Unit 06_005 – Syntax

Video Length 12:45

1. What is the difference between .ascii and the .asciz directive?

2. How many bytes would the directive **`.ascii "dog"`** allocate?

3. How many bytes would the directive **`.asciz "dog"`** allocate?

4. The code below declares a string. Write the declaration of `len` that has the assembler calculate the length of message.

```
message: .ascii "Hello, World!"
len:     .quad
```

5. Modify the code below to load the address of message into the rdi register.

```
movq  letters, %rdi
```

6. Modify the code below to load the contents of rdi into the r8b register.

```
movb  %rdi , %r8b
```

7. Consider the previous two questions. One of them was moving a quad. The second was only moving a byte. Explain why.

Unit 06_010 – Syscall

Video Length 11:30

8. What is syscall? Why do we need it?

9. Suppose you wanted to convert a program written for x86-64 to run on an ARM processor. Would syscall need to be adapted to run on ARM?

10. What three registers will we be using to communicate with syscall?

11. If syscall is considered a function call, how are parameters passed to the function?

Unit 06_020 Writing known length –

Video Length 14:00

12. What four registers are used for writing through the use of syscall? What goes in each register? (You may refer to the cheatsheet for Exam 02)

    1.

    2.

    3.

    4.

NOTE: In the video I forgot to mention the return value in RAX. After syscall, RAX contains the number of bytes written.

13. Write the 5 lines of code needed to print the string buffer named `message`. There is no variable with the length, but you know it will always be 30 bytes long.

Unit 06_030 – Writing null-terminated strings, Part 1

Video Length 13:45

14. Summarize in words (not in code) how to write a null terminated string if the length is not known in advance.

15. This was not discussed in the video. Look closely at the code in the body of the loop. Could any of the instructions done in the body of the loop have been done before the loop began?

Unit 06_030 – Optimized loop, part 2

Video Length 9:30

16. Explain, in words, how the loop can be made to run faster.

17. This is asking for your opinion. Am I being paranoid by putting the "movq $1, %rax" statement in the loop? Explain your answer.

18. Write out the loop by hand. Think about it as you write it. Did I miss any other improvements? If so, comment on them. (The instructor is old school, and he still thinks that writing things by hand helps you understand them.) Also, remember that the source code is on github in the sources folder if don't want to stare at the paused video to get the code.

If you have any lingering questions or problems, please write them here or see me.

CSC 264

MOV AX,BX
ADD AX,1
CALL FUNC

Assembly Language ar
Computer Architecture