

YILDIZ TEKNİK
ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ
BLM2021
ALT SEVİYE PROGRAMLAMA
1. ÖDEV RAPORU

İsim: Orçun
Soyisim: Çelik
Numara: 15011053

1. Soru

```

1 void SteganografiBul(int n, int resim_org, int resim_ste, int steganografi_adres) {
2     __asm {
3         MOV ESI, resim_org           ;Orijinal resim dizisinin adresini ESI tutar.
4         MOV EDI, resim_ste           ;Steganografik resim dizisinin adresini EDI
           tutar.
5         MOV EBX, steganografi_adres ;Sifreyi yazacagimiz dizinin adresini EBX
           tutar.
6         MOV ECX, n                   ;32 bitlik count registeri ECX matrisin a*a = n
           boyut tutar.
7     L1: MOV AX, [EDI]                ;Steganografik resim dizini degerini AX e atariz.
8         CMP AX, [ESI]                ;Steganografik resim dizisi ile orijinal resim
           dizisin karsilastirilmesi.
9         JA toplam                    ;Ste dizisi eger Org dizisinden buyuk ise mod alma
           durumu olmadan ekleme sozkonusu, toplam labelina gidilir.
10        JE ortak                     ;Esit oldugunda ortak labelindan loopa devam eder.
11        ADD AX, 256                   ;Org dizisi eger Ste dizisinden buyukse 256 ya
           gore mod alma islemi.
12    toplam: SUB AX, WORD PTR[ESI]     ;Sifreyi bulmak icin Ste dizisinden Org dizisi
           cikarilir.
13        MOV BYTE PTR[EBX], AL        ;AL registerindeki sifre steganografi_adres e
           atilir.
14        INC EBX                      ;steganografi_adres in indisi 1 arttirilir.(1byte
           oldugundan)
15        JMP ortak                    ;ortak arttirilacak indislere dallanma yapilir.
16    ortak: ADD EDI, 2                 ;Steganografik resim dizisinin indisi 2
           arttirilir.(2byte oldugundan)
17        ADD ESI, 2                   ;Steganografik resim dizisinin indisi 2
           arttirilir.(2byte oldugundan)
18        LOOP L1
19
20        ;Numara yazma islemi
21        MOV BYTE PTR[EBX], 45; Tire karakteri ASCII karşılığı
22        INC EBX
23        MOV BYTE PTR[EBX], 49; 1 ASCII karşılığı
24        INC EBX
25        MOV BYTE PTR[EBX], 53; 5 ASCII karşılığı
26        INC EBX
27        MOV BYTE PTR[EBX], 48; 0 ASCII karşılığı
28        INC EBX
29        MOV BYTE PTR[EBX], 49; 1 ASCII karşılığı
30        INC EBX
31        MOV BYTE PTR[EBX], 49; 1 ASCII karşılığı
32        INC EBX
33        MOV BYTE PTR[EBX], 48; 0 ASCII karşılığı
34        INC EBX
35        MOV BYTE PTR[EBX], 53; 5 ASCII karşılığı
36        INC EBX
37        MOV BYTE PTR[EBX], 51; 3 ASCII karşılığı
38        INC EBX
39        MOV CX, 20
40    L0: MOV BYTE PTR[EBX], 0; Gereksiz bolgeleri temizlemek icin.
41        INC EBX
42        LOOP L0
43    }
44 }
45

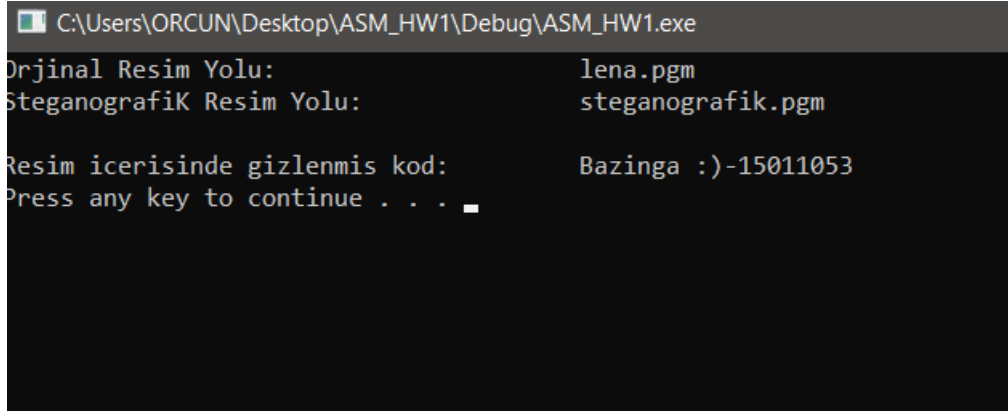
```

1. Soru

Şifreyi çözmek için orijinal resim ve steganografi dizilerini karşılaştırdım. Eğer steganografi dizisi büyükse normal olarak

eklenme durumu vardır. Orijinal resim dizisi büyük ise mod 256 işlemini yaparak, farklı olan piksellerin değerlerini steganografi adres dizine göndererek sonuca ulaştım

EKRAN ÇIKTISI



```
C:\Users\ORCUN\Desktop\ASM_HW1\Debug\ASM_HW1.exe
Orjinal Resim Yolu:          lena.pgm
Steganografik Resim Yolu:    steganografik.pgm

Resim icerisinde gizlenmis kod:      Bazinga :) -15011053
Press any key to continue . . .
```

2. Soru

Bu soru için iki farklı yaklaşım düşündüm.

İlk olarak iç içe 3 for döngüsüyle çözmek lakin karmaşıklık $O(n^3)$ olduğundan dolayı ikinci seçenek olarak bubble sort algoritması yardımıyla diziyi sıraladım $O(n^2)$ ardından ise while döngüsü içerisinde if bloklarıyla çözüme ulaştım.

EKRAN ÇIKTILARI

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
C:\ASM>15011053s2
Boyut girin.
->9

Sayilari girin.
->12

->5

->9

->51

->7

->8

->18

->1

->14

Sonuc: 5 7 8
C:\ASM>
```

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Drive C is mounted as local directory c:\
Z:\>c:\
C:\>cd asm
C:\ASM>15011053s2
Boyut girin.
->6

Sayilari girin.
->500

->100

->50

->10

->250

->1

Ucgen sarti saglanmiyor!!!
C:\ASM>_
```

```

1  myds SEGMENT PARA 'data'
2  CR EQU 13
3  LF EQU 10
4  n DW ? ;Dizi
   boyutu
5  dizi dw ? ;Dizi
6  MSG1 DB 'Boyut girin.',0
7  MSG2 DB CR, LF, 'Sayilari girin.',0
8  LINE DB CR, LF, '->',0 ;Next
   line ve tasarim acisindan optimizasyon '->'
9  SPACE DB ' ',0
   ;Output verirken tasarim acisindan optimizasyon
10 HATA0 DB CR, LF, 'Hata!!! 0-1000 arasinda bir sayi girin.',0
11 HATA1 DB CR, LF, 'Negatif sayi giremezsiniz!!! Lutfen tekrar girin.',0
12 HATA2 DB CR, LF, 'Hata!!! Lutfen rakam girin.',0
13 HATA3 DB CR, LF, 'Ucgen sarti saglanmiyor!!!',0
14 SONUC DB CR, LF, 'Sonuc: ',0
15
16 myds ENDS
17 myss SEGMENT PARA STACK 'stack'
18 DW 12 DUP(?)
19 myss ENDS
20 mycs SEGMENT PARA 'code'
21 ASSUME CS:mycs, DS:myds, SS:myss
22 ANA PROC FAR
23 PUSH DS
24 XOR AX,AX
25 PUSH AX
26 MOV AX,myds
27 MOV DS,AX
28 ;EXE tipi programlar icin on ayar
29
30 MOV AX, OFFSET MSG1 ;Boyut girin mesajı
31 CALL PUT_STR
32 MOV AX, OFFSET LINE
33 CALL PUT_STR ;
34 CALL GETN
35 MOV n,AX ;Boyut 'n' de saklanır
36 CMP AX,2 ;3 elemandan az girilirse ucgen sarti saglanmadigindan
   program hata mesajı verir.
37 JA devam ;Eger boyut 2'den buyukse program devam eder.
38 MOV AX, OFFSET HATA3 ;Ucgen sarti saglanmadigi hatasinin OFFSET'i AX
   registerina aktarilir.
39 CALL PUT_STR ;Ucgen sarti saglanmadigina dair hata ekrana basılır.
40 JMP Lexit ;Ucgen sarti saglanmadigi icin programdan cikis yapilir.
41
42 devam:
43 XOR SI,SI
44 MOV AX, OFFSET MSG2 ;Sayilari girin mesajı
45 CALL PUT_STR
46 ;Input alma islemi
47 MOV CX,n ;n dizi boyutu kadar donguye girer.
48 Lin: MOV AX, OFFSET LINE
49 CALL PUT_STR ;'->' Tasarım için.
50 CALL GETN ;Konsoldan girilen sayi AX'e aktarilir.
51 MOV dizi[SI],AX ;Dizinin elemanları AX'ten alinir.
52 ADD SI,2
53 LOOP Lin
54
55
56 ;___BUBBLE_SORT___

```

```

57 ;Bubble Sort kucukten buyuge siralama algoritması
58     MOV CX,n
59     XOR BX,BX                                ;i
60     DEC CX                                    ;n-1
61     XOR SI,SI
62
63 ;Bubble Sort kucukten buyuge siralama algoritması
64     LB1: PUSH CX                                ;n-1 stackte
65           MOV CX,n                            ;yeni cx
66           DEC CX
67           SUB CX, BX                        ;n-i-1
68           XOR SI,SI                        ;SI index 0
69           LB2: MOV AX, dizi[SI]                ;İlk eleman AX'te
70                   CMP AX, dizi[SI+2]          ;İkinci elemanla karşılaştırıyoruz
71                   JB cikis                    ;Jump below ise cik buyuk ise devam
72                   XCHG dizi[SI+2], AX        ;Dizinin 2. elemanla AX yer degistir.
73                   MOV dizi[SI], AX          ;Dizinin 1. elamana AX'i ata.
74                   cikis:
75                   ADD SI,2
76           LOOP LB2
77           INC BX                            ;i
78           POP CX
79     LOOP LB1
80 ;
81     XOR SI,SI    ;dizi
82     XOR BX,BX    ;i gorevi goruyor.
83     XOR DI,DI    ;sign
84     MOV CX,n     ;n
85
86     LWHILE: CMP BX,CX
87             JA exitwhile
88             CMP DI,0                                ;Eger en kucuk ucgen olusmadiysa 0 dir
89             JNE exitWHILE
90             MOV AX, dizi[SI]                        ;a
91             ADD AX,dizi[SI+4]                        ;a+c
92             CMP AX, dizi[SI+2]                      ;comparing a+c, b
93             JB exitIF
94             MOV AX, dizi[SI]                        ;a
95             ADD AX, dizi[SI+2]                      ;a+b
96             CMP AX, dizi[SI+4]                      ;comparing a+b, c
97             JB exitIF
98             MOV AX, dizi[SI+2] ;b
99             ADD AX, dizi[SI+4] ;b+c
100            CMP AX, dizi[SI] ;comparing b+c, a
101            JB exitIF
102            ;eger tum sartlari sagliyorsa
103            MOV BX,SI ;konum icin
104            PUSH BX ;en kucuk ucgenin ilk indsinin konumunu
105            stacke attim
106            MOV DI, 1 ;sign 1 oldu,en kucuk ucgen
107            bulundu
108            JMP exitIF
109
110            exitIF: ADD SI,2
111                    DEC CX ;While dongusunda oldugumuzdan count registerinin
112                    degerini azalttik.
113                    INC BX ;i degerini arttirdik.
114                    JMP LWHILE
115
116 exitWHILE:
117     CMP DI,0 ;ucgen sarti saglanmissa DI 1 dir.
118     JNE ucgen ;DI 0'a esit degilse ucgen sarti saglanmis demektir.
119     MOV AX, OFFSET HATA3 ;Ucgen sarti saglanamadigina dair hata mesajı
120     CALL PUT_STR

```



```

116         JMP Lexit                ;Ucgen sarti saglanmadigi icin programdan cikis
        yapilir.
117 ucgen:  ;Eger sonuc olustuysa ekrana yazdirma islemi.
118         MOV AX, OFFSET SONUC
119         CALL PUT_STR
120         POP BX
121         MOV CX,3
122         print: ;Sonucun ekranda yazdirilmasi
123             MOV AX,dizi[BX]
124             CALL PUTN
125             MOV AX, OFFSET SPACE
126             CALL PUT_STR
127             ADD BX,2
128         LOOP print
129 Lexit:   ;Programdan cikis labelimiz.
130         RETF
131 ANA     ENDP
132 ;
133 ;-----
134 ;Kullanicidan veri girisi alabilmemiz icin gerekli yordamlar
135
136 GETC PROC NEAR ;Klavyeden alinan karakteri alir ve AL yazmacina alip, ekranda
        gosterir
137         MOV AH, 1h
138         INT 21H
139         RET
140 GETC ENDP
141 ;-----
142 PUTC PROC NEAR ;AL yazmacindaki degeri ekranda gosterir
143         PUSH AX
144         PUSH DX
145         MOV DL, AL
146         MOV AH,2
147         INT 21H
148         POP DX
149         POP AX
150         RET
151 PUTC ENDP
152 ;-----
153 GETN PROC NEAR ;Klavyeden girilen degeri degeri AX registirina atar.
154         PUSH BX
155         PUSH CX
156         PUSH DX
157         PUSH DI                ;Basamak sayisini tutar.(3 haneyi gecmesin diye)
158 GETN_START:
159         ; MOV DX,1
160         XOR BX,BX
161         XOR CX,CX
162         XOR DI,DI
163 NEW:     CMP DI,3                ;Sayimiz 4 basamakli olamaz, bunun icin basamak kontrolü.
164         JA ERROR0                ;
165         CALL GETC
166         CMP AL,CR
167         JE FIN_READ                ;Enter tusuna basildiysa okuma biter
168         CMP AL, '-'                ;AL - mi geldi mi?
169         JNE CTRL_NUM                ;Gelen sayi 0-9 arasında mi
170 NEGATIVE:
171         JMP ERROR1                ;Negatif hata mesajı
172 CTRL_NUM:
173         CMP AL, '0'                ;Sayi 0 ile 9 arasında mi kontrolu
174         JB ERROR2
175         CMP AL, '9'
176         JA ERROR2

```

```

176     SUB AL,'0'
177     MOV BL,AL
178     MOV AX,10
179 ;     PUSH DX
180     MUL CX
181 ;     POP DX
182     MOV CX,AX
183     ADD CX,BX
184     INC DI; ;Basamak arttirdik
185     JMP NEW ;Klavyeden yeni basilan degeri al
186 ERROR0:
187     MOV AX, OFFSET HATA0
188     CALL PUT_STR ;Hata!!! 0-1000 arasinda bir sayi girin:'
189     MOV AX, OFFSET LINE
190     CALL PUT_STR
191     JMP GETN_START
192 ERROR1:
193     MOV AX, OFFSET HATA1
194     CALL PUT_STR ;Negatif sayi giremezsiniz!!!
195     MOV AX, OFFSET LINE
196     CALL PUT_STR
197     JMP GETN_START
198 ERROR2:
199     MOV AX, OFFSET HATA2
200     CALL PUT_STR ;Hata!!! Lutfen rakam girin
201     MOV AX, OFFSET LINE
202     CALL PUT_STR
203     JMP GETN_START
204
205
206 FIN_READ:
207     MOV AX, CX
208 FIN_GETN:
209     POP DI
210     POP DX
211     POP CX
212     POP DX
213     RET
214 GETN ENDP
215 ;-----
216 PUTN PROC NEAR ;AX de bulunan sayiyi onluk tabanda hane hane yazdırır
217     PUSH CX
218     PUSH DX
219     XOR DX,DX
220     PUSH DX
221     MOV CX,10
222     CMP AX,0
223     JGE CALC_DIGITS
224     NEG AX
225     PUSH AX
226     MOV AL, '-'
227     CALL PUTC
228     POP AX
229 CALC_DIGITS:
230     DIV CX
231     ADD DX, '0'
232     PUSH DX
233     XOR DX,DX
234     CMP AX,0
235     JNE CALC_DIGITS
236 DISP_LOOP:
237     POP AX

```

```

238     CMP AX,0
239     JE END_DISP_LOOP
240     CALL PUTC
241     JMP DISP_LOOP
242 END_DISP_LOOP:
243     POP DX
244     POP CX
245     RET
246 PUTN ENDP
247 ;-----
248 PUT_STR PROC NEAR;AX de adresi verilen sonunda 0 olan stringi karakter karakter
yazdırır. BX stringe indis olarak kullanılır
249     PUSH BX
250     MOV BX, AX
251     MOV AL, BYTE PTR[BX]
252 PUT_LOOP:
253     CMP AL,0
254     JE PUT_FIN
255     CALL PUTC
256     INC BX
257     MOV AL, BYTE PTR[BX]
258     JMP PUT_LOOP
259 PUT_FIN:
260     POP BX
261     RET
262 PUT_STR ENDP
263
264 mycs ENDS
265     END ANA
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289

```