Microprocessor Final Project

Rahma Abdulhameed 7358

Project#8: ATM Machine

In ATM machine, an electronic lock circuit is used to authorize the card. This circuit has two input codes: The card number that consists of 16 bits and the password that consists of 4 bits. If the bank has 20 customers only uses the cards. Construct their database and store it in the memory. Then write a program for card authorization. The inputs of the program are the card number and the password. The output is one bit (0/1) that means (denied/allowed).

The database used is:

Card Number	6053	2387	4700	7448	7603	7356	1287	7878	5792	3571
Password	8	2	6	1	0	3	11	7	14	9

Card Number	0096	1567	6050	9012	8845	1358	3250	1111	1357	7893
Password	12	5	13	9	15	4	0	12	10	3

We start by saving the database in the memory; the card number as 16 bit number (a word) and the password as a 4 bit (a byte) number.

Then we proceed to take input from the user, we check if the password is in the right range (0→15), if not the user is asked to enter the password again.

Then the searching for the card number entered takes place by looping on the cards array, increase the index each time by 2 since the memory is byte addressable. While doing that a counter is counting the number of items compared in the array.

If the card number is found, the counter is used to get the corresponding password directly and then compare it with the one the user has entered.

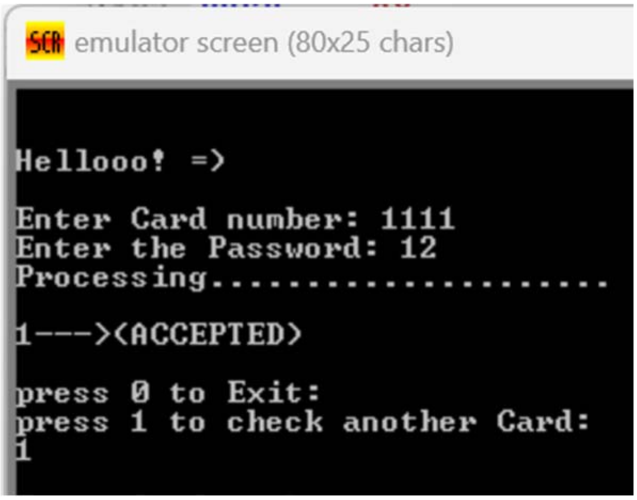
If it's not correct, a message indicating that the password is wrong is printed to the user and it's declined, however if it's correct, the card is accepted.

While if the card was not found, a jump is made and a message indicating that the card number is not found appears to the user and it's declined.

After that, the user has 2 options, either to enter another card or to exit the program. If the user chooses to continue, the whole previous process is repeated while as if he chooses to exit the program is stopped and the control is returned to the operating system. (Note that: the code is attached to this report!)

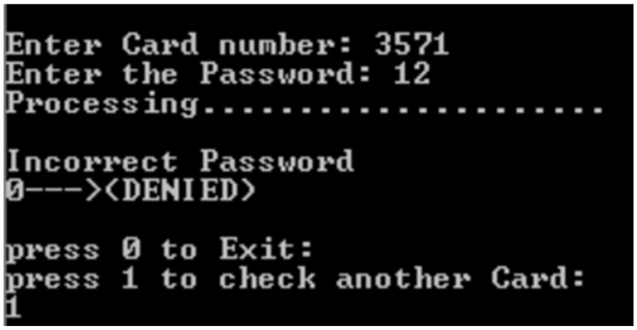
Some Samples runs to illustrates how the program works:

Case 1: If the card number and the password are correct:



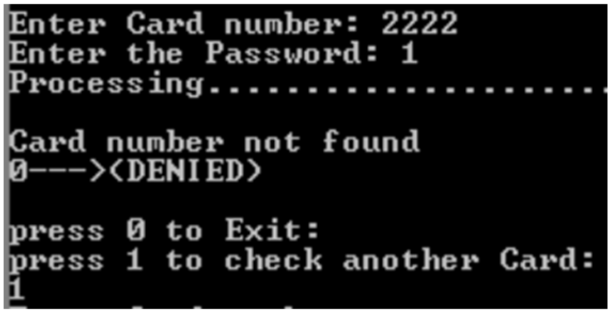
```
Hellooo! =>

Enter Card number: 1111
Enter the Password: 12
Processing.....................

1--->(ACCEPTED)

press 0 to Exit:
press 1 to check another Card:
1
```

Case 2: If the card number is found but the password is incorrect:



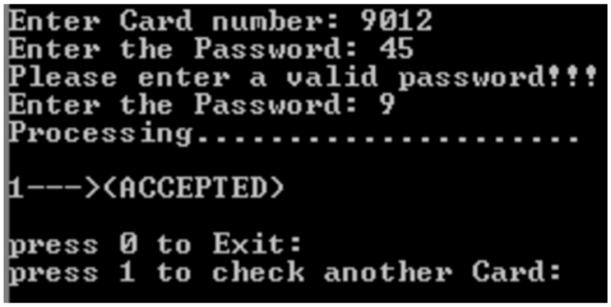
```
Enter Card number: 3571
Enter the Password: 12
Processing.....................

Incorrect Password
0--->(DENIED)

press 0 to Exit:
press 1 to check another Card:
1
```

Case 3: If the Card number is not found:



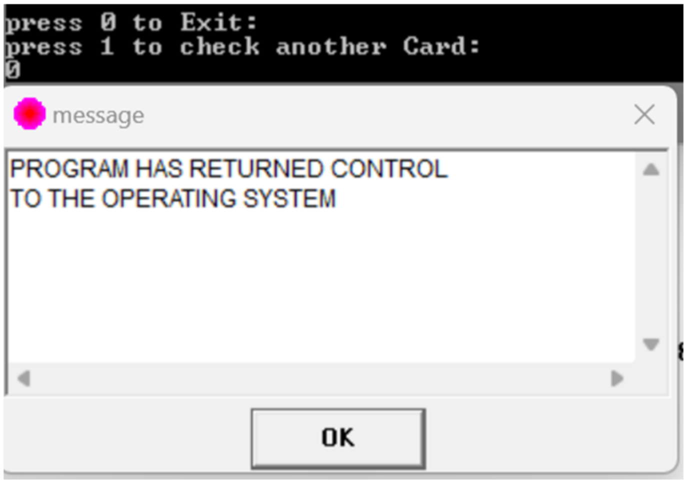
```
Enter Card number: 2222
Enter the Password: 1
Processing.....................

Card number not found
0--->(DENIED)

press 0 to Exit:
press 1 to check another Card:
1
```

Case 4 : If the password entered is out of range:



```
Enter Card number: 9012
Enter the Password: 45
Please enter a valid password!!!
Enter the Password: 9
Processing.....................

1--->(ACCEPTED)

press 0 to Exit:
press 1 to check another Card:
```

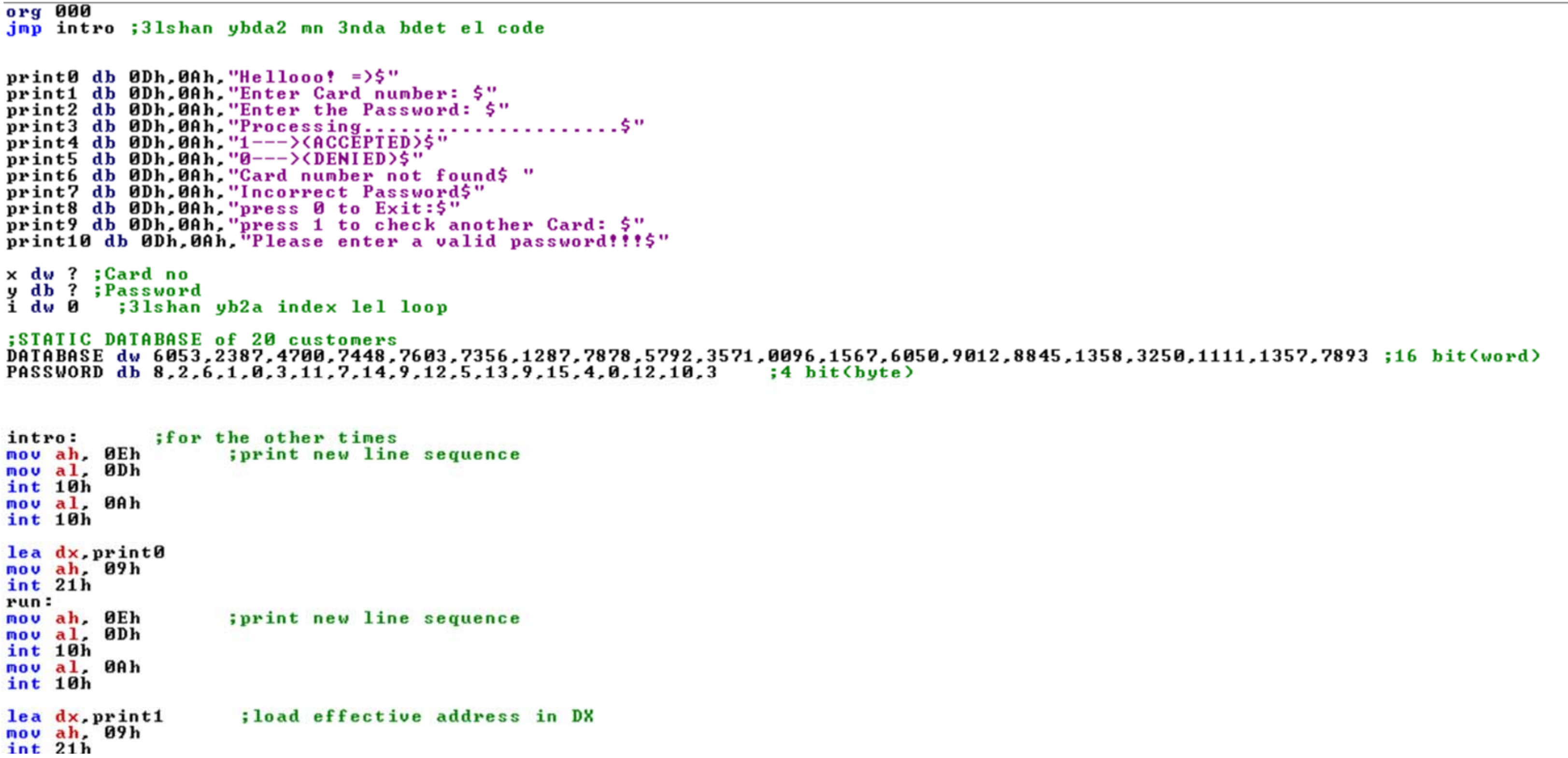
By pressing 0 to exit:



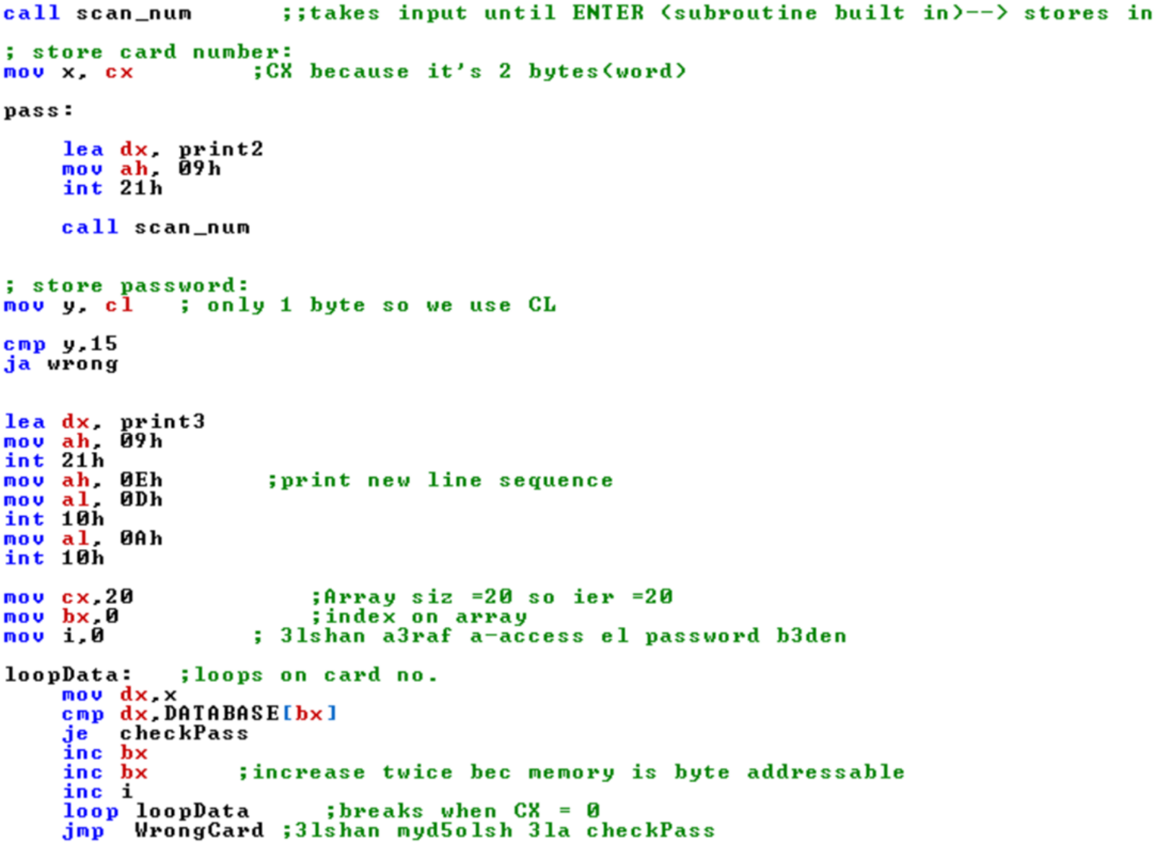
```
press 0 to Exit:
press 1 to check another Card:
0
```

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

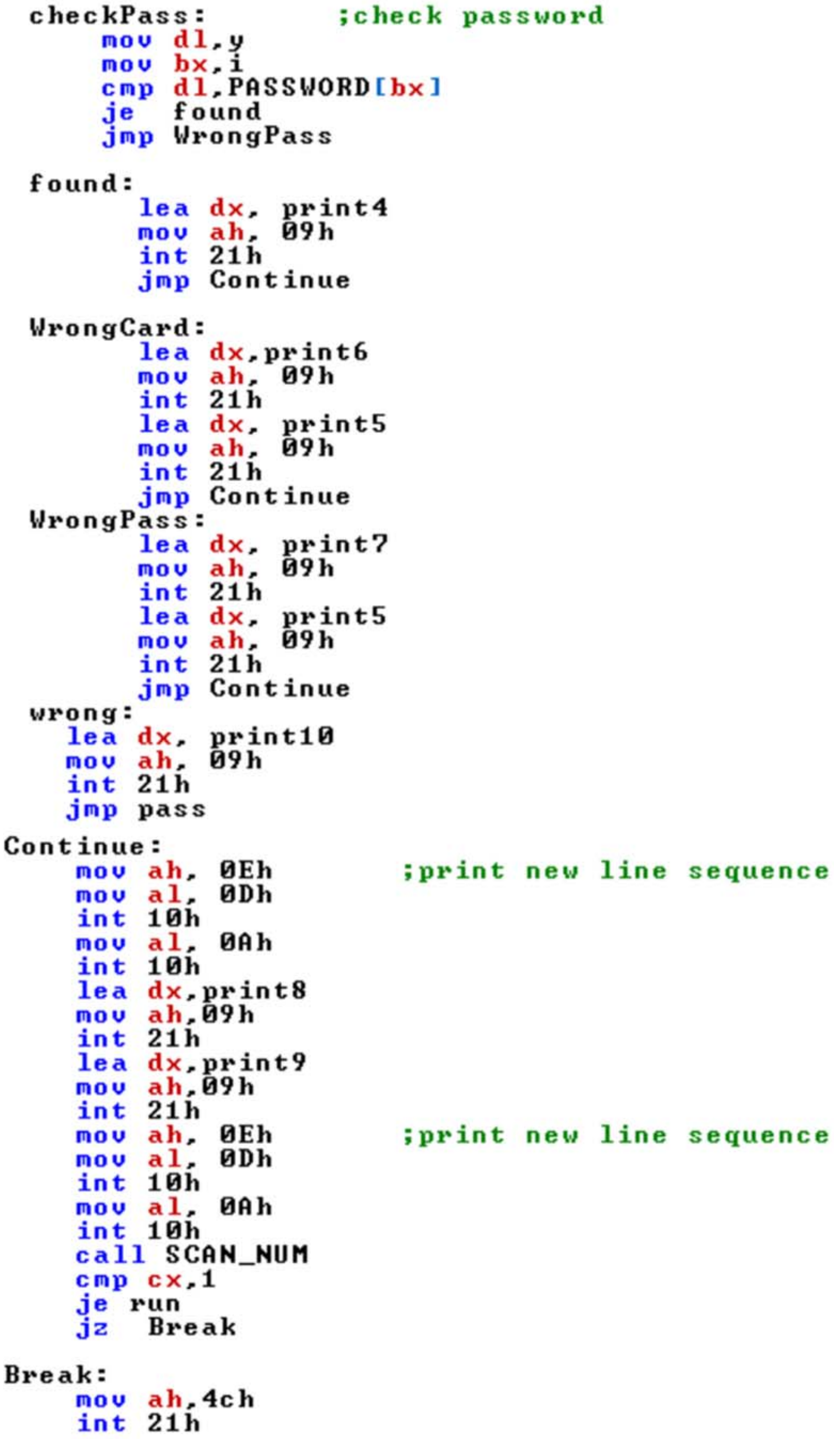
The Code:



```
org 000
jmp intro ;3lshan ybda2 mn 3nda bdet el code


print0 db 0Dh,0Ah,"Hellooo! =>$"
print1 db 0Dh,0Ah,"Enter Card number: $"
print2 db 0Dh,0Ah,"Enter the Password: $"
print3 db 0Dh,0Ah,"Processing......................$"
print4 db 0Dh,0Ah,"1--->(ACCEPTED)$"
print5 db 0Dh,0Ah,"0--->(DENIED)$"
print6 db 0Dh,0Ah,"Card number not found$ "
print7 db 0Dh,0Ah,"Incorrect Password$"
print8 db 0Dh,0Ah,"press 0 to Exit:$"
print9 db 0Dh,0Ah,"press 1 to check another Card: $"
print10 db 0Dh,0Ah,"Please enter a valid password!!!$"

x dw ? ;Card no
y db ? ;Password
i dw 0   ;3lshan yb2a index lel loop

;STATIC DATABASE of 20 customers
DATABASE dw 6053,2387,4700,7448,7603,7356,1287,7878,5792,3571,0096,1567,6050,9012,8845,1358,3250,1111,1357,7893 ;16 bit(word)
PASSWORD db 8,2,6,1,0,3,11,7,14,9,12,5,13,9,15,4,0,12,10,3     ;4 bit(byte)


intro:       ;for the other times
mov ah, 0Eh        ;print new line sequence
mov al, 0Dh
int 10h
mov al, 0Ah
int 10h

lea dx,print0
mov ah, 09h
int 21h
run:
mov ah, 0Eh        ;print new line sequence
mov al, 0Dh
int 10h
mov al, 0Ah
int 10h

lea dx,print1       ;load effective address in DX
mov ah, 09h
int 21h
```



```
call scan_num       ;;takes input until ENTER (subroutine built in)--> stores in

; store card number:
mov x, cx          ;CX because it's 2 bytes(word)

pass:

    lea dx, print2
    mov ah, 09h
    int 21h

    call scan_num


; store password:
mov y, cl    ; only 1 byte so we use CL

cmp y,15
ja wrong


lea dx, print3
mov ah, 09h
int 21h
mov ah, 0Eh        ;print new line sequence
mov al, 0Dh
int 10h
mov al, 0Ah
int 10h

mov cx,20              ;Array siz =20 so ier =20
mov bx,0               ;index on array
mov i,0            ; 3lshan a3raf a-access el password b3den

loopData:   ;loops on card no.
    mov dx,x
    cmp dx,DATABASE[bx]
    je  checkPass
    inc bx
    inc bx       ;increase twice bec memory is byte addressable
    inc i
    loop loopData      ;breaks when CX = 0
    jmp  WrongCard ;3lshan myd5olsh 3la checkPass
```



```
 checkPass:        ;check password
     mov dl,y
     mov bx,i
     cmp dl,PASSWORD[bx]
     je  found
     jmp WrongPass

 found:
       lea dx, print4
       mov ah, 09h
       int 21h
       jmp Continue

 WrongCard:
       lea dx,print6
       mov ah, 09h
       int 21h
       lea dx, print5
       mov ah, 09h
       int 21h
       jmp Continue
 WrongPass:
       lea dx, print7
       mov ah, 09h
       int 21h
       lea dx, print5
       mov ah, 09h
       int 21h
       jmp Continue
 wrong:
   lea dx, print10
   mov ah, 09h
   int 21h
   jmp pass
Continue:
    mov ah, 0Eh        ;print new line sequence
    mov al, 0Dh
    int 10h
    mov al, 0Ah
    int 10h
    lea dx,print8
    mov ah,09h
    int 21h
    lea dx,print9
    mov ah,09h
    int 21h
    mov ah, 0Eh        ;print new line sequence
    mov al, 0Dh
    int 10h
    mov al, 0Ah
    int 10h
    call SCAN_NUM
    cmp cx,1
    je run
    jz  Break

Break:
    mov ah,4ch
    int 21h
```