

**TÜRKİYE CUMHURİYETİ**  
**YILDIZ TEKNİK ÜNİVERSİTESİ**  
**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**



**BLM2021 Alt Seviye Programlama Dersi**  
**2. Ödev Raporu**

**İsim Soyisim:** Berkay Gümüşay

**Numara:** 21011084

**Tel No:** 0 535 840 79 78

**E-Posta Adresi:** [berkaygumusay@std.yildiz.edu.tr](mailto:berkaygumusay@std.yildiz.edu.tr)

**Not :** **Erosion** ve **Dilation** 3 satır hariç tamamen aynı kodlara sahip olduğu için tek kod üzerinden anlatılmış, gereken yerlerde ayırım yapılmıştır.

## 1- Satır ve sütunları gezecek döngünün adım sayısını belirleme

```
142  __asm {
143      // resimde sütun / satir döngüsü
144      mov eax, 512
145      sub eax, filter_size
146      inc eax
147      mov ecx, eax
148      mov edx, eax
149      xor eax, eax
```

- a- Burada **mov eax, 512** komutu ile satır sayısı olan **512**, **eax** registerine aktarılır.
- b- **sub eax, filter\_size** ve **inc eax** komutları ile satırda ilerlenebilecek en son nokta tespit edilir.
- c- **mov ecx, eax** ile döngü oluşması için bu sayı **ecx** e aktarılır.
- d- **mov edx, eax** ile bu değer **edx** değişkeninde saklanır.
- e- **xor eax, eax** komutu ile daha sonra kullanılacak olan **eax** registeri sıfırlanır.

## 2- Resimin satırlarını gezecek olan döngü (İlk kısım)

```
151  resimSatirLoop :
152      push ecx
153      mov edi, tmp
154      mov esi, resim_org
155      add esi, eax
156      add edi, eax
157      mov ecx, edx
158      push edx
159      xor edx, edx
```

- a- **resimSatirLoop:** komutu ile resmin satırlarını gezecek olan döngü oluşturulur.
- b- **push ecx** ile önceki döngünün adım sayısı değişmemesi için **ecx** registeri **stack'e** atılır.
- c- **mov edi, tmp** komutu ile ek dizinin adresi **edi** registerine aktarılır.
- d- **mov esi, resim\_org** ile resim dizisinin adresi **esi** registerine aktarılır.
- e- **add esi, eax** ve **add edi, eax** komutları ile her döngünün başında **esi** ve **edi** sonraki satıra geçer (**Devamı raporun ilerisinde**).
- f- **mov ecx, edx** ile önceden **edx'e** atadığımız adım sayısını yeni döngü için de uyguluyoruz çünkü elimizdeki resim, kare bir resimdir.
- g- **push edx** ile daha sonra **dx** registerini kullanacağımız için **edx** registeri değişmesin diye **stack'e** atarız.
- h- **xor edx, edx** ile **edx** registeri sıfırlanır.

### 3- Resimin sütunlarını gezecek olan döngü (İlk kısım)

Eroton:

Dilation:

```
161      resimSutunLoop :
162          push ecx
163          mov ecx, filter_size
164          push eax
165          xor eax, eax
166          mov dx, 255
```

```
87      resimSutunLoop :
88          push ecx
89          mov ecx, filter_size
90          push eax
91          xor eax, eax
92          xor dx, dx
```

- a- **resimSutunLoop**: komutu ile resmin sütunlarını gezecek olan döngü oluşturulur.
- b- **push ecx** ile önceki döngünün adım sayısı değişmemesi için **ecx** registeri **stack'e** atılır.
- c- **mov ecx, filter\_size** ile sonraki döngüde adım sayısı filtre boyutu kadar olacağı için bu değer **ecx'e** atılır.
- d- **push eax** ile ileride **eax** üzerinde yapılacak değişikliklerden dolayı önceki değeri saklamak için **eax, stack'e** atılır.
- e- **xor eax, eax** ile **eax** registeri sıfırlanır.
- f- **mov dx, 255** ve **xor dx, dx** :

**f.1-** Erosionda en küçük arandığı için tüm değerlerden büyük olacak olan **255** değeri seçilmiştir.

**f.2-** Dilation fonksiyonunda ise bu değer **0** dir. Çünkü en büyük değer aranmaktadır.

### 4- Filtrenin satırlarını gezecek olan döngü (İlk kısım)

```
169      filtreSatirLoop:
170          mov ebx, esi
171          add ebx, eax
172          push ecx
173          mov ecx, filter_size
174          //-----
```

- a- **filtreSatirLoop**: komutu ile filtrenin satırlarını gezecek olan döngü oluşturulur.
- b- **mov ebx, esi** komutu **resim\_org** dizisindeki mevcut değer adresini(**filtrenin başlangıç noktası**) **ebx** registerine koyar.
- c- **add ebx,eax** komutu ile resim dizisini gezerken yapıldığı gibi **eax** registerine ileride eklenecek değerler ile **ebx** registeri 2. Döngü ve sonrasında aşağı satırlara iner.
- d- **push ecx** komutu ile döngünün adım sayısı **stackte** saklanır.
- e- **mov ecx, filter\_size** ile sütunları gezecek döngünü adım sayısı olan filtre boyutu **ecx** registerine aktarılır.

## 5- Filtrenin sütunlarını gezecek olan döngü (İlk kısım)

Erosion:

```
173 //-----
174 filtreSutunLoop :
175     cmp dx, WORD PTR[ebx]
176     jb  küçük
177     mov dx, WORD PTR[ebx]
178     küçük : add ebx, 2
179     loop filtreSutunLoop
180 //-----
```

Dilation:

```
100
101
102
103
104
105
    filtreSutunLoop :
        cmp dx, WORD PTR[ebx]
        jae büyük
        mov dx, WORD PTR[ebx]
        büyük : add ebx, 2
        loop filtreSutunLoop
```

- a- **filtreSutunLoop**: komutu ile filtrenin sütunlarını gezecek olan döngü oluşturulur.
- b- **cmp dx, WORD PTR[ebx]** komutu ile filtrede gezerken mevcut değer **Erosion için Min** değerini **Dilation için ise Max** Değerini tutan **dx** registeri ile karşılaştırılır.
- c- **jb küçük (Erosion) / jae büyük (Dilation)** komutları ile fonksiyona göre **büyüklik/küçüklük kontrolü** yapılır ve duruma göre **jump** işlemi gerçekleştirilir.
- d- **mov dx, WORD PTR[ebx]** komutu ile **Erosion için** eğer mevcut değer **dx** ten daha küçükse **dx** mevcut değer ile güncellenir, **Dilation için** ise eğer mevcut değer **dx'ten daha büyükse dx** mevcut değer ile güncellenir.
- e- **kucuk: add ebx, 2 / büyük: add ebx, 2** komutları ile **ebx** bir sonraki değere geçer.

## 6- Filtenin satırlarını gezecek olan döngü (Son Kısım)

```
181
182
183
    pop ecx
    add eax, 1024
    loop filtreSatirLoop
```

- a- **pop ecx** ile ecx stackten çekilerek döngünün devamı sağlanır.
- b- **add eax, 1024** ile öncesinde anlatıldığı gibi her adımda **ebx'in** bir satır aşağıya inmesi sağlanır.
- c- **loop filtreSatirLoop** ile döngü bitirilir.

## 7- Resimin sütunlarını gezecek olan döngü (Son Kısım)

```
184      mov WORD PTR[edi], dx
185      pop eax
186      pop ecx
187      add edi, 2
188      add esi, 2
189      loop resimSutunLoop
```

- a- **mov WORD PTR[edi], dx** ile ek dizinin mevcut noktasına filtreden elde ettiğimiz piksel yerleştirilir.
- b- **pop eax/pop ecx** ile stackteki **eax** ve **ecx** değerleri **stackten** çekilir.
- c- **add edi, 2/add esi, 2** ile **edi** ve **esi** registerleri bir sonraki değişkeni gösterecek şekilde ayarlanır.
- d- **loop resimSutunLoop** ile loop sonlandırılır.

## 8- Resimin satırlarını gezecek olan döngü (Son Kısım)

```
191      //-----
192      add eax, 1024
193      pop edx
194      pop ecx
195      loop resimSatirLoop
```

- a- **add eax, 1024** ile her adımda **esi** ve **edi'nin** bir satır aşağı inmesi sağlanır.
- b- **pop edx** ve **pop ecx** ile daha önce **stack'e** atılan **edx** ve **ecx** registerları **stackten** çekilir.
- c- **loop resimSatirLoop** ile loop sonlandırılır.

## 9- Ek diziyi ana resim dizisine kopyalama

```
197      mov ebx, tmp
198      mov esi, resim_org
199      mov ecx, n
200      loop1 : mov ax, [ebx]
201      mov[esi], ax
202      add esi, 2
203      add ebx, 2
204      loop loop1
```

- a- **mov ebx,tmp/mov esi, resim\_org** komutları ile **ebx** registerine **tmp** dizisinin, **esi** registerine ise **resim\_org** dizisinin adresi atanır.
- b- **mov ecx, n** komutu döngünün adım sayısını **ecx** registerine atar.
- c- **loop1: mov ax,[ebx]** komutu **tmp** dizisinin mevcut indexindeki değeri **ax'e** atar.
- d- **mov [esi], ax** komutu ile **resim\_org** dizisinin mevcut indexine **ax** registerindeki değer yazılır.
- e- **add esi,2/add ebx,2** komutu **ebx** ve **esi** registerlerini bir sonraki değere geçirir.
- f- **loop loop1** komutu döngüyü sonlandırır

## 8- Kodun Çalıştırılması Sonucu Üretilen Resimler

### 1- Erosion (filter\_size = 3)



### 2- Erosion (filter\_size = 5)



### 3- Erosion (filter\_size = 7)



### 4- Dilation (filter\_size = 3)



### 5- Dilation (filter\_size = 5)



### 6- Dilation (filter\_size = 7)

