YILDIZ TEKNİK ÜNİVERSİTESİ ELEKTRİK-ELEKTRONİK FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



BLM2021 ALT SEVİYE PROGRAMLAMA 1.ÖDEV RAPORU

AHMET SAİD SAĞLAM 17011501

SORU 2:

n elemanlı bir dizide değerleri 0 ila 1000 arasında değişen üçgen kenarı olmaya aday uzunlukları kullanıcıdan komut satırı aracılığıyla alan, bu dizideki kenar uzunluklarından, en az uzunluktaki çevreye sahip olacak üçgeni oluşturan kenarları ekrana yazdıran EXE tipinde assembly programını yazınız.

ÇÖZÜM:

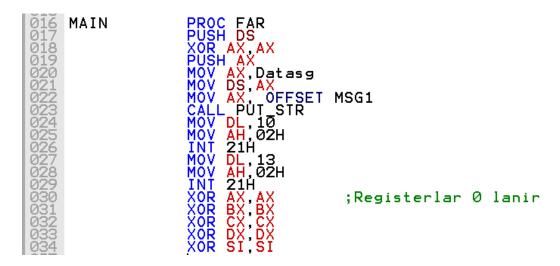
```
Stacksg SEGMENT PARA STACK 'STACK'
DW 32 DUP(?)

003 Stacksg ENDS
004 Datasg SEGMENT PARA 'DATA'
005 Ucgendizi DW 100 DUP(0)
006 ELEMAN DW ?

007 MSG1 DB 'ELEMAN SAYISINI VERINIZ:',0
008 MSG2 DB 'ELEMANLARI GIRINIZ:',0
009 MSG3 DB 'UYGUN KENAR YOK',0
010 MSG4 DB 'UYGUN KENARLAR:',0
011 MSG5 DB 'ONEMLI NOT:LUTFEN 3 BASAMAKLI SAYI GIRDIGINIZDE ENTER TUSUNA BASMAYINIZ!',0
012 Datasg ENDS
013 Codesg SEGMENT PARA 'CODE'
ASSUME CS:Codesg,DS:Datasg,SS:Stacksg
```

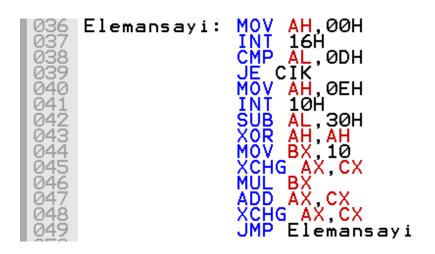
Kodun ilk kısmında öncelikle EXE tipi program yazmak için gerekli STACK SEGMENT tanımı yapılmaktadır. Daha sonra DATA SEGMENT kısmında ilk olarak programda bize gerekecek olan 100 elemanlı Ucgendizi isminde WORD tipinde tüm elemanları 0 olan bir dizi tanımlanmıştır. Bu diziye kullanıcıdan alınan üçgen kenarı olmaya aday elemanlar atılacaktır. Dizinin WORD tipinde tanımlanmasının nedeni kullanıcının girebileceği değerlerin 0 ile 1000 arasında olmasından kaynaklıdır. Dizinin tüm elemanlarının 0 tanımlanmasının sebebi ise daha sonradan girilen değerlerin 100 elemanın hepsini doldurmaması durumunda sıralama işleminde bozulmaya yol açmamasıdır.

Sonraki adımda ELEMAN isimli, Ucgendizi'nin eleman sayısını tutacak olan WORD tipinde bir değişken tanımıdır. WORD tipinde olmasının nedeni CX yazmacıyla yapacağımız işlemler sırasında tip uyumsuzluğu vermemesi amacıyladır. 7. satırdan 11. satıra kadar olan kısımda ekrana yazdırılacak olan mesajlar tanımlanmıştır. Daha sonra ise CODE SEGMENT kısmındaki gerekli tanımlama ve işlemlerden sonra MAIN etiketiyle ana program yazılmaya baslanmıştır.



22. satırda OFFSET sözde komutu yardımıyla MSG1 isimli değişkenin adresi AX yazmacına aktarılmıştır ve hemen sonraki satırdaki CALL komutu aracılığıyla PUT_STR yordamı çağırılarak AX yazmacındaki adresten itibaren bellekteki değişkenler NULL ASCII ile karşılaşana dek ekrana yazdırılmıştır. Bu yazdırma işlemi PUT_STR yordamıyla alakalıdır ve raporun ilerleyen kısımlarında nasıl çalıştığı açıklanmıştır.

24. ve 29. satırlar arasında DL yazmacındaki ASCII değeri aracılığıyla ve AH yazmacına 02H değeri atanarak INT 21H komutuyla kesme yapılarak ekranda düzenli bir şekilde yeni satıra geçilmesi sağlanmıştır. Daha sonraki kısımda ise kullanılacak yazmaçlar sıfırlanmıştır.



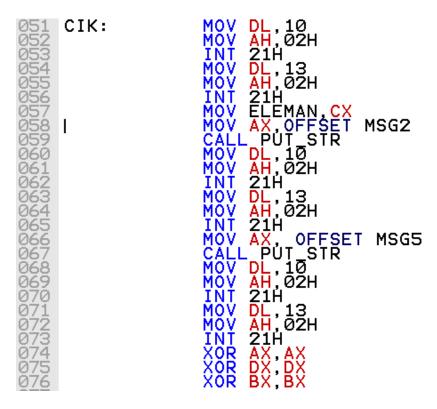
36. satırda dizinin eleman sayısını almak amacıyla yani kullanıcının kaç adet kenar değeri gireceğini öğrenmek için AH yazmacına 00H değeri atanarak ve INT 16H kesmesiyle klavyeden alınan ASCII değeri AL yazmacına atılmış ve hemen sonraki adımda CMP komutu vesilesiyle bu değerin Enter tuşunun ASCII değeri olup olmadığı karşılaştırılmıştır çünkü eğer kullanıcı Enter tuşuna

bastıysa kullanıcıdan eleman alma işlemi sonlanmak isteniyor demektir. Bu durumda CIK etiketine dallanma gerçekleşir. Aksi durumda AL yazmacında bulunan kullanıcıdan alınan değerden SUB komutu yardımıyla 30H çıkarılır ve yine AL yazmacında tutulur. Bu işlemin sebebi alınan ASCII karakterin unsigned sayı değerine çevrilmesi gerekliliğindendir. Bu işlem kodun ilerleyen kısımlarında ihtiyaç oluşmasıyla birlikte tekrarlanacaktır. Daha sonra AH yazmacı, AL yazmacındaki değer ile AX yazmacı vasıtasıyla işlem yapılacağından sıfırlanmıştır. Hemen sonraki satırda BX yazmacına bir sonraki işlemde kullanılmak üzere 10 değeri atanmıştır.

45-48 satırları arasındaki işlemler kodun daha sonraki kısımlarında da kullanılmıştır. Öncelikle AX ve CX yazmaçlarının değerleri XCHG komutu yardımıyla yer değiştirilmiştir. Sonrasında MUL komutu yardımıyla AX yazmacında bulunan ama esasında CX yazmacına ait olan değer, BX yazmacıyla yani 10 ile çarpılmıştır. Hemen sonraki satırda da bu çarpım CX yazmacında bulunan ama esasında AL yazmacına ait olan ve kullanıcıdan alınan değer ile toplanmıştır. En son adımda ise AX ve CX yazmaçlarındaki değerler yeniden yer değiştirilerek CX yazmacında olması gereken değer yeniden atanmıştır. Tüm bu işlem bloğunun amacı kullanıcının değerleri giriş sırasına ve 10'luk sayı sistemine göre basamakları belirlemek ve nihai eleman sayısına ulaşmaktır.

49. satırda JMP komutu ile Elemansayi etiketine dönülmüştür ve bu komut sayesinde kullanıcı Enter tuşuna basana dek bu işlem bloğu kendini tekrar edecektir.

Bu kısımda ödev dokümanında belirtildiği üzere kullanıcının dizinin boyutunu maksimum 100 vereceği kabul edilerek tam sayı, karakter veya girilen sayının büyüklüğü kontrol edilmemiştir. Programın kullanımı esnasında kullanıcının bu hususa dikkat etmesi gerekmektedir.



CIK etiketinde 51-56, 60-65 ve 68-73 satırları arasındaki işlemler daha önce de belirtildiği üzere ekranda yeni satıra geçme ile alakalı düzenleme komutlarıdır. 57. satırda CX yazmacındaki eleman sayısı ELEMAN isimli değişkene atılmıştır. 58-59 ve 66-67 satırlarında ekrana gerekli mesajlar yazdırılmıştır. 74-76 satırları arasında da daha sonra kullanılacak yazmaçlar sıfırlanmıştır.

```
Basadon: MOV AH, 00H INT 16H CMP AL, 0DH JE Enter JMP Karakter

Karakter: CMP AL, 58 JB Kucuk JMP Basadon

Buyuk: CMP AL, 58 JB Kucuk JMP Basadon

Kucuk: MOV AH, 0EH INT 10H SUB AL, 30H XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH WOV CX, 10 XOR AH, AH W
```

Basadon etiketi ile başlayan satırda kullanıcıdan kesme yardımıyla bir değer alınır ve bu değerin CMP komutu yardımıyla Enter tuşu olup olmadığı kontrol edilir. Eğer ki Enter ise Enter isimli etikete dallanma gerçekleşir. Aksi durumda Karakter isimli etikete dallanma olur. Karakter ve Buyuk isimli etiketlerde kullanıcının klavyeden değer girerken rakamlar hariç herhangi bir tuşa basması durumunda Basadon etiketine dallanma yapılarak sanki o tuşa hiç basılmamışçasına bir etki oluşturulur. Böylece kullanıcı pozitif tam sayılar hariç hiçbir değeri giremez. Bu kontrol basılan tuşun ASCII değerleri karşılaştırılarak yapılır. Eğer tüm şartlar sağlanıyorsa Kucuk isimli etikete geçilir.

93-94 satırlarında kullanıcının girdiği değer ekrana yazdırılır. 95. satırdan itibaren daha önce AX, BX ve CX yazmaçlarıyla yapılan ASCII değeri unsigned değere çevirme ve basamak kontrolü işlemleri aynı şekilde yapılır ancak burada bazı farklar bulunmaktadır. İlk olarak burada CX ve BX yazmaçlarındaki değere ihtiyaç duyduğumuz için önce o değerler yığına atılır. CX yazmacına 10 değeri atılarak çarpma işleminde kullanılır. BX yazmacının kullanılma nedeni ise MUL işleminde AX yazmacıyla beraber DX yazmacının da değeri değiştiğinden kısa süreliğine DX yazmacının yerini alması gerekliliğindendir.

CX yazmacındaki orijinal değere döngü sayısını belirlemek adına ihtiyaç duyulurken BX yazmacındaki değere ise ödev dokümanında belirtildiği üzere kullanıcının 1000'e kadar sayı girebilme izninin kontrolü sırasında ihtiyaç duyulmaktadır. Bu kontrol şu şekilde işlemektedir : kullanıcı her tuşa bastığında BX yazmacının değeri 1 arttırılmıştır. Eğer yazmacın değeri 3 olursa Enter isimli etikete dallanma olur ve kullanıcıdan başka bir değer için tuşa basılması istenir. Böylece kullanıcı 3 basamaklıdan daha fazla basamağa sahip hiçbir sayı değerini giremez. Burada dikkat edilmesi gereken bir husus vardır. Eğer kullanıcı 3 basamaklı bir değer girerse son hamlede Enter tuşuna basmamalıdır çünkü program otomatik olarak 3. basamaktan sonra bir sonraki değeri almaya hazır konuma gelecektir.

```
113 Enter: MOV Ucgendizi[SI],DX
MOV DL,10
MOV AH,02H
INT 21H
MOV DL,13
MOV AH,02H
INT 21H
INT 21H
XOR DX,DX
XOR BX,BX
ADD SI,2
LOOP Basadon
```

Enter isimli etikette daha önceki işlemlerde elde ettiğimiz ve DX yazmacında bulunan, kullanıcının girdiği üçgen kenarı olmaya aday uzunluk değeri Ucgendizi isimli dizinin ilk kontrolde, SI yazmacı yardımıyla ilk elemanına atılır ve sonrasında dizi WORD tipinde olduğu için SI 2 arttırılır. Böylece sonraki değer dizinin bir sonraki gözüne yerleştirilebilir hale gelir. 114-119 satırları ekranda yeni satıra geçmek amacıyla yazılmıştır. 120. ve 121. satırlarda ise sonraki adımda kullanılacak yazmaçlar sıfırlanır ve nihayetinde LOOP komutu ile Basadon etiketine dizinin eleman sayısı kadar dönülür. Böylece eleman sayısı kadar değer kullanıcıdan alınmış olunur.

Bu blokta dizinin elemanları, SI yazmacı yardımıyla dizinin elemanlarına erişerek, CX yazmacı yardımıyla eleman sayısı kadar çift döngünün içinde ve AX yazmacı sayesinde Bubble Sort algoritmasına göre küçükten büyüğe sıralanmıştır. Bu sıralamanın nedeni en küçük çevreye sahip üçgeni yakalarken kullanılan algoritmada gerekli olmasındandır. Burada bir detay bulunmaktadır. Dizinin elemanları 0 lar dahil olmak üzere 100 tanedir ama sıralamada sadece

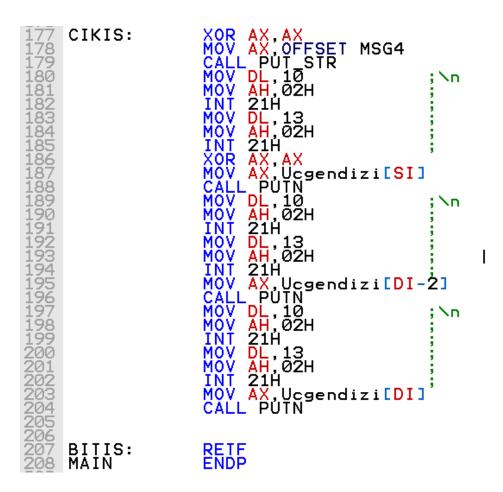
eleman sayısı kadar döngüye girilmiştir. Bu sayede Ucgendizi dizisinin içinde sadece kullanıcı tarafından verilen elemanlar kendi arasında sıralanmıştır. Üçgeni bulan algoritmada da bu husus göz önünde bulundurulmuştur.

```
142
143
1443
1445
1446
1447
1448
1499
1550
1491
1551
1552
149
1556
1557
1558
1661
1667
1668
1667
1668
1670
1771
1712
173
```

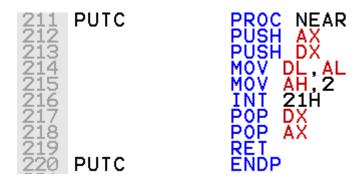
142-146 satırlarında üçgeni bulmakta kullanılacak olan yazmaçlar sıfırlanır ve hemen akabinde DI yazmacına 2 eklenir. Bunun nedeni DI yazmacının SI yazmacının gösterdiği dizi elemanından bir sonrakini göstermesi isteğidir. Sonrasında DL yazmacına bölmede kullanmak amacıyla 2 değeri atanmıştır. Döngüde kullanılacak olan CX yazmacına eleman sayısının 2 eksiği bir değer atanır. Bunun sebebi ilk iki elemanı zaten en başta elimizde bulundurmamız ve kontrole hiç sokmamızdır. 152. Satırdan itibaren dış döngünün içinde iç döngüdeki çevrim sayısını bulmak amacıyla yazmaçlar kullanılarak işlemler yapılmıştır. Böylece her kontrolden sonra bir eleman ilerleyen dizide bir sonraki kontrolü etkileyecek bir taşma oluşmaz. Bu kontroller şu şekilde işlemektedir:

Dizinin ilk elemanından başlamak üzere bir eleman BX yazmacına atılır ve kendisinden bir sonraki elemandan başlayarak son eleman hariç diğer tüm elemanlarla küçüklük-büyüklük sırasına göre toplanır ve toplanan elemandan bir sonraki eleman ile karşılaştırılır. Dizinin elemanlarının küçükten büyüğe sıralanması burada faydalı olmuştur. Bu karşılaştırmadan sonra eğer toplam 3. elemandan büyükse CIKIS etiketine dallanma olur ve üçgen oluşmuş demektir.

Aksi halde döngü kendini tekrar eder. İç döngünün tamamlanmasının ardından dış döngü sayesinde aynı işlem 2. eleman ile ondan sonrakilerin toplamı ve karşılaştırılması şeklinde, uygun üçgenin bulunamaması durumunda sondan üçüncü elemana kadar devam eder. Eğer CIKIS etiketine dallanma gerçekleşmez ise uygun kenar değerleri girilmemiş demektir. Bu durumda AX yazmacı ve PUT_STR yordamı sayesinde ekrana uygun mesaj yazdırılır ve BITIS etiketine dallanma gerçekleşir ve de program sonlandırılır.

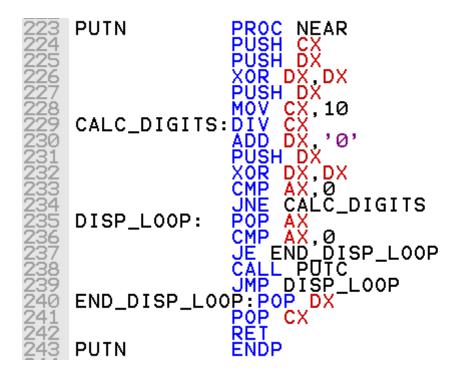


Uygun kenarların bulunması durumunda ise kenarlar PUTN yordamı yardımı yardımıyla ekrana yazdırılır ve program biter.



PUTC YORDAMI

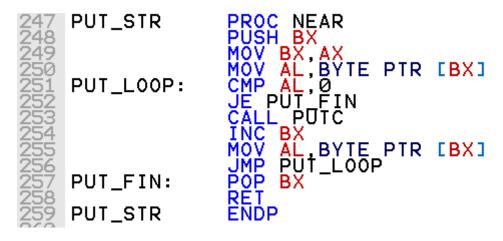
Ekrana kesme kullanarak karakter yazdırmaya yarar. INT 21H kesmesi sayesinde AH yazmacında 2 değeri olduğu sürece, DL yazmacındaki değer ekrana yazdırılır. PUTC yordamı içinde AL yazmacındaki değer DL yazmacına aktarılır ardından da kesme kullanılarak ekrana karakter yazdırılır. Ek olarak AX ve DX yazmaçlarındaki değer bozulmaması amacıyla yığına atılmıştır. Bu işlem diğer kullanılan iki yordamda da , yordamın içinde kullanılan yazmaçlara bağlı olarak tekrarlanır.



PUTN YORDAMI

AX yazmacında bulunan sayıyı PUTC yordamını da kullanarak ekrana onluk tabanda hane hane yazdırmaya yarar. İlk olarak CALC_DIGITS etiketi içinde AX yazmacında bulunan sayıyı CX yazmacı sayesinde 10'a böler ve kalan değerinin (DX yazmacında saklanır) ASCII değerini bularak onu yığına atar

(230-231.satırlar). Sonrasında bölüm değerini 0 ile kıyaslar. Eğer 0 ise bölme işlemi tamamlanmış demektir ve rakamları karakter karakter yazdırmak üzere DISP_LOOP etiketine geçer. Değil ise bölüm 0 olana kadar işlem tekrarlanır. DISP_LOOP etiketinde yığındaki ASCII değerleri AX yazmacına alınır ve PUTC yordamı yardımıyla sonda 0 ASCII değeri görene kadar ekrana sıra ile yazdırılır ve yordam bu şekilde tamamlanır.



PUT_STR YORDAMI

AX yazmacında adresi verilen ve sonunda 0 olan karakter dizisini ekrana karakter karakter yazdırır. Sondaki sıfır değeri karakter dizisinin sonuna gelindiğini anlamaya yarar ve yordamı sonlandırmak üzere dallanma yapılır.



CODE segmentle birlikte kod sona erer.

ÖRNEK ÇIKTILAR

Üçgen Oluşturan Kenar Değerleri Girilmesi ve Program Çıktısı

```
DOSBOX 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG — X

Run File [17011501.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:

C:\>
C:\>debug 17011501.exe

-g
ELEMAN SAYISINI VERINIZ:
9
ELEMANLARI GIRINIZ:
ONEMLI NOT:LUTFEN 3 BASAMAKLI SAYI GIRDIGINIZDE ENTER TUSUNA BASMAYINIZ!
45
17
36
2
5
993
51
17
UYGUN KENARLAR:
2
4
5
```

Üçgen Oluşturmayan Kenar Değerleri Girilmesi ve Program Çıktısı

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
                                                                                   X
                            1,354 27-11-2019 12:48
HWQSC
HWQSC
                              965 27-11-2019 12:47
         OBJ
                          64,982 31-07-1987 1:00
LINK
         EXE
                               57 05-02-2014 0:16
MAKE
         BAT
                          17,024 16-10-1985 23:00
MAKE
         EXE
                          111,611 29-03-1995 11:52
MASM
         EXE
                         52 14-02-2012 12:09
260,852 Bytes.
RUN
         BAT
   18 File(s)
                     262,111,744 Bytes free.
   2 Dir(s)
C:\>debug 17011501.exe
-g
ELEMAN SAYISINI VERINIZ:
ELEMANLARI GIRINIZ:
ONEMLI NOT:LUTFEN 3 BASAMAKLI SAYI GIRDIGINIZDE ENTER TUSUNA BASMAYINIZ!
10
500
300
958
40
67
uygun kenar yok
Program terminated normally
```