Arthur Wei

Sanjana Sasmal

Shreya K

Yugam

LAB 4 ASSEMBLY EXPLANATION

Quick Summery:

The program starts off in the data section witch I used in declaring variables. The symbolic identifier (label) X is used. Then under the X identifier 4 different commands are issued (the "." operator signals commands). These instructions consist of 4 long instructions which tells the system to make a doubleword spaces with the initial data being 1,5,2, and 18. X will be label for these numbers. After this, the label SUM is used, under it having another long instruction allocating 32bits of memory with the value of 0. After this, the text section is called. The text section is where the rest of the code goes. The first command that is issued is global start which globalizes the start function. Next the start function is defined the Assembler knows where to start compiling the program. In start, the first command is a "move long command" were the first operand, is $ 4 meaning 4 is an inline value. The next. Operand is where it will "move" (copy) it to the %eax which % tells the assembler you are referencing a register and eax is a 32-bit registry. After this, there will be value 4 in register eax. The next line, is almost the same, it is another move value where the registry EBX will be allocated with the value of 0. The next line will point the register ECX to the first value of X. After that, a loop is started with the section TOP:. At the start of the loop the assembler will add ECX to EBX which is the first value of the X label (1) to the registry we allocated to store the addition. After that, the assembler will add 4 bytes to ECX which is currently pointing to the first long of X. Each long is 4 bytes in length so adding 4 will move it to the next value. After that it decreases our counter registry (EAX) by one. Then it hits a condition. "Jump if not zero" Witch states that if EAX is not 0 (it currently isn't) then it will go back to the TOP section and re run everything again. It will do this 4 times and every time it will add the next value of X. Once the counter hits zero the program moves onto the Done section witch just says take all the added up data from registry EBX and move it to the first value of label SUM.