Computer Organization and Assembly Language

SEMESTER PROJECT

Submitted By:

Rasikh ur Rehman Natiq

SIMPLE Crypto wallet

- **Menu:**

```
.DATA


msg1     BYTE 0AH
         BYTE   "   ---------------------------------------------", 0dh, 0ah
         BYTE   "   ----------  CIMPLE CRYPTO WALLET  ----------", 0dh, 0ah
         BYTE   "   ---------------------------------------------", 0dh, 0ah, 0ah
         BYTE   "   1-> Add Profits", 0dh, 0ah
         BYTE   "   2-> View Profits", 0dh, 0ah
         BYTE   "   3-> Add Coin", 0dh, 0ah
         BYTE   "   4-> View Coin with buying prices", 0dh, 0ah
         BYTE   "   Press any other digit to exit", 0dh, 0ah
         BYTE   "   Choose Your Option : ", 0


ENT_PROF     BYTE " Enter Today's Profit: ",0
VIEW_PROFIT_MSG BYTE 0Ah,"  Viewing rofits: ",0AH, 0DH, 0
ADD_MSG          BYTE " Enter Coin Name & Buying Price to Add: ", 0dh, 0ah,
               "  Separated By Comma:",0
VIEW_COINS_MSG BYTE 0Ah, "  Viewing Coins in Wallet: ",  0dh, 0ah, 0
EXIT_MSG       BYTE 0AH,
                    "   ----------------- ",0dh, 0ah,
                    "   Logging OFF",0dh, 0ah,
                    "   See you again  :')",0dh, 0ah,
                    "   ------------------", 0
```

This code creates all the messages to be displayed in the program. Store in different variables to call them during the program.

• Variables

```
; variables to maniulate

bool           DWORD ?
filehandle     DWORD ?
BUFFER_SIZE = 5000
buffer_mem   BYTE buffer_size DUP (?)
buffer_book  BYTE buffer_size DUP (?)
bytesRead dword 1 dup(0)
PROFIT   DWORD 1
VIEW_PROFIT DWORD 2
ADD_COIN     DWORD 3
VIEW_COIN    DWORD 4
PROFIT_SIZE = 20
PROFIT1 DB PROFIT_SIZE DUP (?)
PROFIT2 DB PROFIT_SIZE DUP (?)
PROFIT3 DB PROFIT_SIZE DUP (?)
PROFIT4 DB PROFIT_SIZE DUP (?)
PROFIT5 DB PROFIT_SIZE DUP (?)
PROFIT6 DB PROFIT_SIZE DUP (?)
NUM_PROFIT DWORD 0
PROFITS DD PROFIT1, PROFIT2, PROFIT3, PROFIT4, PROFIT5, PROFIT6, 0AH, 0DH, 0

COIN_SIZE = 30
COIN1 DB COIN_SIZE DUP (?)
COIN2 DB COIN_SIZE DUP (?)
COIN3 DB COIN_SIZE DUP (?)
COIN4 DB COIN_SIZE DUP (?)
COIN5 DB COIN_SIZE DUP (?)
COIN6 DB COIN_SIZE DUP (?)
NUM_COIN DWORD 0
COINS DD COIN1, COIN2, COIN3, COIN4, COIN5, COIN6, 0AH, 0DH, 0
```

Here, different variables are being created to store coin's name, buying price and profits.

• Main:

```
.CODE

MSG_DISPLAY proto, var: PTR DWORD
STRING_INPUT proto, var1: PTR DWORD
main PROC
    START:
    INVOKE MSG_DISPLAY,addr MSG1
    CALL READINT
    CMP EAX, PROFIT
    JE REG_M
    CMP EAX, VIEW_PROFIT
    JE VIEW_M
    CMP EAX, ADD_COIN
    JE ADD_B
    CMP EAX, VIEW_COIN
    JE VIEW_B
        JMP EXIT_MENU
```

Here it displays the welcome menu and takes input from the user and stores it in the EAX register.

Then check it with predefined options and jump to that specific section.

- **Add Profit:**

```
    REG_M:
        INVOKE MSG_DISPLAY, ADDR ENT_PROF

    MOV ESI, OFFSET PROFITS
    MOV EAX, PROFIT_SIZE
    MUL NUM_PROFIT
    ADD ESI, EAX
    MOV EDX, ESI
    MOV ECX, PROFIT_SIZE
    CALL READSTRING
    INC NUM_PROFIT
        JMP START
```

In this function, the program will take input from the user and store it in the profit array. Number of profits will be incremented after every new input.

Then the program will jump to Start in which it again displays the menu.

- **View Profit:**

```
    VIEW_M:
        INVOKE MSG_DISPLAY, ADDR VIEW_PROFIT_MSG
    MOV ECX, NUM_PROFIT
    cmp ECX, 0
    JE START
    MOV EBX, 0
OUTPUT:
    MOV ESI, OFFSET PROFITS
    MOV EAX, PROFIT_SIZE
    MUL EBX
    ADD ESI, Eax
    MOV EDX, ESI
    CALL WRITESTRING
    INC EBX
    CALL CRLF
LOOP OUTPUT
JMP START
```

Here it will check if there is any profit stored in the list, if there are no profits present then it will jump to Start where it will display the menu again. And if there, is it will go into the loop and prints all of the profits.

- **Add Coin:**

```
    ADD_B:

    INVOKE MSG_DISPLAY, ADDR ADD_MSG
    MOV ESI, OFFSET COINS
    MOV EAX, COIN_SIZE
    MUL NUM_COIN
    ADD ESI, EAX
    MOV EDX, ESI
    MOV ECX, COIN_SIZE
    CALL READSTRING
    INC NUM_COIN

    JMP START
```

In this function, the program will take input from the user and store it in the coins array. Number of coins will be incremented after every new input.

Then the program will jump to Start in which it again displays the menu.

- **View Coin:**

```
    VIEW_B:

    INVOKE MSG_DISPLAY, ADDR VIEW_COINS_MSG
    MOV ECX, NUM_COIN
    cmp ECX, 0
    JE START
    MOV EBX, 0
OUTPUTB:
    MOV ESI, OFFSET COINS
    MOV EAX, COIN_SIZE
    MUL EBX
    ADD ESI, Eax
    MOV EDX, ESI
    CALL WRITESTRING
    INC EBX
    CALL CRLF
LOOP OUTPUTB
        JMP START
```

Here it will check if there is any coin stored in the list, if there are no coins present then it will jump to Start where it will display the menu again. And if there, is it will go into the loop and prints all of the coins.