

1A) Makro ve prosedürler arası fark nedir.

Prosedürler ana program dışında yazılan, ana program içinde call F1 şeklinde çağırılarak belli işlemler yapılan fonk parçaları gibidir. Program işine gömülü değildir. Makrolar ise ana kodumuzun bir parçası olup kod içine gömülürler. Makroya gelince gömülü olan komut çalışır devam eder. Makrolar genelde AX, BX, DX registerleri ile çalışır. Push, pop komutları sıkça kullanılır.

1- gotoxy ch, cl şeklinde makromuz matrisel işlemlerde satır ve sütun üzerinde hareket edilmesini sağlar. [push-pop AX, BX, DX]

2- putc "x" şeklinde karakterin imlecin gösterdiği kısma yazılmasını sağlar
mov ah, 09H
int 21H gibi kesmeler ile bu işlemi halleder

1A) Makrolarda, kod makroya gömülür. Prosedürlerde ise prosedür çalışır, sonrasında diğer kodlar çalışmaya devam eder, yani main prosedüre geri döndürülür ve main "call" dan itibaren kaldığı yerden devam eder.

Makrolar diğer programlama dillerindeki kütüphaneler gibidir. gotoxy ch, cl makrosu ve sonrasında gelen putc ' ' makrosu ile ekranda belli satır, sütun değerlerine (ch, cl) putc içine koyduğumuz değeri yazdırabiliyoruz.

B-) Sembol: Geleneksel programlama dillerindeki değere benzer. Değere göre başlangıç adresini tutmasıdır.

51B) Aslında semboller, geleneksel programlama dillerindeki değişkenlere karşılık gelir. Sembollerin farkı, kullanılan yazmanın başlangıç adresini tutmaktır.

51C)

1. Yanlış. db, equ, -code gibi komutlar direktiflerdir ve direktifler instruction set içerisinde bulunmazlar.

2. Doğru

3. Yanlış. BSEGm a1'de kalan ah yzamacında tutulur.

4. Doğru

5. Doğru

6. Yanlış. AX toplama, çıkarma gibi akümülatör görevi görür. İndis yzmacı olarak BX kullanılabilir.

7. Yanlış. makrodur.

2A) B Grubu

y db 4 1 5 5 8 2 1

Dizideki tek ve çift sayıları toplayan kodlama

L: mov al, y[si] ← Dizinin el. al'ine alına

test al, 01H ← Tek mi çift mi kontrol ediyoruz. [Maskelama]

JZ, ZZ ← 0 olursa çift sayı ZZ dallan

JNZ, YY ← Not zero tek sayı YY dallan

inc si
cmp si, 4
jnz L ← Diziyi parametresini artır, tüm diziyi alına, (loop), diziyi uz. ile karşılaştır.

ZZ: add A, al
jmp D ← Çift sayıları A'da topla D'ye dallan

YY: add B, al
jmp D ← Tek sayıları B'de topla D'ye dallan.

A: 4+2+8
A: 14 (0E)
B: 1+5+5+1
: 12 (0C)

2

52B)

- $\&x$ db #-x yapılmış. $\&x$ db \$-x olmalıydı.
- `mov cx, &x` yanlıştır. `CX` 16 bit, $\&x$ 8 bittir. (adb olarak verimisi)
- `mov al, x[0-si]` yanlıştır. -si kabul edilmez.
- `cmp si, &xy` yanlıştır. $\&xy$ hesaplanmamış
- `endp` → main `endp` olmalı.
- `Ex` proc içinde `ret` eksik.

Yy ve zZ hataları hata değildir !!
Büyük küçük harf aynı olarak algılanır

new	open	examples	save		H	L
01	MOV	AL	, 03H	AX	00	3C
02	MOV	AH	, 02H	BX	00	14
03	MOV	BL	, 14H	CX	00	00
04				DX	00	00
05	MUL	BL				

DX Değişmez

open	examples	save		H	L
MOV	AL	, 33	AX	00	03
MOV	AH	, 0	BX	00	09
MOV	BL	, 9	CX	00	00
MOV	BH	, 0	DX	00	06
DIV	BX		CS	0100	
			IP	001F	

3 Tam -> AX 6 Kalan-> DX

02	mov	al	, 12
03	mov	bl	, 12
04			
05	shl	al	, 3 ; 8 ile carpma
06	shl	bl	, 1 ; 2 ile carpma
07			
08	add	al	, bl
09			

```

02 .model small
03 .data
04 .code
05
06 mov ax , @data
07 mov ds , ax
08
09 mov dl , "Y"
10
11 mov ah , 01H
12 int 21H
13
14 and al , 0FH
15
16 cmp al , 5
17 jg BY
18
19 D:
20 mov ah , 02H
21 int 21H
22
23 HLT
24
25
26 BY:
27     cmp al , 8
28     jl KC
29     jmp D
30
31 KC:
32     mov dl , "D"
33     jmp D

```

6D

2Y

9Y

S3B İçin Farklı Yaklaşımlar

S3B • model small
• data
• code
mov ax, offset
mov ds, ax
mov ax, 0
add AL, 15
SHL AL, 1 ; 30
SHL AL, 1 ; 60
SHL AL, 1 ; 120 $\Rightarrow 12 \times 10$

3B

$$12 \times 10 = 12 \times (8+2) = 12 \times 2^3 + 12 + 12$$

```
mov bl, 12  
mov al, 12  
shl al, 3  
add al, bl  
add al, bl
```

B-) ; $12 = 2^2 + 2^3 \Rightarrow 12 \times 10 \Rightarrow 10 \cdot 4 + 10 \cdot 8$
• model small
• code
mov cl, 10
shl al, 2
mov bl, 10
shl bl, 3
add al, bl

```

01 .MODEL SMALL
02
03 .DATA
04
05     x db 10 dup(0)
06
07 .CODE
08
09 mov ax , @data
10 mov ds , ax
11
12 mov ah , 01H
13
14 L:
15     int 21H
16     mov x[si] , al
17     inc si
18     cmp si , 10
19     jnz L
20
21 mov ah , 02H
22 mov dl , "-"
23 int 21H
24
25 L1:
26     dec si
27     cmp si , 7
28     jl KC
29     jge BY
30 D:
31     int 21H
32
33     cmp si , 0
34     jne L1
35
36
37 HLT
38
39 KC:
40     mov dl , "*"
41     jmp D
42
43 BY:
44     mov dl , x[si]
45     jmp D

```

```

01 .MODEL SMALL
02
03 .DATA
04
05
06 x db 2,0,1,4,      1,0,2,3,      3,2,1,0,      4,2,1,5
07 v db 0,0,0,0
08
09 .CODE
10
11 mov ax , @data
12 mov ds , ax
13
14
15 mov ax , 0
16 mov cx , 0
17 mov si , 0
18 mov di , 0
19
20 L0: mov ch , 0
21
22     L1:
23         mov al , x[si]
24         cmp al , cl
25         jg T
26     D:
27         inc si
28         inc ch
29         cmp ch , 4
30         jne L1
31
32         inc cl
33         inc di
34         cmp di , 4
35         jne L0
36
37 HLT
38
39 T:
40
41     add v[di] , al
42     jmp D
43

```