

Tarih: 02.01.2021



YILDIZ TEKNİK ÜNİVERSİTESİ
ELEKTRİK ELEKTRONİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

ALT SEVİYE PROGRAMLAMA

1.ÖDEV

GRUP-2

Dersi Yürüten: Furkan ÇAKMAK

Engin MEMİŞ

19011040

1119040@std.yildiz.edu.tr

MENÜ EKRANI

Engin Memis
19011040

MENU

- 1.Yeni Dizi Gir!
- 2.Diziyi Gorsellestir!
- 3.Yeni Eleman Ekle:
- 4.Cikis

1.SEÇENEKTE DİZİ ELEMANLARINI GİRİYORUZ

Engin Memis
19011040
Dizinin Boyutunu Giriniz:9
Dizinin Elemanlarini Giriniz:
9
12
6
1
8
17
-2
4
3

DİZİ GÖRSELLEŞTİR SEÇENEĞİNİ SEÇİNCE

Engin Memis
19011040

*****Dizinin Elemanlari*****

9 12 6 1 8 17 -2 4 3

*****Linkler*****

1 5 4 8 0 -1 3 2 7

*****En Kucuk Eleman Indisi*****

6

*****Dizinin Sirali Sekildeki Hali

-2 1 3 4 6 8 9 12 17

YENİ ELEMAN EKLEME

```

      MENU
1.Yeni Dizi Gir?
2.Diziyi Gorsellestir?
3.Yeni Eleman Ekle:
4.Cikis
3

Engin Memis
19011040
Eklenilecek Yeni Elemani Giriniz:
19
```

YENİ ELEMAN EKLEME

```

      MENU
1.Yeni Dizi Gir?
2.Diziyi Gorsellestir?
3.Yeni Eleman Ekle:
4.Cikis
3

Engin Memis
19011040
Eklenilecek Yeni Elemani Giriniz:
-5
```

YENİ ELEMAN EKLEME

```

      MENU
1.Yeni Dizi Gir!
2.Diziyi Gorsellestir!
3.Yeni Eleman Ekle:
4.Cikis
3

Engin Memis
19011040
Eklenilecek Yeni Elemani Giriniz:
10
```

YENİ DİZİYİ GÖRSELLEŞTİRME

```

*****Dizinin Elemanlari*****
9   12   6   1   8   17  -2   4   3   19  -5   10

*****Linkler*****
11   5   4   8   0   9   3   2   7  -1   6   1

*****En Kucuk Eleman Indisi*****
10

*****Dizinin Sirali Sekildeki Hali
-5  -2   1   3   4   6   8   9  10  12  17  19
```

DATA SEGMENT

```

DSEG          SEGMENT PARA 'DATA'
head          DW -1
dizi          DW 100 DUP (?)
adresler      DW 100 DUP (?)
boyut         DW 100
CR            EQU 13
LF            EQU 10
OGRENCI       DB CR,LF,'Engin Memis',LF, '19011040',LF,0
BOSLUK        DB ' ',0
BOSDIZI       DB CR,LF,LF,'*****Dizide Eleman Yok*****',LF,LF,0
ELEMANLAR     DB CR,LF,LF,'*****Dizinin Elemanlari*****',LF,0
ENKUCUK       DB CR,LF,LF,'*****En Kucuk Eleman Indisi*****',LF,0
ADRES         DB CR,LF,LF,'*****Linkler*****',LF,0
SIRALI        DB CR,LF,LF,'*****Dizinin Sirali Sekildeki Hali',LF,0
HATA          DB CR,LF,'Dikkat !!! Sayi vermediniz yeniden giris yapiniz.!!! ',0
MENU          DB LF,LF,' MENU',LF,'1.Yeni Dizi Gir!',LF,'2.Diziyi Gorsellestir!',LF,'3.Yeni Eleman Ekle:',LF,'4.Cikis',LF,0
nAl           DB 'Dizinin Boyutunu Giriniz:',0
ElemanAl      DB 'Dizinin Elemanlarini Giriniz:',LF,0
YeniEleman    DB 'Eklenilecek Yeni Elemani Giriniz:',LF,0
n             DW (?)
secim         DW (?)

DSEG          ENDS

```

```

dongu:
        XOR SI, SI
        MOV AX, OFFSET OGRENCI
        CALL PUT_STR
        MOV AX, OFFSET MENU
        CALL PUT_STR
        CALL GETN
        MOV secim, AX

```

Menüyü ekrana yazdırıp hangi seçeneği gireceğimizi input olarak alıyoruz.

```
CMP secim, 1
JE elemanGir
CMP secim, 2
JE yazdir
CMP secim, 3
JE yeniElemanGir
CMP secim, 4
JE bitis
JMP dongu
```

Inputu karşılaştırıp belli seçeneğe göre dallanma yapıyoruz.

elemanGir:	MOV AX, OFFSET OGRENCI CALL PUT_STR CALL TEMIZLE CALL ELEMANEKLE JMP dongu
------------	--

Diziyi gireceğimiz zaman öncelikle diziyi TEMIZLE yordamı ile sıfırlayıp daha sonra ELEMANEKLE yordamı ile sıra ile eleman ekliyoruz.

yazdir:	MOV AX, OFFSET OGRENCI CALL PUT_STR CALL DIZIYAZ JMP dongu
---------	---

Diziyi görselleştirme yapacağımız zaman DIZIYAZ yordamını çağırıyoruz.

yeniElemanGir:	MOV AX, OFFSET OGRENCI CALL PUT_STR MOV AX, OFFSET YeniEleman CALL PUT_STR CALL GETN MOV dizi[SI], 0 MOV adresler[SI], -1 ADD n, 1 CALL LINK ADD SI, 2 JMP dongu
----------------	--

Diziye yeni eleman eklemek istediğimizde adresler dizisinde ondan sonraki gelecek elemanın indisini -1 koyuyoruz. Daha sonra eleman sayısını 1 arttırıp LINK yordamını çağırıyoruz.

YORDAMLAR

ELEMANEKLE

PROC NEAR

PUSH AX

PUSH CX

MOV AX, OFFSET nAl

CALL PUT_STR

CALL GETN

MOV n, AX

XOR SI, SI

MOV CX, n

MOV AX, OFFSET ElemanAl

CALL PUT_STR

CALL GETN

CALL LINK

ADD SI, 2

LOOP L1

POP CX

POP AX

RET

ENDP

L1:

ELEMANEKLE

Yeni dizi eklenileceği zaman ilk olarak ‘n’ değişkenine dizi boyutunu alıp daha sonra bir döngüde elemanları aldıktan sonra LINK yordamını çağırıyoruz.

LINK YORDAMI

1. KISIM

LINK

PROC NEAR

PUSH DX

PUSH SI

PUSH AX

PUSH BX

PUSH DI

PUSH CX

;DİZİDE HİÇ ELEMAN YOK İSE

CMP head, -1

JNE ikinci

MOV dizi[SI], AX

MOV adresler[SI], -1

MOV head, SI

JMP son

Eleman eklenilecek dizide hiç eleman yok ise yani 'head' değişkeni -1 ise diziyeye elemanı ekledikten sonra da adresler dizisinde son eleman olduğu için -1 koyuyoruz ve dizinin en küçük elemanını tutan 'head' değişkenine şu anki elemanın indisini yazıyoruz.

2. KISIM

```
ikinci:      ;GELEN ELEMAN EN BAŞTAKİ ELEMANDAN KÜÇÜK İSE
MOV DI, head
CMP AX, dizi[DI]
JGE ucuncu
MOV dizi[SI], AX
MOV BX, head
MOV adresler[SI], BX
MOV head, SI
JMP son
```

Yeni eklenilecek eleman dizinin en küçük elemanından küçük ise yeni eklenilen eleman diziyeye eklenildikten sonra 'BX' değişkenine 'head' değişkenini atıyoruz. Daha sonra adresler dizisinde yeni elemanın olduğu indise 'head' değişkeninin değerinin tutan BX'i atıyoruz. 'head' değişkenini de yeni elemanı tutacak şekilde güncelliyoruz.

3. KISIM

```
ucuncu:      ;DİZİDE 1DEN FAZLA ELEMAN VARSA YERİNİ BULUP EKLENECEK
MOV dizi[SI], AX
MOV DI, head
bas:         CMP adresler[DI], -1
             JE bulundu
             MOV BX, adresler[DI]
             CMP AX, dizi[BX]
             JL bulundu
             MOV DI, adresler[DI]
             JMP bas
```

Yeni gelen elemanı diziyeye ekledikten sonra en küçük elemanı tutan 'head' değişkenini DI'ya atıyoruz. 'head' değişkeninin tuttuğu elemandan sonra gelecek eleman yok ise gelen eleman son eleman olacağı için yerini bulmuş oluyoruz. Daha sonra eleman var ise yeni gelen elemanımız o sonraki elemandan küçük mü diye karşılaştırma yapıyoruz eğer küçük ise eklenilecek yerini tekrar bulmuş oluyor. Eğer küçük değil ise DI'yı bir sonraki elemanı gösterecek olarak güncelliyoruz ve döngü olarak yerini bulana kadar devam ediyoruz.

4. KISIM

```

;BULUNAN YERE YENİ ELEMANIN LİNKLENMESİ

bulundu:    MOV CX, adresler[DI]
            MOV adresler[SI], CX
            MOV adresler[DI], SI
            JMP son

son:        POP CX
            POP DI
            POP BX
            POP AX
            POP SI
            POP DX
            RET

LINK       ENDP

```

Yeni gelen eleman yerini bulduktan sonra eklenileceği yerden bir önceki elemanın gösterdiği yeri artık yeni elemanımız gösterecek şekilde güncelledikten sonra eklenilecek olan yerden bir önceki eleman da yeni eklenilecek elemanı gösterecek şekilde güncelliyoruz.

TEMİZLE YORDAMI

TEMİZLE		PROC NEAR
		PUSH CX
		PUSH SI
		MOV CX, boyut
		XOR SI, SI
		MOV head, -1
temizlenmedi:		MOV dizi[SI], 0
		CMP adresler[SI], -1
		JNE temizlendi
		ADD SI, 2
		LOOP temizlenmedi
temizlendi:		POP SI
		POP CX
		RET
TEMİZLE		ENDP

Tekrardan yeni bir dizi girmek istediğimizde elemanların olduğu diziyi ve indislerin olduğu adresler dizisini sıfırlıyoruz. En küçük elemanı tutan 'head' değişkenini de -1 olarak değiştiriyoruz.

DIZIYAZ YORDAMI

1.KISIM

DIZIYAZ		PROC NEAR
		PUSH SI
		PUSH CX
		CMP head, -1
		JE eleman_yok_dizi
		MOV AX, OFFSET ELEMANLAR
		CALL PUT_STR
eleman_yok_dizi:		MOV AX, OFFSET BOSDIZI
		CALL PUT_STR

Dizide hiç eleman yok ise ‘Dizide Eleman Yok’ şeklinde ekrana yazı yazdırıyoruz.

2.KISIM

dizi_devam:	<pre>XOR SI, SI MOV CX, n MOV AX, dizi[SI] CALL PUTN MOV AX, OFFSET BOSLUK CALL PUT_STR ADD SI, 2 LOOP dizi_devam</pre>
-------------	---

Dizinin elemanlarını baştan sona yazdırıyoruz.

3.KISIM

adres_devam:	<pre>XOR SI, SI MOV CX, n MOV AX, OFFSET ADRES CALL PUT_STR MOV AX, adresler[SI] SAR AX, 1 CALL PUTN MOV AX, OFFSET BOSLUK CALL PUT_STR ADD SI, 2 LOOP adres_devam</pre>
--------------	--

İndisleri tutan adresler dizisi baştan sona yazdırıyoruz. Yazdırırken elemanları ikiye bölerek yazdırmamızın sebebi işlemler yapılırken dizi WORD tipinde olduğu için 0-2-4 vb. şekilde ikinin katları halinde olduğundan dolayıdır.

4.KISIM

```
MOV AX, OFFSET ENKUCUK
CALL PUT_STR
MOV AX, head
SAR AX, 1
CALL PUTN
```

Dizinin en küçük elemanını tutan 'head' değişkenini yazdırıyoruz.

LINKLIYAZ YORDAMI

LINKLIYAZ	PROC NEAR
	PUSH SI
	PUSH DI
	CMP head, -1
	JE eleman_yok
	MOV AX, OFFSET SIRALI
	CALL PUT_STR
	XOR SI, SI
	MOV SI, head
devam_dizi:	MOV AX, dizi[SI]
	CALL PUTN
	MOV AX, OFFSET BOSLUK
	CALL PUT_STR
	MOV SI, adresler[SI]
	CMP SI, -1
	JNE devam_dizi
	JMP yaz_son
eleman_yok:	MOV AX, OFFSET BOSDIZI
	CALL PUT_STR
yaz_son:	POP DI
	POP SI
LINKLIYAZ	RET
	ENDP

Diziyi küçükten büyüğe yazdırmak için en başta ‘head’ değişkeninin tuttuğu en küçük elemandan başlayarak ekrana yazdırıp daha sonra o elemandan sonra gelen elemanın indisini tutan adresler dizisinden bir sonraki elemanın indisini yeni indisimiz olarak güncelleyerek döngü şeklinde bütün elemanları linkler sayesinde küçükten büyüğe ekrana yazdırıyoruz.