# **CS 410 Binary to Assembly Activity Template**

**Step 1:** List the binary file name.

**Step 2:** Identify the functions in the binary file.

**Step 3**: Convert the binary file to assembly code.

**Step 4:** Align the blocks of assembly code with their corresponding function in the binary file.

**Step 5:** Explain the functionality of the blocks of assembly code.

**File One: assignment3\_1.o**

| **Functions** | **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- | --- |
| 0x000000000040057d main | 40057d: 55 push %rbp  40057e: 48 89 e5 mov %rsp,%rbp  400581: bf 34 06 40 00 mov $0x400634,%edi  400586: e8 c5 fe ff ff callq 400450 <puts@plt>  40058b: bf 48 06 40 00 mov $0x400648,%edi  400590: e8 bb fe ff ff callq 400450 <puts@plt>  400595: bf 5c 06 40 00 mov $0x40065c,%edi  40059a: e8 b1 fe ff ff callq 400450 <puts@plt>  40059f: bf 00 00 00 00 mov $0x0,%edi  4005a4: e8 d7 fe ff ff callq 400480 <exit@plt>  4005a9: 0f 1f 80 00 00 00 00 nopl 0x0(%rax) | This method returns a hardcoded string ‘Ship to:…’ |
|  |  |  |
|  |  |  |
|  |  |  |

**File Two: assignment3\_2.o**

| **Functions** | **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- | --- |
| 000000000040062d <main>: | 40062d: 55 push %rbp  40062e: 48 89 e5 mov %rsp,%rbp  400631: 48 83 ec 20 sub $0x20,%rsp  400635: 64 48 8b 04 25 28 00 mov %fs:0x28,%rax  40063c: 00 00  40063e: 48 89 45 f8 mov %rax,-0x8(%rbp)  400642: 31 c0 xor %eax,%eax  400644: bf 14 07 40 00 mov $0x400714,%edi  400649: e8 92 fe ff ff callq 4004e0 <puts@plt>  40064e: 48 8d 45 e0 lea -0x20(%rbp),%rax  400652: 48 89 c6 mov %rax,%rsi  400655: bf 2b 07 40 00 mov $0x40072b,%edi  40065a: b8 00 00 00 00 mov $0x0,%eax  40065f: e8 bc fe ff ff callq 400520 <\_\_isoc99\_scanf@plt>  400664: 48 8d 45 e0 lea -0x20(%rbp),%rax  400668: 48 89 c6 mov %rax,%rsi  40066b: bf 2e 07 40 00 mov $0x40072e,%edi  400670: b8 00 00 00 00 mov $0x0,%eax  400675: e8 76 fe ff ff callq 4004f0 <printf@plt>  40067a: bf 00 00 00 00 mov $0x0,%edi  40067f: e8 ac fe ff ff callq 400530 <exit@plt>  400684: 66 2e 0f 1f 84 00 00 nopw %cs:0x0(%rax,%rax,1)  40068b: 00 00 00  40068e: 66 90 xchg %ax,%ax | Takes in a string and then returns ‘Welcome Mr. <string>’ |
|  |  |  |
|  |  |  |
|  |  |  |

**File Three: assignment3\_3.o**

| **Functions** | **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- | --- |
| 0000000000400641 <main>: | 400641: 55 push %rbp  400642: 48 89 e5 mov %rsp,%rbp  400645: 48 83 ec 10 sub $0x10,%rsp  400649: bf 34 07 40 00 mov $0x400734,%edi  40064e: e8 8d fe ff ff callq 4004e0 <puts@plt>  400653: 48 8d 55 f8 lea -0x8(%rbp),%rdx  400657: 48 8d 45 f4 lea -0xc(%rbp),%rax  40065b: 48 89 c6 mov %rax,%rsi  40065e: bf 47 07 40 00 mov $0x400747,%edi  400663: b8 00 00 00 00 mov $0x0,%eax  400668: e8 b3 fe ff ff callq 400520 <\_\_isoc99\_scanf@plt>  40066d: 8b 55 f8 mov -0x8(%rbp),%edx  400670: 8b 45 f4 mov -0xc(%rbp),%eax  400673: 89 d6 mov %edx,%esi  400675: 89 c7 mov %eax,%edi  400677: e8 b1 ff ff ff callq 40062d <AddNumbers>  40067c: 89 45 fc mov %eax,-0x4(%rbp)  40067f: 8b 55 f8 mov -0x8(%rbp),%edx  400682: 8b 45 f4 mov -0xc(%rbp),%eax  400685: 8b 4d fc mov -0x4(%rbp),%ecx  400688: 89 c6 mov %eax,%esi  40068a: bf 4d 07 40 00 mov $0x40074d,%edi  40068f: b8 00 00 00 00 mov $0x0,%eax  400694: e8 57 fe ff ff callq 4004f0 <printf@plt>  400699: bf 00 00 00 00 mov $0x0,%edi  40069e: e8 8d fe ff ff callq 400530 <exit@plt>  4006a3: 66 2e 0f 1f 84 00 00 nopw %cs:0x0(%rax,%rax,1)  4006aa: 00 00 00  4006ad: 0f 1f 00 nopl (%rax) | Takes an input of two numbers and adds them together |
|  |  |  |
|  |  |  |
|  |  |  |

**File Four: assignment3\_4.o**

| **Functions** | **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- | --- |
| 0000000000400717 <main>: | 400717: 55 push %rbp  400718: 48 89 e5 mov %rsp,%rbp  40071b: 48 83 ec 10 sub $0x10,%rsp  40071f: c7 45 f8 00 00 00 00 movl $0x0,-0x8(%rbp)  400726: eb 78 jmp 4007a0 <main+0x89>  400728: b8 00 00 00 00 mov $0x0,%eax  40072d: e8 ad ff ff ff callq 4006df <DisplayMenu>  400732: bf 86 08 40 00 mov $0x400886,%edi  400737: e8 a4 fd ff ff callq 4004e0 <puts@plt>  40073c: 48 8d 45 f8 lea -0x8(%rbp),%rax  400740: 48 89 c6 mov %rax,%rsi  400743: bf 99 08 40 00 mov $0x400899,%edi  400748: b8 00 00 00 00 mov $0x0,%eax  40074d: e8 ce fd ff ff callq 400520 <\_\_isoc99\_scanf@plt>  400752: 8b 45 f8 mov -0x8(%rbp),%eax  400755: 83 f8 03 cmp $0x3,%eax  400758: 74 20 je 40077a <main+0x63>  40075a: bf 9c 08 40 00 mov $0x40089c,%edi  40075f: e8 7c fd ff ff callq 4004e0 <puts@plt>  400764: 48 8d 45 fc lea -0x4(%rbp),%rax  400768: 48 89 c6 mov %rax,%rsi  40076b: bf 99 08 40 00 mov $0x400899,%edi  400770: b8 00 00 00 00 mov $0x0,%eax  400775: e8 a6 fd ff ff callq 400520 <\_\_isoc99\_scanf@plt>  40077a: 8b 45 f8 mov -0x8(%rbp),%eax  40077d: 83 f8 01 cmp $0x1,%eax  400780: 75 0c jne 40078e <main+0x77>  400782: 8b 45 fc mov -0x4(%rbp),%eax  400785: 89 c7 mov %eax,%edi  400787: e8 a1 fe ff ff callq 40062d <PrintFact>  40078c: eb 12 jmp 4007a0 <main+0x89>  40078e: 8b 45 f8 mov -0x8(%rbp),%eax  400791: 83 f8 02 cmp $0x2,%eax  400794: 75 0a jne 4007a0 <main+0x89>  400796: 8b 45 fc mov -0x4(%rbp),%eax  400799: 89 c7 mov %eax,%edi  40079b: e8 e8 fe ff ff callq 400688 <PrintSum>  4007a0: 8b 45 f8 mov -0x8(%rbp),%eax  4007a3: 83 f8 03 cmp $0x3,%eax  4007a6: 75 80 jne 400728 <main+0x11>  4007a8: bf 00 00 00 00 mov $0x0,%edi  4007ad: e8 7e fd ff ff callq 400530 <exit@plt>  4007b2: 66 2e 0f 1f 84 00 00 nopw %cs:0x0(%rax,%rax,1)  4007b9: 00 00 00  4007bc: 0f 1f 40 00 nopl 0x0(%rax) | Has a menu with 3 options and one is factorial two is summation and three is quit.  Takes an input of 1-3 for the options listed above and then for options 1 and 2 it either factorials the number or summations the number. Option 3 returns to the command line. |
|  |  |  |
|  |  |  |
|  |  |  |