SHEET 10

#	Hex	Binary	Assembly code	Description
0	2e	001 0 1110	Load 14	Load the value of memory address 14 into the accumulator
1	b0	101 1 0000	Equal #0	If the value of the accumulator is equal to 0, skip the next instruction
2	d4	110 1 0100	Jump #4	Jump to instruction 4
3	e0	111 0 0000	Halt	Stop execution
4	2f	001 0 1111	Load 15	Load the value of memory address 15 into the accumulator
5	6f	011 0 1111	Add 15	Add the value of memory address 15 to the accumulator
6	4f	010 0 1111	Store 15	Store the value of the accumulator in memory address 15
7	2e	001 0 1110	Load 14	Load the value of memory address 14 into the accumulator
8	91	100 1 0001	Sub #1	Subtract 1 to the value of the accumulator
9	4e	010 0 1110	Store 14	Store the value of the accumulator in memory address 14
10	cb	110 0 1011	Jump 11	Jump to instruction 11
11	00	000 0 0000	DATA #0	This memory address stores value 0 when execution starts
12	00	000 0 0000	DATA #0	This memory address stores value 0 when execution starts
13	00	000 0 0000	DATA #0	This memory address stores value 0 when execution starts
14	06	000 0 0110	DATA #6	This memory address stores value 6 when execution starts
15	01	000 0 0001	DATA #1	This memory address stores value 1 when execution starts

ACC 6 1 2 6 5 5 2 4 5 4 4 4 8 4 3 3 8 16 3 2

2 16 32 2 1 1 32 64 1 0 0

MEM[14] 6 5 4 3 2 1 0

MEM[15] 1 2 4 8 16 32 64

The program starts loading the value of memory address 14 (value 6) into the accumulator. Then we see if this value contained in the accumulator is equal to 0. 6 is not equal to 0, so we go to jump instruction, which indicates us to go to instruction 4. There we load the value of memory address 15 (value 1) into the accumulator. Then the value of memory address 15 (value 1) is added so now the accumulator takes the value 2. This value (2) is stored in the memory address 15. Then the value of the memory address 14 (value 6) is loaded again into the accumulator and we subtract 1 from it, so the result is 5. 5 is stored in the memory address 14. Afterwards we jump to instruction 11 which indicates us to go to the memory address 0. There, 5 is loaded into the accumulator. It is compared with the value 0 and as 5 ≠ 0, we perform all other steps again and again. The loop ends when 0 is loaded into the accumulator because 0 is equal to 0 so the program stops the execution. As we see, the values of memory address 14 change by 1, starting with 6, 5... and ending with 0. The values of memory address 15 change in this form 2^n so in the end it is $2^6 = 64$

If the value stored in memory cell 14 is changed to 10 before execution starts, we can understand that the program will take longer to be executed. The values of memory address 14 will go by the same pattern as before (will be changed by 1) so 10, 9, 8... 0 while the values of memory address 15 will be doubled until the value of memory address 14 goes to 0. It will start with 1 and the last value will be $2^{10} = 1024$