 9 |
| 0x000000000000093f <+53>: lea 0x2006da(%rip),%rdi # 0x201020 <\_ZSt4cout@@GLIBCXX\_3.4>  0x0000000000000946 <+60>: callq 0x7e0 <\_ZNSolsEi@plt>  0x000000000000094b <+65>: lea 0x153(%rip),%rsi # 0xaa5  0x0000000000000955 <+75>: callq 0x7b0 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  0x0000000000000965 <+91>: callq 0x7e0 <\_ZNSolsEi@plt>  0x000000000000096a <+96>: lea 0x138(%rip),%rsi # 0xaa9  0x0000000000000974 <+106>: callq 0x7b0 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  0x0000000000000984 <+122>: callq 0x7e0 <\_ZNSolsEi@plt>  0x0000000000000999 <+143>: callq 0x7c0 <\_ZNSolsEPFRSoS\_E@plt> | Cout <<a<<”\*”<<i<<”=”<<x<<endl | Calls for cout to print, moves strings to registers, moves variables to callable registers, calls to print endline |
| 0x000000000000099e <+148>: addl $0x1,-0xc(%rbp)    0x00000000000009a4 <+154>: addl $0x1,-0x8(%rbp)  0x00000000000009a8 | a++  i++ | Increments a by 1  Increments i by 1 |
| 0x00000000000009b3 <+169>: retq | Return 0; | Returns 0 and exits |

## **File Two**

**Step 2:** Explain the functionality of the blocks of assembly code.

| **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- |
| 0x00000000000009fa <+0>: push %rbp  0x00000000000009fb <+1>: mov %rsp,%rbp  0x00000000000009fe <+4>: sub $0x30,%rsp | Begins the program and moves stacks |
| 0x0000000000000a02 <+8>: mov %fs:0x28,%rax  0x0000000000000a0b <+17>: mov %rax,-0x8(%rbp)  0x0000000000000a0f <+21>: xor %eax,%eax | Declares variables to be used |
| 0x0000000000000a11 <+23>: lea 0x191(%rip),%rsi # 0xba9  0x0000000000000a18 <+30>: lea 0x201601(%rip),%rdi # 0x202020 <\_ZSt4cout@@GLIBCXX\_3.4>  0x0000000000000a1f <+37>: callq 0x890 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  0x0000000000000a24 <+42>: mov %rax,%rdx  0x0000000000000a27 <+45>: mov 0x2015a2(%rip),%rax # 0x201fd0  0x0000000000000a2e <+52>: mov %rax,%rsi  0x0000000000000a31 <+55>: mov %rdx,%rdi  0x0000000000000a34 <+58>: callq 0x8a0 <\_ZNSolsEPFRSoS\_E@plt> | Performs a print and calls the strings defining what the user must input |
| 0x0000000000000a39 <+63>: lea -0x14(%rbp),%rax  0x0000000000000a3d <+67>: mov %rax,%rsi  0x0000000000000a40 <+70>: lea 0x2016f9(%rip),%rdi # 0x202140 <\_ZSt3cin@@GLIBCXX\_3.4>  0x0000000000000a47 <+77>: callq 0x870 <\_ZNSirsERi@plt> | Gets a user entered value and moves the value to the registers |
| 0x0000000000000a4c <+82>: mov -0x14(%rbp),%edx  0x0000000000000a4f <+85>: mov -0x14(%rbp),%eax  0x0000000000000a52 <+88>: imul %eax,%edx  0x0000000000000a55 <+91>: mov -0x14(%rbp),%eax  0x0000000000000a58 <+94>: imul %edx,%eax  0x0000000000000a5b <+97>: mov %eax,-0x14(%rbp)  0x0000000000000a5e <+100>: mov -0x14(%rbp),%eax  0x0000000000000a61 <+103>: cvtsi2sd %eax,%xmm0  0x0000000000000a65 <+107>: movsd 0x15b(%rip),%xmm1 # 0xbc8  0x0000000000000a6d <+115>: mulsd %xmm1,%xmm0  0x0000000000000a71 <+119>: movsd %xmm0,-0x10(%rbp) | Performs the calculation for volume of a circle |
| 0x0000000000000a76 <+124>: lea 0x13a(%rip),%rsi # 0xbb7  0x0000000000000a7d <+131>: lea 0x20159c(%rip),%rdi # 0x202020 <\_ZSt4cout@@GLIBCXX\_3.4>  0x0000000000000a84 <+138>: callq 0x890 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt>  0x0000000000000a89 <+143>: mov %rax,%rdx  0x0000000000000a8c <+146>: mov -0x10(%rbp),%rax  0x0000000000000a90 <+150>: mov %rax,-0x28(%rbp)  0x0000000000000a94 <+154>: movsd -0x28(%rbp),%xmm0  0x0000000000000a99 <+159>: mov %rdx,%rdi  0x0000000000000a9c <+162>: callq 0x8d0 <\_ZNSolsEd@plt> | Performs the print function and calls the string as well as necesarry variables to display the result |
| 0x0000000000000aa1 <+167>: mov $0x0,%eax  0x0000000000000aa6 <+172>: mov -0x8(%rbp),%rcx  0x0000000000000aaa <+176>: xor %fs:0x28,%rcx  0x0000000000000ab3 <+185>: je 0xaba <main+192>  0x0000000000000ab5 <+187>: callq 0x8b0 <\_\_stack\_chk\_fail@plt> | Clears registers, and moves to return statement, error call |
| 0x0000000000000aba <+192>: leaveq  0x0000000000000abb <+193>: retq | Exits the program |

**Step 4:** Convert the assembly code to C++ code.

**Step 5:** Explain how the C++ code performs the same tasks as the blocks of assembly code.

| **Blocks of Assembly Code** | **C++ Code** | **Explanation of Functionality** |
| --- | --- | --- |
| 0x0000000000000a02 <+8>: mov %fs:0x28,%rax  0x0000000000000a0b <+17>: mov %rax,-0x8(%rbp) | Double radius;  Double volume; | Declares radius as variable;  Declares volume as a variable; |
| 0x0000000000000a11 <+23>: lea 0x191(%rip),%rsi # 0xba9  0x0000000000000a18 <+30>: lea 0x201601(%rip),%rdi # 0x202020 <\_ZSt4cout@@GLIBCXX\_3.4>  0x0000000000000a1f <+37>: callq 0x890 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt> | Cout << “Enter Radius:” << endl | Prints “Enter Radius:” |
| 0x0000000000000a3d <+67>: mov %rax,%rsi  0x0000000000000a40 <+70>: lea 0x2016f9(%rip),%rdi # 0x202140 <\_ZSt3cin@@GLIBCXX\_3.4> | Cin << volume<< endl; | Accepts user input and stores to radius variable |
| 0x0000000000000a4c <+82>: mov -0x14(%rbp),%edx  0x0000000000000a4f <+85>: mov -0x14(%rbp),%eax  0x0000000000000a52 <+88>: imul %eax,%edx  0x0000000000000a58 <+94>: imul %edx,%eax  0x0000000000000a61 <+103>: cvtsi2sd %eax,%xmm0  0x0000000000000a65 <+107>: movsd 0x15b(%rip),%xmm1 # 0xbc8  0x0000000000000a6d <+115>: mulsd %xmm1,%xmm0  0x0000000000000a71 <+119>: movsd %xmm0,-0x10(%rbp) | Height = radius;  Volume = pi \* (radius \* radius)\*height | Stores radius entered to be the same as height  Performs the caluclation for cylinder volume () |
| 0x0000000000000a84 <+138>: callq 0x890 <\_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc@plt> | Cout<<The volume is<< volume; | Prints the output of the volume with the notifying string |
| 0x0000000000000abb <+193>: retq | Return 0; | Returns zero and exits program |

## **File Three**

**Step 2:** Explain the functionality of the blocks of assembly code.

| **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Step 4:** Convert the assembly code to C++ code.

**Step 5:** Explain how the C++ code performs the same tasks as the blocks of assembly code.

| **Blocks of Assembly Code** | **C++ Code** | **Explanation of Functionality** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## **File Four**

**Step 2:** Explain the functionality of the blocks of assembly code.

| **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Step 4:** Convert the assembly code to C++ code.

**Step 5:** Explain how the C++ code performs the same tasks as the blocks of assembly code.

| **Blocks of Assembly Code** | **C++ Code** | **Explanation of Functionality** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |