

Türkiye Cumhuriyeti
Yıldız Teknik Üniversitesi
Bilgisayar Mühendisliği Bölümü



Alt Seviye Programlama Dersi 1. Ödevi

Öğrenci Adı Soyadı: Anıl Kutay Uçan

Öğrenci Numarası: 20011025

Öğrenci e-posta: 11120025@std.yildiz.edu.tr

Ders: BLM2021 Alt Seviye Programlama

Ders Yürütücüsü

Öğr.Gör. Furkan Çakmak

İçerik:

- Ödevde İstenilenler
- Not-Bilgilendirme
- Ana Menü'nün Açıklanması
- 1. Alt Menü'nün Açıklanması
- 2. Alt Menü'nün Açıklanması
- 3. Alt Menü'nün Açıklanması
- Kullanılan Fonksiyonlar ve Açıklamaları
- Kaynakça

Ödevde İstenilenler:

Bu ödevde aşağıda istenilenleri gerçekleştiren bir konsol uygulaması yazmanız beklenmektedir. Ödeviniz bir ana ekran menüsünden ve aşağıda anlatılan 3 fonksiyonu barındıran alt menülerden oluşmalıdır. Menü tasarımıınız ekrana yazdırılacak karakterlerle basitçe gerçekleştirilebilmesi, alt menü seçimleri de programa girdi olarak verilen karakterlerle (örnek 1, 2, 3) belirlenmelidir. Menü tasarımıınızın bir yerinde (tüm alt menülerde de görünmek kaydıyla) öğrenci numaranız ve isminiz yazmalıdır.

Alt Menü 1: Kullanıcıdan alınan n değeri ve bu n değeri kadar dizi elemanlarının sisteme girilmesinden oluştur.

a) Dizi girildiği anda bir yordam ile bu diziyi Linkli Liste veri yapısında oluşturacak şekilde linkleri oluşturunuz.

Alt Menü 2: Kullanıcı tarafından girilen dizinin değerlerinin ve linklerinin anlaşılabilir bir şekilde görselleştirilmesini gerçekleştirir.

Alt Menü :. Linkli listeye kullanıcının girmek istediği yeni bir veri eklenmesi için kullanılır. Veri eklenmesi gerçekleştirdikten sonra linklerde ilgili düzenlemeler yapılmalıdır. Bu işleminden sonra 2 numaralı alt menü kullanıldığında yeni eklenen değer ve güncellenmiş linkler görülmelidir.

Not-Bilgilendirme:

Bütün yazılacak mesajlar data segmentin içinde tanımlanmıştır.

Bütün menülerde, ekranın üst kısmında öğrencinin adı, soyadı ve numarası yazdırılmıştır.

Yapılan Proje exe tipinde yazılmıştır. 1 ana menü ve 3 tane alt menü'den oluşur. Kullanıcıdan alınan girdinin sonlanması için enter'a basılması gerekmektedir.

Programdan çıkmak için listeye veri ekleme kısmı hariç herhangi bir yerde q veya Q harflerine basmanız yeterlidir. Sonradan enter'a basmaya gerek yoktur.

Ana Yordam far tipindeki START yordamıdır. Burada sürekli bir döngü içinde near tipindeki MAINMENU yordamı çağırılır. Döngüden çıkmak için kullanıcıdan q veya Q harfleri girilmelidir. Bu harfler girildiğinde buradaki sonsuz döngü bozulur ve retf ile kontrol devredilir.

Ana Menü:

Ana Menü MAINMENU adlı near tipindeki bir yordamın çağırılması ile çalışmaya başlar. Ana Menü'nün üst kısmında öğrencinin adı soyadı ve numarası bulunmaktadır. Bu menüde öncelikle, hangi fonksiyonun nasıl çalıştırılacağı gösterilmektedir.

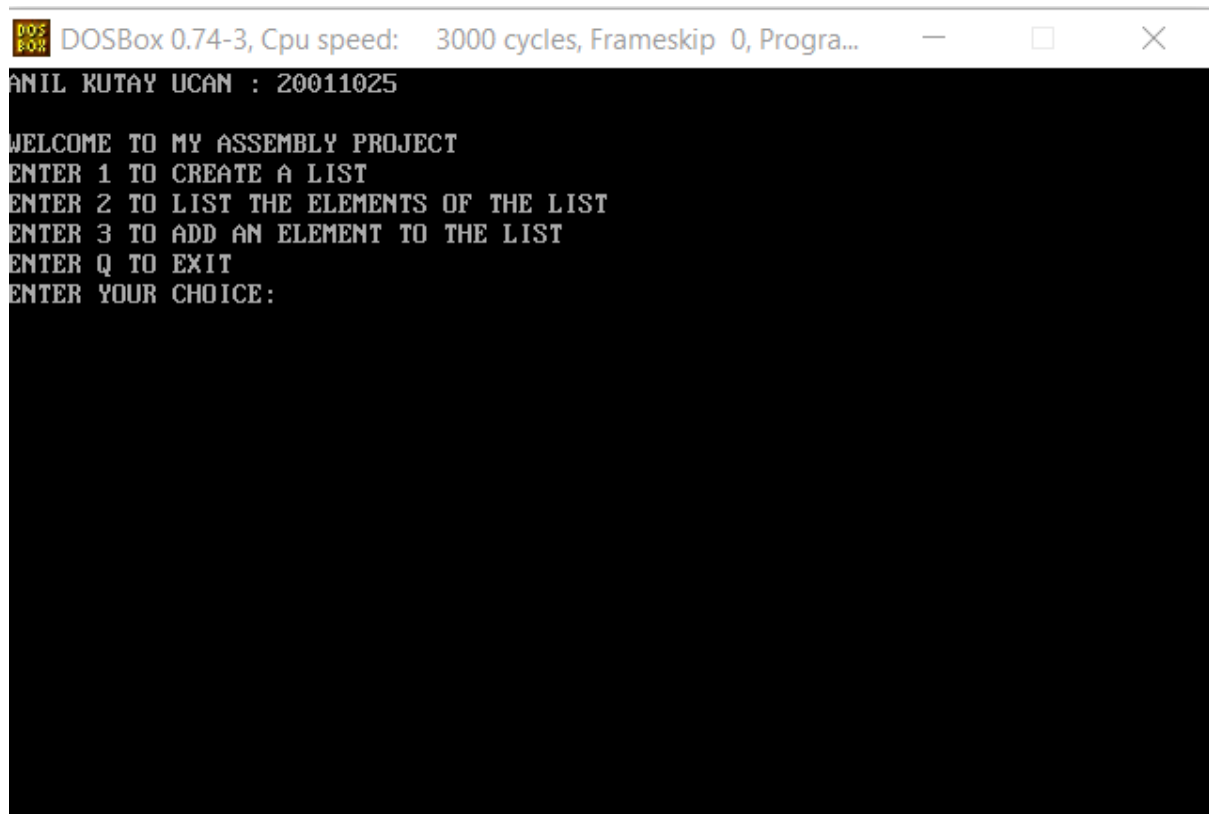
Eğer 1'e basılırsa Alt Menü 1'in bulunduğu yordam çağırılır (SUBMENUONE) ve bu yordam sayesinde kullanıcının hızlı bir şekilde liste yapması sağlanır.

Eğer 2'ye basılırsa, 2. alt menüye geçilir. Bu da SUBMENUTWO yordamının çağırılmasıyla yapılır. Burada yapılan listenin elemanları kullanıcıya gösterilir.

Eğer 3'e basılırsa, 3. alt menüye geçilir. Bu SUBMENUTHREE yordamının çağırılması ile yapılır. Burada listeye bir tane eleman eklenmesi sağlanır.

Eğer Q veya q'ya basılırsa işlem durdurulur ve ana yordamdan retf komutu ile çıkar.

Eğer farklı herhangi bir sayıya basılırsa, yeniden ana menü gösterilir. Eğer sayıya basılmadıysa, ve basılan harf q değilse hata mesajı gösterilir.

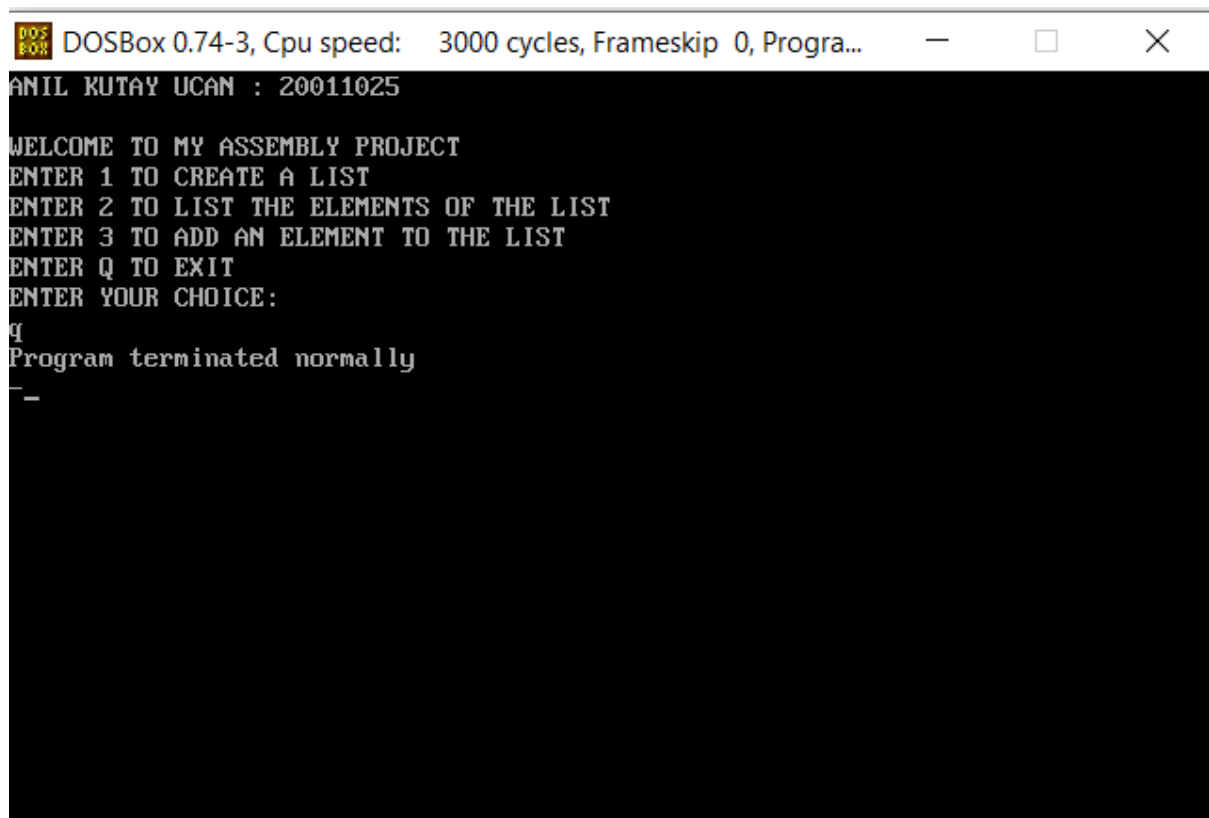


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
ANIL KUTAY UCAN : 20011025

WELCOME TO MY ASSEMBLY PROJECT
ENTER 1 TO CREATE A LIST
ENTER 2 TO LIST THE ELEMENTS OF THE LIST
ENTER 3 TO ADD AN ELEMENT TO THE LIST
ENTER Q TO EXIT
ENTER YOUR CHOICE:
```

Ana Menü



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...

```
ANIL KUTAY UCAN : 20011025

WELCOME TO MY ASSEMBLY PROJECT
ENTER 1 TO CREATE A LIST
ENTER 2 TO LIST THE ELEMENTS OF THE LIST
ENTER 3 TO ADD AN ELEMENT TO THE LIST
ENTER Q TO EXIT
ENTER YOUR CHOICE:
q
Program terminated normally
_
```

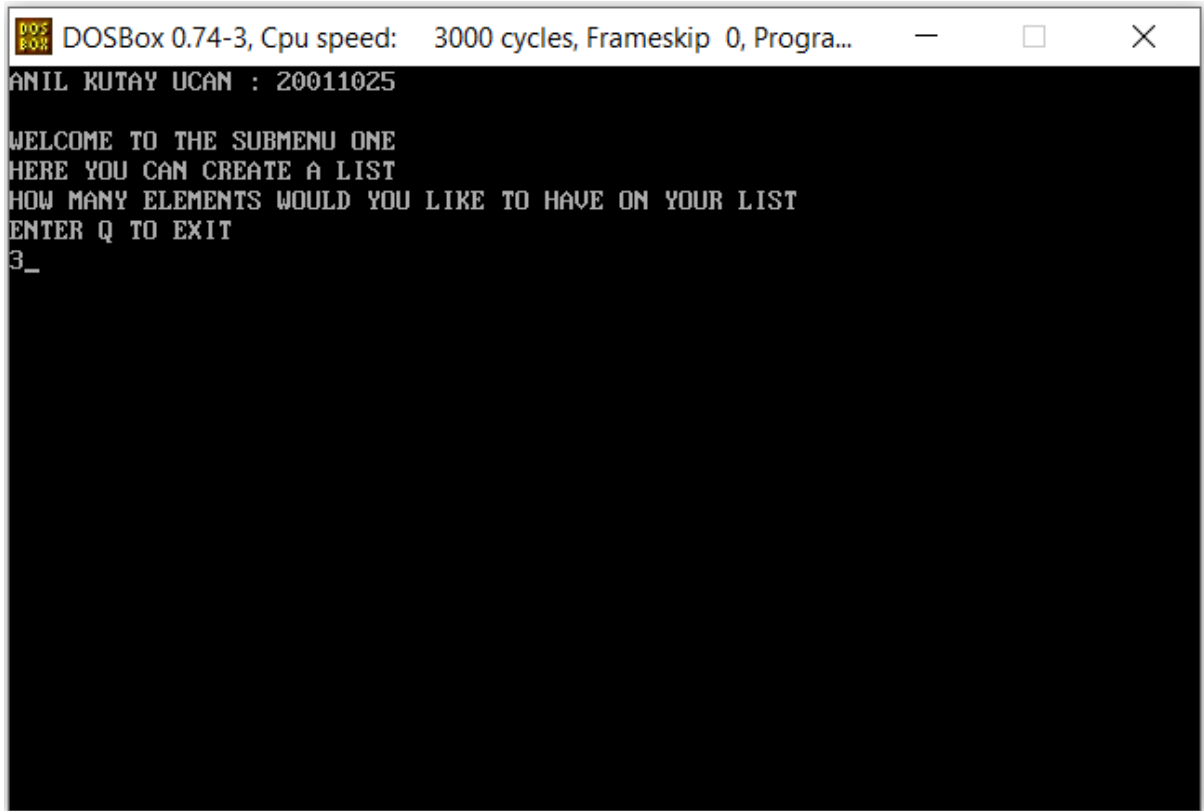
Ana Menü: q girdisi verildiğinde program'dan çıkılıyor.

Alt Menü 1:

Alt Menü 1 SUBMENUONE adlı near tipindeki yordamın çağrılması ile çalışmaya başlar. Bu menünün amacı kullanıcının istediği boyutta ve kullanıcıdan alınan verilerle bir liste oluşturmaktır. Alt Menü 1'in üst kısmında öğrencinin adı soyadı ve numarası bulunmaktadır. Menü öncelikle kendi işlevini ve kullanıcıdan ne için girdi istediğini yazdırır.

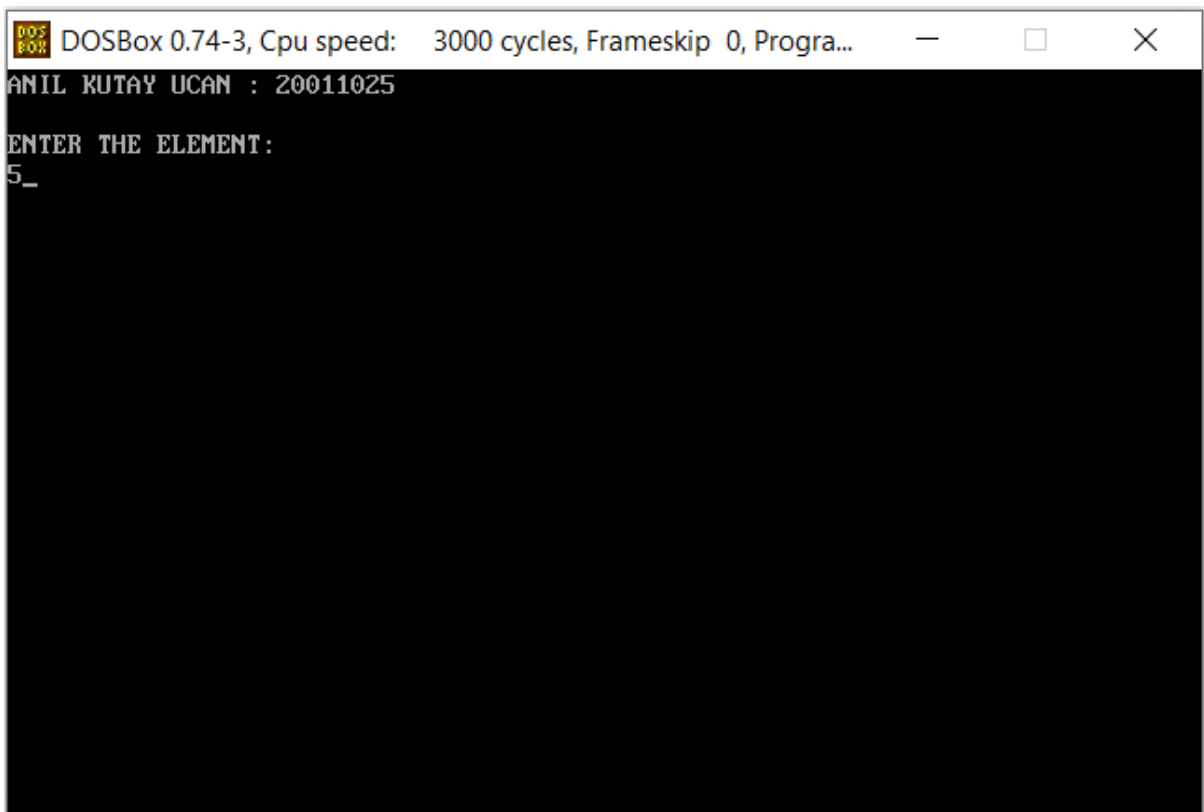
Bu menü kullanıcıya kaç elemanlı bir liste oluşturmak istediğini sorar. Sonra kullanıcı tarafından verilen girdi kadar, kullanıcıdan yeni bir eleman girmesini ister. Bu işlem loop kullanılarak yapılmıştır. Her yeni eleman girildiğinde onu listeye koyar ve linkleri yeni baştan düzenler. Bu işlemler ADDELEMENT yordamı çağrılarak yapılmaktadır.

Burada da q veya Q'ya basıldığında program durdurulur ve ana yordamdan retf komutu ile çıkılır.

A screenshot of a DOSBox window. The title bar reads "DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...". The window contains a text-based menu. The text is as follows:

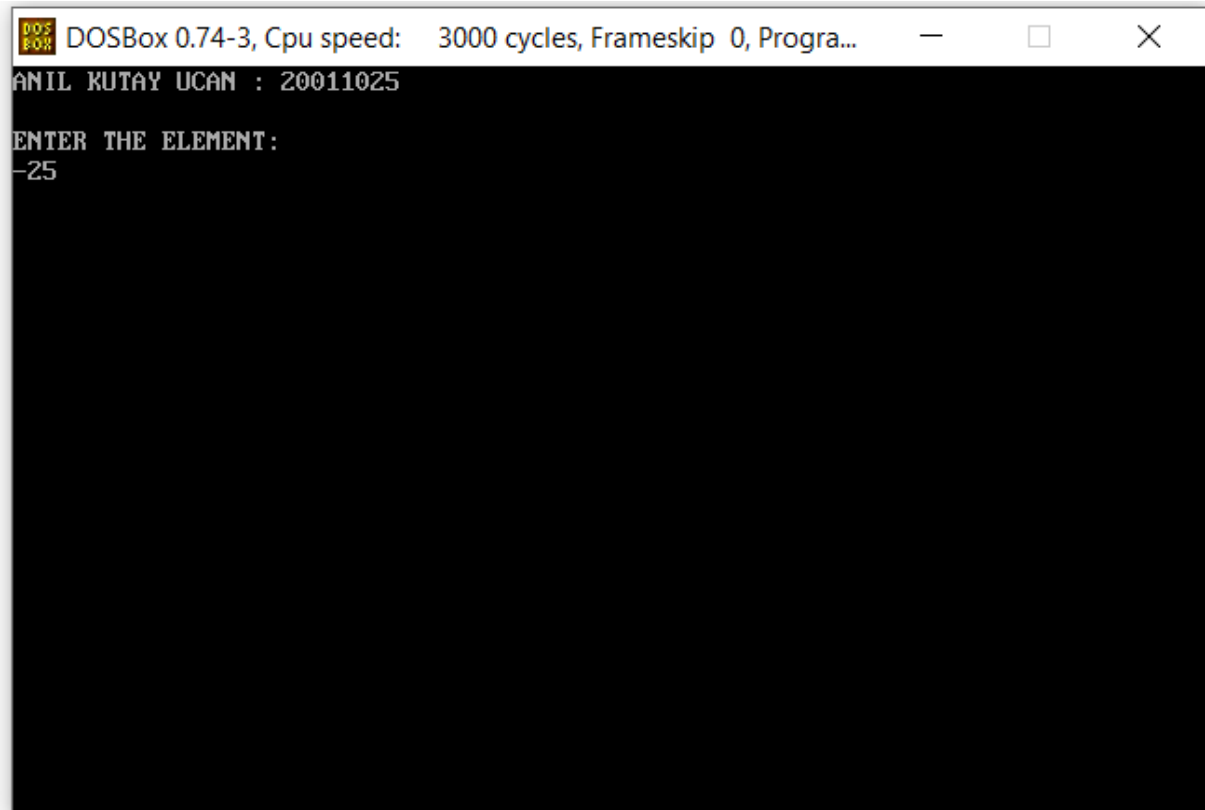
```
ANIL KUTAY UCAN : 20011025  
  
WELCOME TO THE SUBMENU ONE  
HERE YOU CAN CREATE A LIST  
HOW MANY ELEMENTS WOULD YOU LIKE TO HAVE ON YOUR LIST  
ENTER Q TO EXIT  
3_
```

Alt Menü 1'in İlk Sorusu: Cevap olarak 3 verildiği için 3 tane eleman alacak.

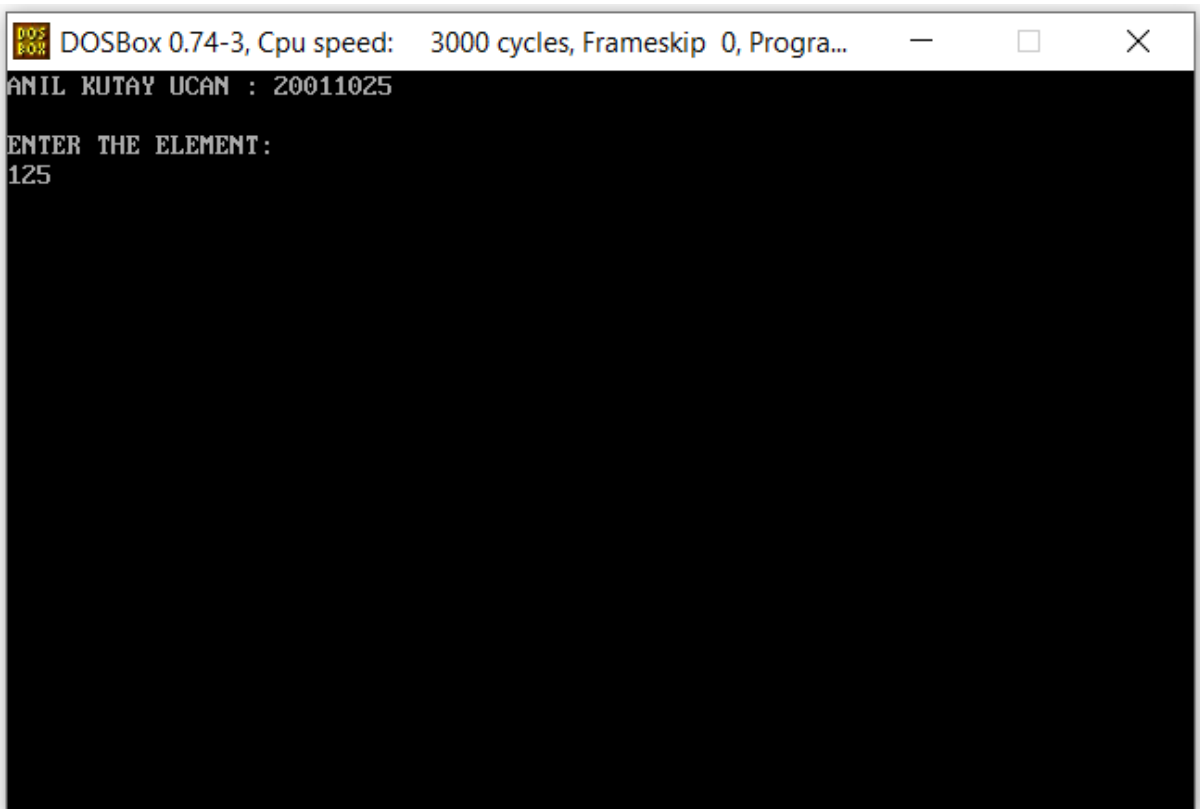
A screenshot of a DOSBox window. The title bar reads "DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...". The window contains a text-based prompt for entering an element. The text is as follows:

```
ANIL KUTAY UCAN : 20011025  
  
ENTER THE ELEMENT:  
5_
```

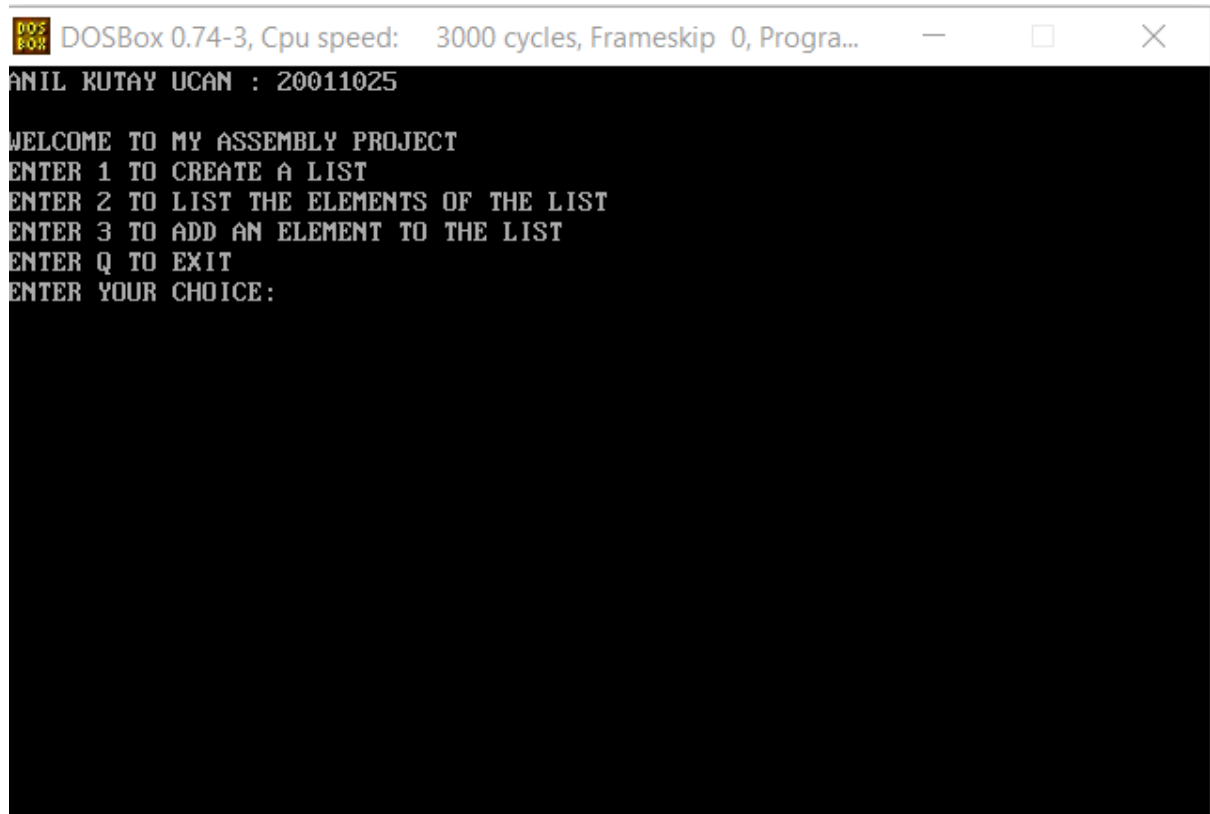
Alt Menü 1'in İkinci Sorusu: Burada hangi elemanın listeye eklenmesini istersek onu yazıyoruz.



Alt Menü 1'in İkinci Sorusu: 3 tane eleman girmek istediğimizi söyledik. Bu yüzden burada 2. elemanı alıyor.



Alt Menü 1'in İkinci Sorusu: 3 tane eleman girmek istediğimizi söyledik. Bu yüzden burada 3. elemanı alıyor.

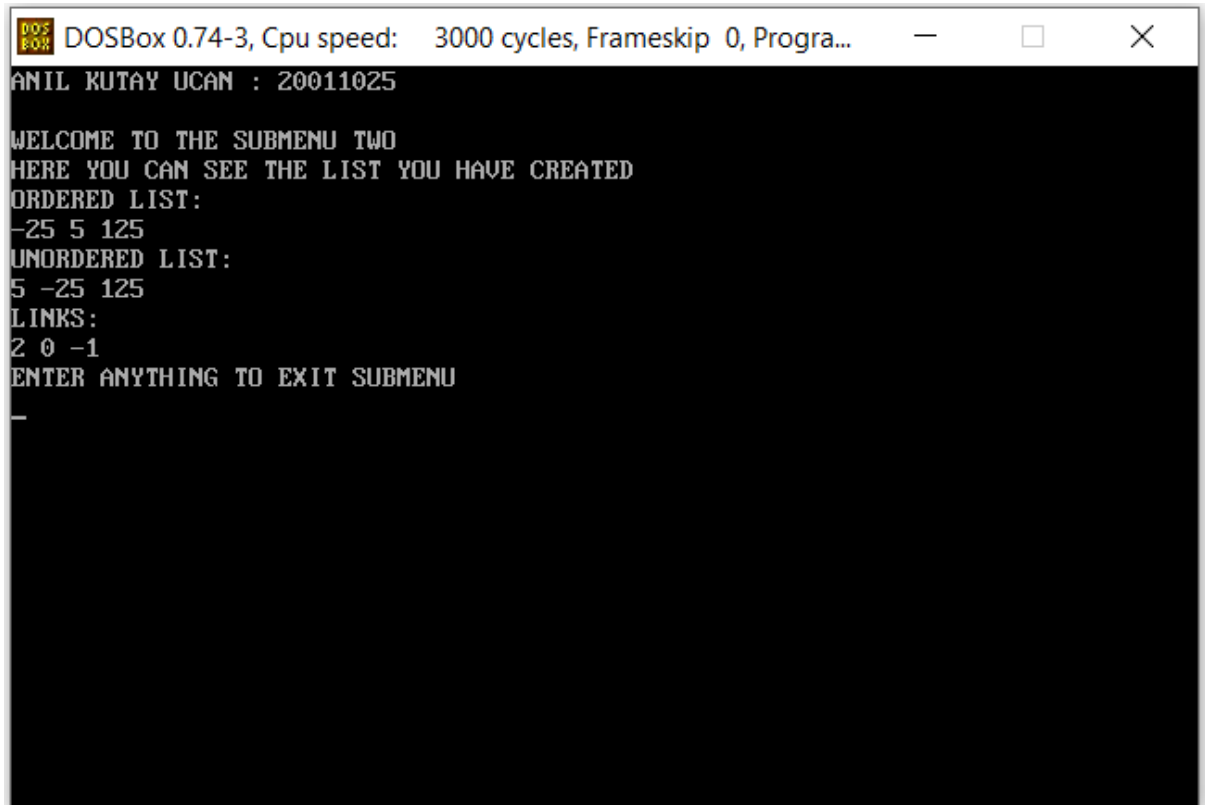
A screenshot of a DOSBox window. The title bar reads "DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...". The window contains a text-based menu for an assembly project. The text is as follows:

```
ANIL KUTAY UCAN : 20011025  
  
WELCOME TO MY ASSEMBLY PROJECT  
ENTER 1 TO CREATE A LIST  
ENTER 2 TO LIST THE ELEMENTS OF THE LIST  
ENTER 3 TO ADD AN ELEMENT TO THE LIST  
ENTER Q TO EXIT  
ENTER YOUR CHOICE:
```

Ana Menü: Bütün işlemler bittikten sonra ana menüye geri dönüyoruz.

Alt Menü 2:

Alt Menü 2 SUBMENUTWO adlı near tipindeki yordamın çağrılması ile çalışmaya başlar. Bu menünün amacı kullanıcıdan alınan verilerle oluşturulan linkleri ve listeyi sıralı ve sırasız bir şekilde göstermektir. Alt Menü 2'in üst kısmında öğrencinin adı soyadı ve numarası bulunmaktadır. Menü öncelikle kendi işlevini yazdırır, sonra da listeyi ve linkleri yazdırır. En sonda da kullanıcının herhangi bir harf girmesini veya enter'a basmasını bekler. Kullanıcı enter'a basana kadar aynı ekranda durur. Kullanıcı girdi sağlarsa, ana menüye geri geçer.



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
ANIL KUTAY UCAN : 20011025
WELCOME TO THE SUBMENU TWO
HERE YOU CAN SEE THE LIST YOU HAVE CREATED
ORDERED LIST:
-25 5 125
UNORDERED LIST:
5 -25 125
LINKS:
2 0 -1
ENTER ANYTHING TO EXIT SUBMENU
-
```

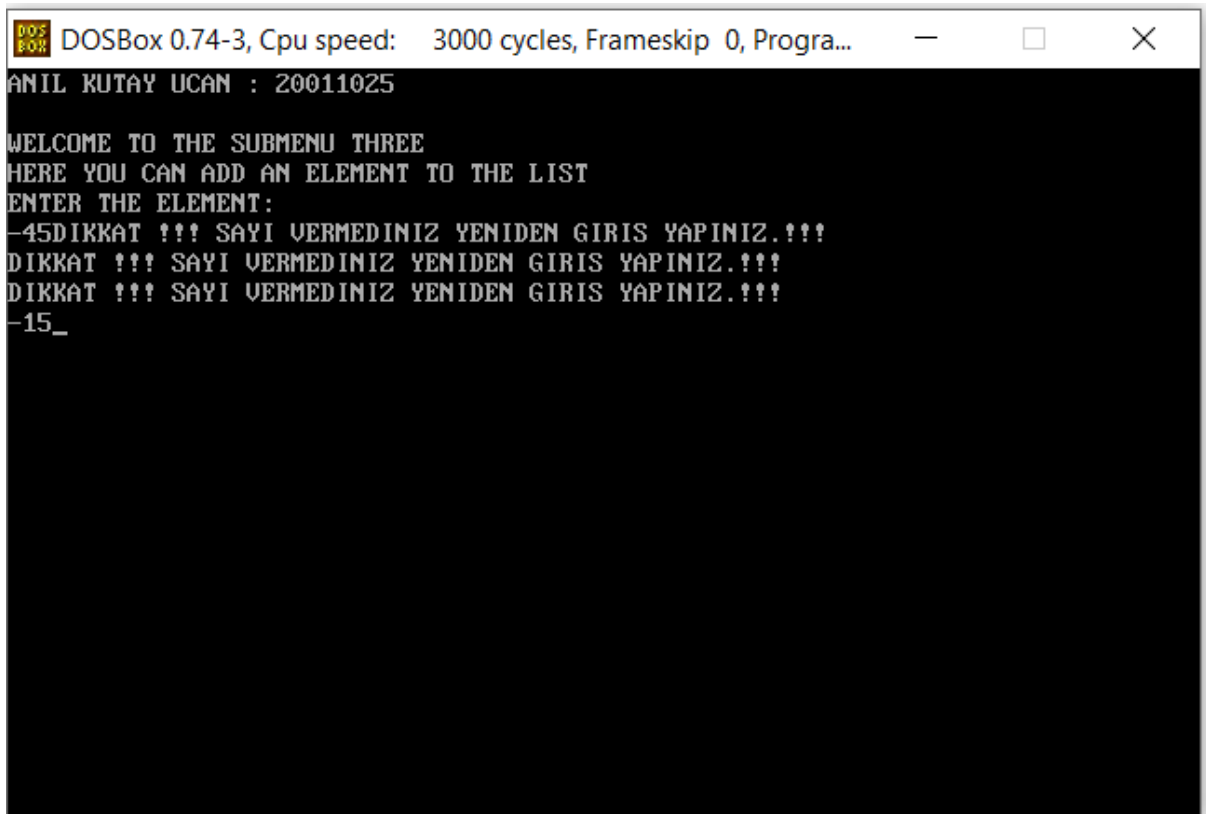
Alt Menü 2: Liste sıralı ve sırasız bir şekilde yazdırılır. Altında da linkler yazdırılır.

Alt Menü 3:

Alt Menü 3 SUBMENU THREE adlı near tipindeki yordamın çağrılması ile çalışmaya başlar. Bu menünün amacı kullanıcıdan bir eleman alıp listenin sonuna eklemek ve linkleri ona göre düzenlemektir. Bu işlemler ADDELEMENT yordamı çağrılarak yapılmaktadır.

Alt Menü 3'ün üst kısmında öğrencinin adı soyadı ve numarası bulunmaktadır. Menü öncelikle kendi işlevini yazdırır, sonra da kullanıcıdan bir eleman girmesini ister. Eğer girdi yanlış verildiyse hata mesajı verilir ve kullanıcının yeni girdi vermesi beklenir.

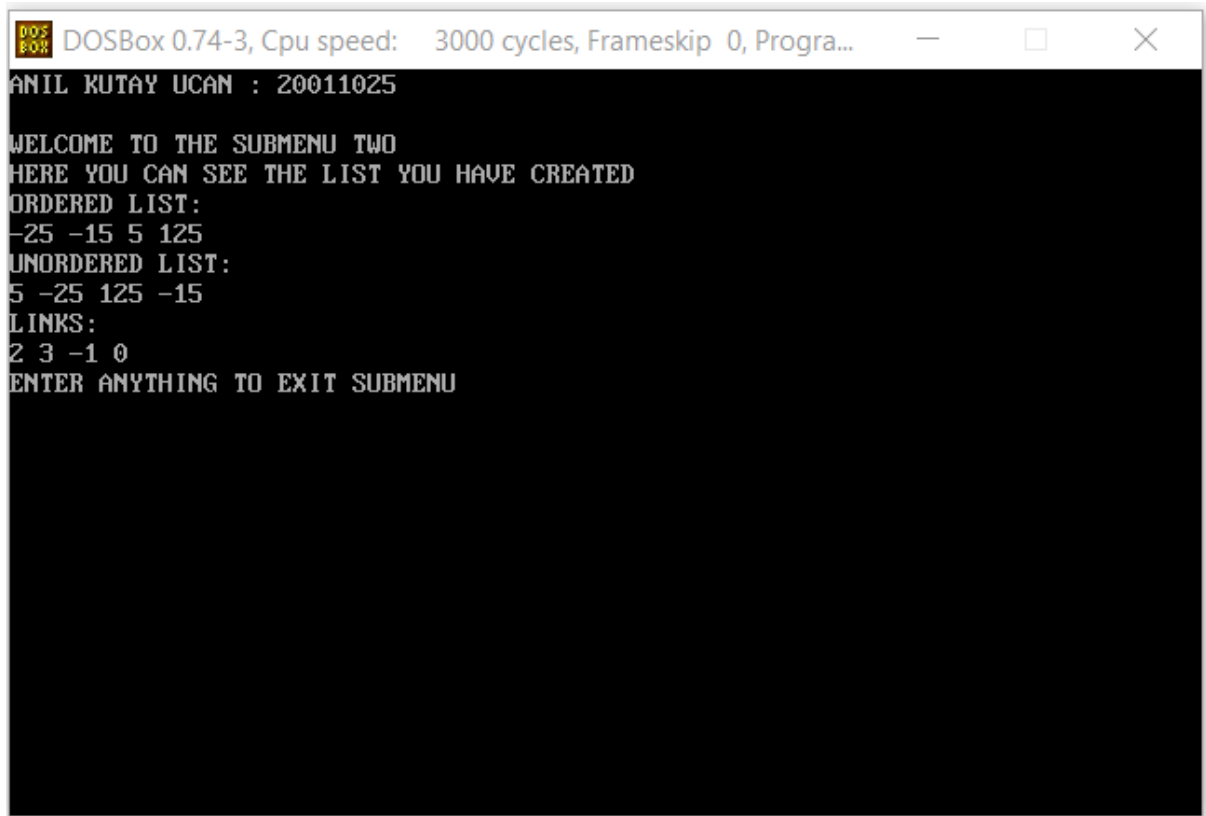
Alt Menü 3 kullanıldıktan sonra tekrar alt menü 2 çağırılırsa, listeye yeni girilen verinin eklendiği ve linklerin ona göre düzenlendiği görülür.



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
ANIL KUTAY UCAN : 20011025

WELCOME TO THE SUBMENU THREE
HERE YOU CAN ADD AN ELEMENT TO THE LIST
ENTER THE ELEMENT:
-45DİKKAT !!! SAYI VERMEDİNİZ YENİDEN GİRİŞ YAPINIZ.!!!
DİKKAT !!! SAYI VERMEDİNİZ YENİDEN GİRİŞ YAPINIZ.!!!
DİKKAT !!! SAYI VERMEDİNİZ YENİDEN GİRİŞ YAPINIZ.!!!
-15_
```

Alt Menü 3: Liste sıralı ve sırasız bir şekilde yazdırılır. Altında da linkler yazdırılır. Yanlış bir girdi verilirse yukarıda görülen hata mesajı gözüktür.

A screenshot of a DOSBox window. The title bar reads "DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...". The window contains a text-based menu system. The text is as follows:

```
ANIL KUTAY UCAN : 20011025  
  
WELCOME TO THE SUBMENU TWO  
HERE YOU CAN SEE THE LIST YOU HAVE CREATED  
ORDERED LIST:  
-25 -15 5 125  
UNORDERED LIST:  
5 -25 125 -15  
LINKS:  
2 3 -1 0  
ENTER ANYTHING TO EXIT SUBMENU
```

Alt Menü 2: Alt Menü 3 kullanıldıktan sonra bir daha Alt Menü 2 çağırılırsa.

Kullanılan Fonksiyonlar ve Açıklamaları:

PUTN:

Ekranı yazı yazdırmak için iki tür fonksiyon kullanılmıştır. İlk PUTN adlı yordamda bulunmaktadır. Bu fonksiyon hafızada hex olarak saklanan sayıyı ekrana ondalık biçimde bastırıyor ve sonuna bir boşluk koyuyor. Bu fonksiyon <https://github.com/80x86adtcebi/bookexamples> adresinden alınmıştır ve isteğe bağlı olarak biraz düzenlenmiştir.

PRINT:

İkinci türü ise ekrana string yazdırmak için kullanılmıştır ve PRINT yordamındadır. Sonu \$ ile biten stringleri yazdırmak için kullanılmaktadır. DX'in içine, yazdırılacak string'in adresi konulur. Sonra ah 09h iken int 21 çağırılır. En son da bir alt satıra geçmek için INCROW yordamı çağırılır. https://tr.wikibooks.org/wiki/X86_Assembly/%C3%96nemli_i nt_servisleri burada açıklanan tarife uygun olarak yapılmıştır.

Bunu Yapan Kod Parçası:

```
LEA DX, text  
MOV AH, 09H  
INT 21H  
CALL INCROW
```

CLEAR:

Ekranı temizlemek için CLEAR yordamı çağırılmaktadır. Bu fonksiyon <https://github.com/80x86adtcebi/bookexamples> sitesinden alınmıştır. Orijinal olarak makro tanımlıdır. Bu yordamın içinden ekstra olarak RESETROW adlı yordam çağırılır ve en son ekranın başına öğrencinin adının, soyadının ve numarasının yazdırılması sağlanır.

RESETROW:

Bu yordam satır başını ekranın en üst noktasına alır ve CURSORPLACE adlı değişkeni 0 yapar. Bu işlemi yapmak için dh, 0'a eşitlenir ve ah 02h'dayken int 10 çağırılır. Bu sayede ekrana bir şey yazdırmak istenildiğinde ekranın en başından başlanır. Bu işlem [https://tr.wikibooks.org/wiki/X86_Assembly/%C3%96nemli i nt_servisleri](https://tr.wikibooks.org/wiki/X86_Assembly/%C3%96nemli_i nt_servisleri) sitesinde açıklanan tarife uygun olarak yapılmıştır.

Bunu Yapan Kod Parçası:

```
MOV BYTE PTR CURSORPLACE, 0
MOV DH, 0
MOV DL, 0
MOV BH, 0

MOV AH, 02H
INT 10H
```

INCROW:

Ekrana bir şey yazdırıldıktan sonra bir alt satıra geçmek için INCROW yordamı çağırılmaktadır.

Bu yordam CURSORPLACE adlı değişkende tutulan hangi satırda olduğu bilgisini bir arttırır ve dh yazmacının içine koyar.

Ah yazmacı 02h 'da iken int 10 çağırıldığında dh'ın içindeki değer yeni satır başı olur ve yazdırma oradan devam eder.

Böylece bir satır aşağı geçilmiş olur. Bu işlem

https://tr.wikibooks.org/wiki/X86_Assembly/%C3%96nemli_int_servisleri sitesinde açıklanan tarife uygun olarak yapılmıştır.

Bunu Yapan Kod Parçası:

```
INC BYTE PTR CURSORPLACE
MOV DH, BYTE PTR CURSORPLACE
MOV DL, 0
MOV BH, 0
```

```
MOV AH, 02H
INT 10H
```

ADDELEMENT:

Listeye bir şeyler eklenmesi ve linklerin düzenlenmesi için ADDELEMENT yordamı çağrılır. Bu yordamın işleyişi bir kaç parçadan oluşur.

İlk başta listeye kullanıcıdan alınan veri eklenir. Bu verinin eklenmesi aşağıda anlatılmıştır:

INDEX değişkeninde tutulan ve listenin hangi indexinde olduğumuzu belirten değer bx register'ına alınır. Sonra kullanıcıdan alınan ve choice adlı değişkende tutulan veri ax registerına alınır. Listenin Bx indexli bölgesine kullanıcıdan alınan veri koyulur ve index, liste word tipinde olduğundan dolayı 2 arttırılır.

Bunu Yapan Kod Parçası:

```
MOV BX, WORD PTR INDEX  
MOV AX, WORD PTR CHOICE  
MOV WORD PTR LIST[BX], AX  
ADD WORD PTR INDEX, 2
```

Sonra Linklerin düzenlenmesi gerekmektedir. Bir kaç muhtemel durum bulunmaktadır. Listede sadece bir eleman olabilir, elemanın sırası listenin başında, ortasında ya da sonunda olabilir.

İlk başta dizide 1’den fazla eleman olup olmadığı kontrol edilir. Eğer sadece 1 eleman varsa onun linki -1 olarak ayarlanır.

Bunu Yapan Kod Parçası:

```
XOR SI,SI  
MOV AX, WORD PTR INDEX  
CMP AX, 0002H  
JE AFIRST  
AFIRST: MOV WORD PTR LINK[SI], -1
```

Eğer dizide 1’den fazla eleman varsa dizideki en küçük eleman bulunur ve indexi BX register’ında saklanır. Eğer yeni girilen elemandan daha küçük elemanlar varsa onlar arasından en büyük olanı bulunur ve indexi DI registerında saklanır.

Eğer yeni girilen eleman, dizideki en küçük elemandan daha küçükse, yeni girilen elemanın gösterdiği link, bx registerının içindeki değerdir. Bu yüzden link listesinin, yeni elemanın eklendiği indexli bölgesine bx ‘teki değer atanır.

Eğer yeni girilen eleman dizideki en küçük eleman değilse, ondan bir küçük elemanın indexi DI registerında saklanıyor

demektir. Bu listenin DI indexli elemanın linki, yeni eklenen elemanın indexi olur. Yeni eklenen elemanın gösterdiği link ise listenin DI indexli elemanın gösterdiği link değeri olur. Böylece listenin arasına veya sonuna eleman eklenmiş olur.

Bunu Yapan Kod Parçası:

```
XOR AX,AX          ;AX -> COMPARED VALUE
XOR BX,BX          ;BX -> LOWEST NUMBER'S INDEX;
XOR DI,DI          ;DI -> GREATEST NUMBER BEFORE THE NEW NUMBER'S INDEX
XOR CX,CX          ;CX -> USED FOR FOR LOOP -> WE WILL LOOP N-1 TIME

MOV DI, WORD PTR LISTSIZE    ;ASSIGN DI TO THE LAST PLACE IN LIST
SUB DI,2
MOV WORD PTR LIST[DI],8000H   ;ASSIGN TO THE SMALLEST SIGNED NUMBER

;DETERMINE LOOP TIME = N-1 TIME
MOV CX, WORD PTR INDEX
SHR CX,1                ;DIVIDE BY TWO BECAUSE THE LIST IS WORD TYPED
DEC CX                  ;MAKE N, N-1

;TRY TO FIND THE GREATEST VALUE BEFORE THE CHOICE

AFIND:
    MOV AX, WORD PTR LIST[SI]    ;GET THE FIRST ELEMENT OF THE LIST TO AX
    CMP AX, WORD PTR CHOICE      ;COMPARE IT WITH THE NEW GIVEN NUMBER(CHOICE)
    JGE ASMALLER                 ;IF THE ELEMENT FROM LIST IS BIGGER DO NOTHING

    CMP AX, WORD PTR LIST[DI]    ;FIND OUT WHICH IS CLOSER TO THE GIVEN NUMBER
    JLE ASMALLER                 ;IF AX GREATER THAN LIST[DI], CHANGE DI
    MOV DI, SI                   ;DI NOW STORES THE INDEX OF AX

ASMALLER:
    CMP AX, WORD PTR LIST[BX]    ;COMPARE CURRENT LIST ELEMENT WITH THE
SMALLEST THOUGHT ONE
    JGE ABIGGER                  ;IF AX IS SMALLER CHANGE BX TO AX'S INDEX
    MOV BX, SI

ABIGGER:
    ADD SI,2                     ;SET SI TO THE NEXT ELEMENT (WORD TYPE)
    LOOP AFIND                   ;THEN LOOP

    MOV AX, WORD PTR LIST[BX]    ;GET THE SMALLEST VALUE OF THE LIST TO AX
REGISTER
    CMP AX, WORD PTR CHOICE      ;COMPARE NEW ELEMENT WITH THE SMALLEST
ELEMENT
    JGE ASMALLEST                ;IF THE NEW ELEMENT IS SMALLER, THEN ONLY THE LINK OF
;THE NEW ELEMENT WILL BE CHANGED
```

;ELSE THERE ARE 2 LINKS TO CHANGE:

```
MOV AX, WORD PTR LINK[DI]    ;GET THE LINK OF THE MAX VALUE BEFORE THE NEW  
ADDED ELEMENT TO AX
```

```
MOV SI, WORD PTR INDEX      ;SI NOW STORES THE INDEX OF THE NEW ADDED  
                             ;ELEMENT
```

```
SUB SI,2
```

```
MOV WORD PTR LINK[DI], SI    ;CHANGE THE LINK OF THE MAX VALUE BEFORE THE NEW  
                             ;ADDED ELEMENT TO THE NEW ADDED ONE'S INDEX
```

```
MOV WORD PTR LINK[SI], AX    ;CHANGE THE NEW ADDED ONE'S LINK TO THE ONE'S  
                             ;BEFORE THIS ONE
```

```
JMP AEXIT                    ;EXIT
```

ASMALLEST:

```
;CHANGE LINK OF THE NEW ADDED VALUE
```

```
MOV SI, WORD PTR INDEX      ;GET THE INDEX OF THE NEW ADDED VALUE  
SUB SI,2
```

```
MOV WORD PTR LINK[SI], BX    ;CHANGE THE NEW ADDED ONE'S LINK TO THE PREVIOUS  
                             ;SMALLEST ONE
```

```
JMP AEXIT
```

SUBMENUTWO:

Alt Menüler arasında en karmaşığı bu olduğundan burada anlatılacaktır.

İlk başta listedeki en küçük değerin indexini bx registerına koyar. Her elemanı sırayla gezip karşılaştırıyoruz. Bu işlem için loop kullanıyoruz ve n-1 defa dönüyoruz.

Bunu Yapan Kod Parçası:

```
MOV CX, WORD PTR INDEX  
SHR CX,1                    ;LIST IS WORD TYPE, SO WE NEED TO DIVIDE CX BY 2  
DEC CX                      ;WE NEED TO LOOP N-1 TIMES.  
MOV AX, WORD PTR LIST[BX]   ;MOVE THE FIRST ELEMENT OF THE LIST TO AX.
```

FINDSMALLEST:

```
ADD DI,2                    ;WE NEED TO INCREASE DI BY 2 (WORD defined)  
CMP AX, WORD PTR LIST[DI]   ;COMPARE MIN ELEMENT WITH CURRENT ELEMENT  
JLE SMALLER                 ;IF CURRENT ELEMENT IS SMALLER CHANGE AX AND BX
```

```
MOV AX, WORD PTR LIST[DI]    ;CHANGE AX TO THE MIN ELEMENT'S VALUE
MOV BX, DI                    ;CHANGE BX TO THE MIN ELEMENT'S INDEX
```

SMALLER:
LOOP FINDSMALLEST

En küçük elemanı bulduktan sonra n (eleman sayısı) kadar döner ve linklerdeki değeri takip ederek sıralı bir şekilde yazdırır.

Bunu Yapan Kod Parçası:

```
MOV CX,WORD PTR INDEX
SHR CX,1                ;LIST IS WORD TYPE, SO WE NEED TO DIVIDE CX BY 2
```

PRINTORDERED:

```
MOV AX, WORD PTR LIST[BX]    ;MOVE THE ELEMENT TO AX TO PRINT
CALL PUTN                    ;PRINT THE WANTED ELEMENT
```

```
MOV BX, WORD PTR LINK[BX]    ;MOVE TO THE NEXT ELEMENT FROM LINK
LOOP PRINTORDERED
```

Sıralı bir şekilde yazdırdıktan sonra düzensiz bir şekilde elemanları girildiği sırada yazdırır. n defa döner ve listedeki elemanları 0 indexinden başlayarak yazdırır.

Bunu Yapan Kod Parçası:

```
MOV CX, WORD PTR INDEX
SHR CX,1                ;LIST IS WORD TYPE, SO WE NEED TO DIVIDE CX BY 2
XOR BX,BX                ;BX -> CURRENT ELEMENT OF THE LIST. STARTS AT 0
```

PRINTUNORDERED:

```
MOV AX, WORD PTR LIST[BX]    ;MOV THE CURRENT ELEMENT TO AX TO PRINT
CALL PUTN
```

```
ADD BX, 2                ;WE NEED TO INCREASE BX BY 2 (word defined)
LOOP PRINTUNORDERED
```

Bundan sonra da linkleri yazdırır. n defa döner ve linkleri 0 indexinden başlayarak yazdırır. Liste word tipinde olduğundan linkler de 2'li 2'li ilerlemektedir. Bu yüzden

ekrana yazdırılmadan önce 2'ye bölünürler. Bir linkin değeri -1 olacağından, bir kontrol işlemi gerçekleştirilir ve negatif değerler 2'ye bölünmez.

Bunu Yapan Kod Parçası:

```
MOV CX, WORD PTR INDEX
SHR CX,1           ;LIST IS WORD TYPE, SO WE NEED TO DIVIDE CX BY 2
XOR BX,BX          ;BX -> CURRENT ELEMENT OF THE LINKS. STARTS AT 0

PRINTLINKS:

MOV AX, WORD PTR LINK[BX] ;MOV THE CURRENT ELEMENT TO AX TO PRINT
CMP AX,0H
JLE TNEGATIVE
SHR AX,1           ;DIVIDE AX BY 2, BECAUSE THE LINKS ARE IN WORDS
TNEGATIVE:
    CALL PUTN
```

Kaynakça:

https://tr.wikibooks.org/wiki/X86_Assembly/%C3%96nemli_int_servisleri

<https://github.com/80x86adtcebi/bookexamples>