ICS 2020 Problem Sheet #10

Problem 10.1:

a)b)

let Value in the accumulator=va

#	Hex	Binary	Assembly Code	Description
0	2e	001 0 1110	LOAD 14	Load the value inside the address 14 into the accumulator; va=6
1	b0	101 1 0000	EQUAL #0	No skipping because 6 is not equal to 0; va=6
2	d4	110 1 0100	JUMP #4	Jumps to the instruction in 4; va=6
3	e0	111 0 0000	HALT 0	Stops the execution but skipped by JUMP #4; va=6
4	2f	001 0 1111	LOAD 15	Load the value inside the address 15 into the accumulator; va=1
5	6f	011 0 1111	ADD 15	adding the value stored in address 15 to the value in the accumulator (1+1) and leaving it in the accumulator; va=2
6	4f	010 0 1111	STORE 15	Storing the value in the accumulator in the address 15; va=2
7	2e	001 0 1110	LOAD 14	Load the value inside the address 14 into the accumulator; va=6
8	91	100 1 0001	SUB #1	Subtracts 1 from the value last loaded in the accumulator(6-1=5) and leaving it in the accumulator; va=5
9	4e	010 0 1110	STORE 14	Storing the value in the accumulator in the address 14; va=5
10	cb	110 0 1011	JUMP 11	Jumps to the instruction in 11; va=5
11	00	000 0 0000	---------	---------------------
12	00	000 0 0000	---------	---------------------
13	00	000 0 0000	---------	---------------------
14	06	000 0 0110	---------	---------------------
15	01	000 0 0001	---------	---------------------

c)The value in memory cell 15 when the program halts is 64:

The program first loads the value stored inside the address 14 into the accumulator, then the program checks if the value inside the accumulator equals 0 or not, since 6!=0 then nothing is skipped, then the program skips to the instructions in 4: it loads the value inside the address 15 into the accumulator, then it adds the value stored by 15 to the value in the accumulator and stores the result in the address 15. then the program loads the value inside the address 14 into the accumulator then subtracts 1 from that value then it stores the result in the address 14 .now the value in 14 is 5, then the program starts over, it loads 5 into the accumulator since 5!=0 there is no skip (if value inside the accumulator at that step is 0 then JUMP #4 is skipped and HALT 0 is executed and therefore the execution will stop). We notice that everytime the program execute all the instructions the value stored in the address 14 is taken into the accumulator, 1 is subtracted from that value and the new value is then stored in the address 14, which means every time the program loops the value stored in 14 gets subtracted by 1 until it reaches 0, when it does (at the 7th loop) the instruction EQUAL #0 is true since 0=0 then JUMP #4 is skipped and HALT 0 is executed which stops the execution. The value in the address 15 at the 7th loop is 64: everytime the program loops the value in the address 15 get multiplied by 2 then the new value gets stored in the address 15, but this only happens if JUMP #4 isn’t skipped, JUMP #4 is skipped at the loop 7, then the final value stored in the address 15 is 2*2*2*2*2*2=2^6=64.

```
#include <stdio.h>
int main(){
int a=6;
int b=1;
while(1){
   if(a==0){
   break;
   }
   b=2*b;
   a--;
   printf("a=%d\n", a);
   printf("b=%d\n", b);
}
return 0;
}
```

d) the value stored in memory cell 14 is changed to 10:

#	Hex	Binary	Assembly Code	Description
0	2e	001 0 1110	LOAD 14	Load the value inside the address 14 into the accumulator; va=10
1	b0	101 1 0000	EQUAL #0	No skipping because 6 is not equal to 0; va=10
2	d4	110 1 0100	JUMP #4	Jumps to the instruction in 4; va=10
3	e0	111 0 0000	HALT 0	Stops the execution but skipped by JUMP #4; va=10
4	2f	001 0 1111	LOAD 15	Load the value inside the address 15 into the accumulator; va=1
5	6f	011 0 1111	ADD 15	adding the value stored in address 15 to the value in the accumulator (1+1) and leaving it in the accumulator; va=2
6	4f	010 0 1111	STORE 15	Storing the value in the accumulator in the address 15; va=2
7	2e	001 0 1110	LOAD 14	Load the value inside the address 14 into the accumulator; va=10
8	91	100 1 0001	SUB #1	Subtracts 1 from the value last loaded in the accumulator(10-1=9) and leaving it in the accumulator; va=9
9	4e	010 0 1110	STORE 14	Storing the value in the accumulator in the address 14; va=9
10	cb	110 0 1011	JUMP 11	Jumps to the instruction in 11; va=9
11	00	000 0 0000	---------	---------------------
12	00	000 0 0000	---------	---------------------
13	00	000 0 0000	---------	---------------------
14	A	000 0 1010	---------	---------------------
15	01	000 0 0001	---------	---------------------

when memory cell 14 is changed to 10 before the execution start, the value in memory cell 14 is decremented : since 10 was loaded from the address 14 to the accumulator, then 1 got subtracted from it which leaves a 9 in the accumulator which is then stored in the memory cell 14, this goes on and on until the value stored in the address 14 is 0; in the meantime, the value stored by the address 15 changes too:

```
#include <stdio.h>
int main(){
int a=10;
int b=1;
while(1){
   if(a==0)
   break;
   b=2*b;
   a--;
   printf("a= %d\n", a);
   printf("b= %d\n\n", b);}
return 0;}
```

Output:

```
a= 9                a= 4
b= 2                b= 64

a= 8                a= 3
b= 4                b= 128

a= 7                a= 2
b= 8                b= 256

a= 6                a= 1
b= 16               b= 512

a= 5                a= 0
b= 32               b= 1024
```

Therefore when memory cell 14 is changed to 10 the final value stored in b is 1024.