ASSIGNMENT II

Vegard Aa Albretsen

Marius Pedersen

Task 1

1. Lines 5-13:

Et bilde som inneholder tekst, skjermbilde, Font, design

Automatisk generert beskrivelse

These lines of code provide words for us to use later. If we for example want to exit the program, a ‘System Exit’, you would need to put the number 1 in the eax-register and call ‘int 80h’. To make this easier for a human writing the code, we put 1 into a constant named ‘SYS\_EXIT’ so when we want to call a ‘System Exit’ we just write the word, which then places the number 1 into the register.

In the code it looks like ‘mov eax,SYS\_EXIT’ and ‘int 80h’ instead of ‘mov eax,1’ and ‘int 80h’.

It does not make a difference for the program, but it makes it a bit easier to read and understand. That said, you should also write ‘mov ebx,0 (or 1)’ before ‘int 80h’ so that the program exit indicates a successful (0) or unsuccessful (other) termination.

1. Line 17 is a part of the *.bss section*. Resb in ‘siffer resb 4’ stands for ‘reserve bytes’ and means that we have made a variable ‘siffer’ which has reserved 4 bytes for later use.

.bss stands for ‘Block Started by Symbol’ and is a section used for declaring variables. The system typically initializes these variables with the value of zero. This means that even though the variables in this section are not initialized by the programmer, the operating system typically zeroes out these sections and gets rid of ‘garbage values’.

1. The prints happen in the start of the ‘global \_start’ section. More specifically the lines 34-38 (in my code):

Et bilde som inneholder tekst, skjermbilde, Font, nummer

Automatisk generert beskrivelse

This puts the length of the message in edx, the message in ecx, loads the standard output file descriptor (1) into ebx to specify a standard output, and then puts SYS\_WRITE into eax which specify which system call to evoke.

When we then call ‘int 80h’ it sees that it has SYS\_WRITE (4) in eax, then checks ebx to see which type of output it should be, ecx to know where the message is, and edx to see how many signs the word is, so that the system knows where the end of the message is.

1. After the 43rd line is executed the program either continues on to line 45 if the input is valid, or jumps to line 59/60 (the Slutt: label) if the input is invalid.
2. Strictly in the ‘lessiffer:’ block it only says ‘push eax’ and ‘push ebx’, however there is no calls or jumps in the ‘lessiffer’ block so every time you call ‘lessiffer’ the program also executes ‘Lokke:’ which calls a sign from the keyboard, checks it and puts it into ecx.

Et bilde som inneholder tekst, skjermbilde, programvare

Automatisk generert beskrivelse

1. The ret on line 122 returns back to where you called the ‘lessiffer’ was called, either 43 or 50.

Et bilde som inneholder tekst, skjermbilde, Font

Automatisk generert beskrivelse

The code then returns to where you call the ‘lessiffer’ blocks where the program compares edx and 0, which will not be equal since ‘Feil’ puts 1 into edx to signalize a faulty reading. This then jumps to ‘Slutt’.