# **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\_1.0

| **Functions** | **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- | --- |
| main() | 1. mov %edi, 0x400634 2. call puts 3. mov %edi, 0x400648 4. call puts 5. mov %edi, 0x40065c 6. call puts 7. mov %edi, 0x0 8. call exit | 1. Moves “Ship to: John Smith” to %edi 2. Outputs the contents of %edi 3. Moves “123 Los Angeles Rd.” to %edi 4. Output the contents of %edi 5. Moves “Los Angeles, CA 90025” to %edi 6. Output the contents of %edi 7. Moves 0 to %edi 8. Calls exit with the code of 0 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## File Two: assignment3\_2.0

| **Functions** | **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- | --- |
| Main() | 1. mov %edi, 0x400714 2. call puts 3. lea 0x20, %rax 4. mov %rax, %rsi 5. mov 0x40072b, %edi 6. mov 0x0, %eax 7. call \_\_isoc99\_scanf 8. lea 0x20, %rax 9. mov %rax, %rsi 10. mov 0x40072e, %edi 11. mov 0x0, %eax 12. call printf 13. mov 0x0, %edi 14. call exit | 1. Moved "Please enter your name" to % 2. Outputs contents of %edi 3. Reads the input from user into %rax 4. Move contents from %rax to %rsi 5. Moved contents of %rsi to %edi to be output 6. Moves 0 into %eax 7. Call scanf function using %edi 1st arg and %rsi as 2nd 8. Reads input and sets value of 0x20 to %rax 9. Moves contents of %rax into %rsi 10. Moves string "Welcome Mr. %s\n" into %edi 11. Move 0 into %eax 12. Calls printf “Welcome Mr. %s\n” using %edi as 1st arg and %rsi as 2nd 13. Move 0 into %edi 14. Calls exit with a code of 0 |
|  |  |  |
|  |  |  |
|  |  |  |

## File Three: assignment3\_3.0

| **Functions** | **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- | --- |
| AddNumbers() | 1. mov %edi, -0x4(%rbp) 2. mov %esi, -0x8(%rbp) 3. mov -0x8(%rbp), %eax 4. mov -0x4(%rbp), %edx 5. add %edx, %eax 6. pop %rbp 7. retn | 1. Moves 1st input into -0x4 2. Moves 2nd input into -0x8 3. Moves contents of -0x8 into %eax 4. Moves contents of -0x4 into %edx 5. Adds number from %edx and %eax 6. Pops the results back into the registers 7. Return back to main menu |
| Main() | 1. mov 0x400734, %edi 2. call puts 3. lea -0x8, %rdx 4. lea -0xc, %rax 5. mov %rax, %rsi 6. mov 0x400747, %edi 7. mov 0x0, %eax 8. call \_\_isoc99\_scanf 9. mov -0x8, %edx 10. mov -0xc, %eax 11. mov %edx, %esi 12. mov %eax, %edi 13. call AddNumbers 14. mov %eax, -0x4(%rbp) 15. mov -0x8(%rbp), %edx 16. mov -0xc(%rbp), %eax 17. mov -0x4(%rbp), %ecx 18. mov %eax, %esi 19. mov 0x40074d, %edi 20. mov 0x0, %eax 21. call printf 22. mov 0x0, %edi 23. call exit | 1. Moves string “Enter two numbers:” into %edi 2. Output contents of %edi 3. Reads 1st number -0x8 input from scanf() into %rdx 4. Reads 2nd number -0xc input from scanf() into %rax 5. Move contents of %rax to %rsi 6. Moves string %d %d into %edi using decimal formats 7. Moves 0 into %eax to return value 8. Calls scanf() using (%d %d, 1st num and 2nd num 9. Moves contents of -0x8 into %edx 10. Moves contents of -0xc into %eax 11. Moves contents of %edx to %esi 12. Moves contents of %eax into %edi 13. Calls AddNumbers function with %edi 1st arg and %esi as 2nd arg 14. Moves result from AddNumbers into -0x4 15. Moves the 2nd number to %edx 16. Moves the 1st number to %eax 17. Moves values of -0x4 to %ecx 18. Moves contents of %eax to %esi 19. Moves string “%d + %d = %d \n” into %edi 20. Moves 0 into %eax 21. Call print with %esi as 1st arg, %edx as 2nd, and %ecx as 3rd 22. Moves 0 into %edi 23. Calls exit with code 0 |

## File Four: assignment3\_4.o

| **Functions** | **Blocks of Assembly Code** | **Explanation of Functionality** |
| --- | --- | --- |
| Printfact | 1. mov %edi, -0x14(%rbp) 2. mov 0x1, -0x4(%rbp) 3. mov -0x14(%rbp), %eax 4. mov %eax, -0x8 5. jmp 0x400669 Printfact+60 6. mov -0x4(%rbp), %eax 7. imul -0x8(%rbp), %eax 8. mov %eax, -0x4(%rbp) 9. mov -0x8(%rbp), %eax 10. mov %eax, %esi 11. mov 0x400844, %edi 12. mov 0x0, %eax 13. call 0x4004f0 print 14. sub. 0x1, -0x8(%rbp) 15. cmp 0x0, -0x8(%rdp) 16. jg PrintFact+26 17. mov -0x4(%rbp), %eax 18. mov %eax, %esi 19. mov 0x400848, %edi 20. mov 0x0, %eax 21. call 0x4004f0 print 22. mov -0x4(%rbp), %eax 23. leave 24. ret | 1. Move number from %edi into -0x14 2. Moves 1 into -0x4(%rbp) 3. Move -0x14 into %eax 4. Moves %eax into -0x8 5. Jumps to line 60 of Printfact function 6. Move contents of -0x4 to %eax 7. Multiplies contents of -0x8 and %eax and stores in %eax 8. Moves contents %eax into -0x4 9. Moves contents of -0x8 into %eax 10. Moves contents of %eax into %esi 11. Move string “%d” into %edi 12. Moves 0 into %eax 13. Call printf function to output %esi 14. Subract 1 from contents of -0x8 and store in -0x8 15. Compares 1 with content of -0x8 16. Jump to PrintFact function line 26 17. Moves content of -0x4 into %eax 18. Moves %ea into %esi 19. Moves string “ %d \n” into %edi 20. Moves 0 into %eax 21. Call print function to output integer 22. Releases stack 23. Returns with code 0 |
| PrintSum | 1. mov %edi, -0x14(%rbp)  2. mov 0x0, -0x4(%rbp)  3. mov -0x14(%rbp), %eax  4. mov %eax, -0x8(%rbp)  5. jmp 0x4006c0  6. mov -0x8(%rbp), %eax  7. add %eax, -0x4(%rbp)  8. mov -0x8(%rbp), %eax  9. mov %eax, %esi  10. mov 0x400844, %edi  11. mov 0x0, %eax  12. call 0x4004f0  13. sub 0x1, -0x8(%rbp)  14. cmp 0x0, -0x8(%rbp)  15. jg 0x4006a2  16. mov -0x4(%rbp), %eax  17. mov %esi , %eax  18. mov 0x400848, %edi  19. mov 0x0, %eax  20. call printf  21. mov %eax, -0x4(%rbp)  22. leave  23. retn | 1. Moves content from %edi to 0x14  2. Moves 0 into -0x4  3. Moves -0x14 into %eax.  4. Moves %eax into 0x8  5. Jump to 0x4006c0  6. Moves contents from -0x8 into %eax  7. Add %eax and -0x4 and store in -0x4  8. Move -0x8 into %eax  9. Move %eax into %esi  10. Move “%d ” into %edi  11. Move 0 into %eax  12. Call printf() function to output %esi  13. Subtract 1 from -0x8  14. Compare 0 to -0x8  15. Jump to function Printsum() line 26 if -0x8 is greater than 0.  16. Moves -0x4 into %eax  17. Move %eax into %esi  18. Move “%d \n” into edi  19. Move 0 into %eax  20. Call print from %edi and and the value of %esi  21. Move value of -0x4 into %eax  22. Release stack  23. Return value store in %eax |
| DisplayMenu() | 1. mov %edi, 0x400851 2. call puts 3. mov %edi, 0x400864 4. call puts 5. mov %edi, 0x400871 6. call puts 7. mov %edi, 0x40087e 8. call puts 9. mov edi, 0x400851 10. call puts 11. pop %rbp 12. retn | 1. Move "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" into %edi 2. Output contents of %edi 3. Move "1. Factorial" to %edi 4. Output contents of %edi 5. Move "2. Summation" into %edi 6. Output contents of %edi 7. Move "3. Quit" into %edi 8. Output contents of %edi 9. Move "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" into %edi 10. Output contents of %edi 11. Pop stack into register 12. Return to previous |
| Main() | 1. mov %eax, 0x0 2. call DisplayMenu 3. mov %edi, 0x400886 4. call puts 5. lea %rax, -0x8(%rbp) 6. mov %rsi, %rax 7. mov %edi, 0x400899 8. mov %eax, 0x0 9. call \_\_isoc99\_scanf 10. mov %eax, 0x8(%rbp) 11. cmp %eax, 0x3 12. je 0x40077a 13. mov %edi, 0x40089c 14. call puts 15. lea %rax, --0x4%rbp) 16. mov %rsi, %rax 17. mov %edi, 18. mov %eax, -0x0 19. call \_\_isoc99\_scanf 20. mov %eax, -0x8(%rbp) 21. cmp %eax, -0x1(%rbp) 22. jne 0x40078e 23. mov %eax, -0x4(%rbp) 24. mov %edi, %eax 25. call PrintFact 26. jmp 0x4007a0 27. mov %eax, -0x8(%rbp) 28. cmp %eax, 0x2 29. jne 0x4007a0 30. mov %eax, 0x4(%rbp) 31. mov %edi, %eax 32. call PrintSum 33. mov %eax, -0x8(%rbp) 34. cmp %eax, 0x3 35. jne 0x400728 36. mov %edi, 0x0 37. call exit | 1. Move 0 into %eax 2. Call DisplayMenu 3. Move {"Enter your number:" into %edi 4. Output %edi 5. Read -0x8 and move into %rax 6. Move %rax into %rsi 7. Move into %edi 8. Move 0 into %eax 9. Call scanf using value from %rsi 10. Move from -0x8 into %eax 11. Compare 3 with %eax 12. Jump to 0x40077a 13. Moves "Enter a number:" into %edi 14. Output %edi 15. Read -0x4 to %rax 16. Move from %rax to %rsi 17. Move input to %edi 18. Move 0 into %eax 19. Call scanf using %rsi 20. Move 0x8 into %eax 21. Compare 1 to %eax 22. Jump to 0x40078e 23. Move -0x4 into %eax 24. Move %eax to %edi 25. Call PrintFact using %edi 26. Jump to 0x4007a0 27. Move -0x8 to %eax 28. Compare 2 to %eax 29. Jump to 0x4007a0 30. Move -0x4 to %eax 31. Move %eax to %edi 32. Call Printsum using %edi 33. Move -0x8 to %eax 34. Compare 3 to %eax 35. Jump to 0x400728 36. Move 0 to %edi 37. Exit with code 0 |