

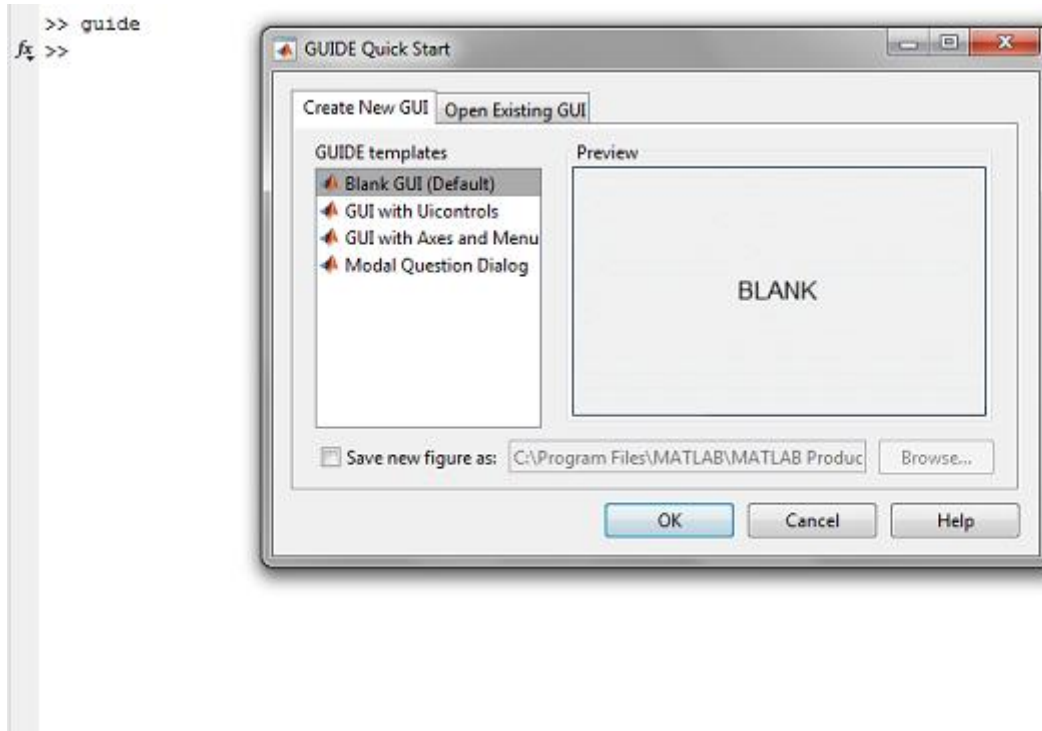
ALGORİTMALAR VE PROGRAMLAMA

DENEY-4: MATLAB’da Grafiksel Kullanıcı Arayüzü

GUIDE MATLAB’ın GUI tasarımcılarına sunduğu içerisinde çeşitli araçlar içeren ve kolaylık sağlayan bir grafiksel GUI geliştirme ortamıdır. GUIDE kullanılarak tıkla ve sürükle-bırak tekniği ile GUI arayüzüne nesneler (örneğin butonlar, text kutuları, liste kutuları, grafikler vs.) kolaylıkla eklenebilir. Ayrıca, eklenen nesnelerin hizalanması, tab sırasının değiştirilmesi, görsel ayarlar üzerinde manipülasyonlar yapılması da bu ortamın tasarımcılara sunduğu imkânlardan bazılarıdır.

Bu aracı çalıştırmak için ya MATLAB komut satırından GUIDE komutu verilir. Bu adımdan sonra karşımıza Şekil 1’deki gibi bir pencere gelir.

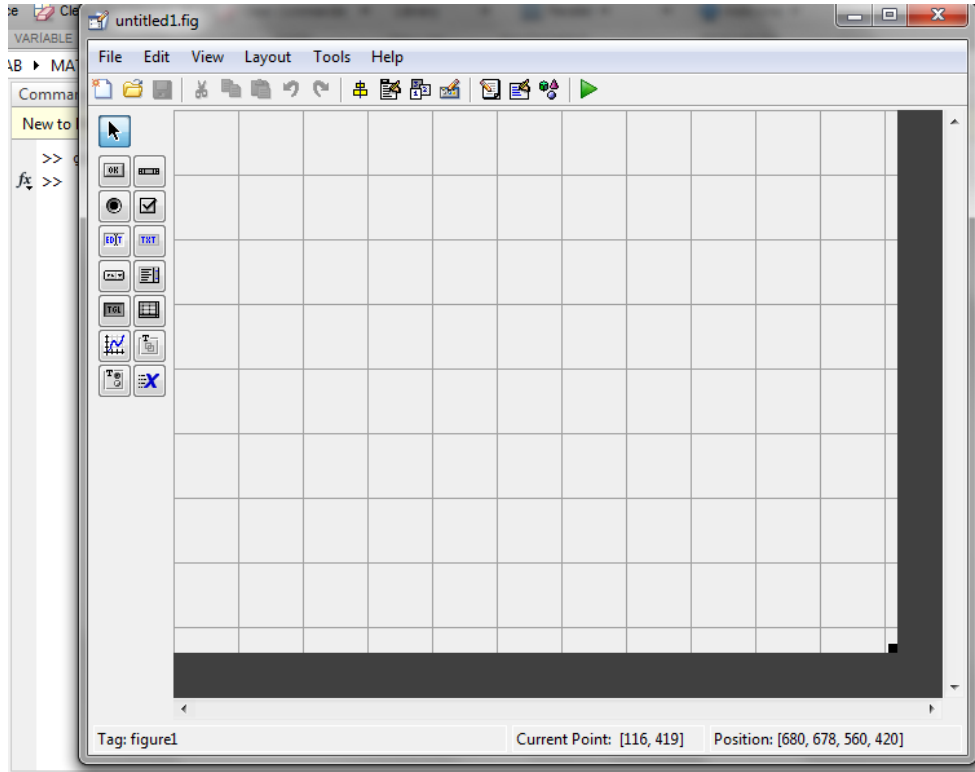
```
>> guide
```



Şekil 1.

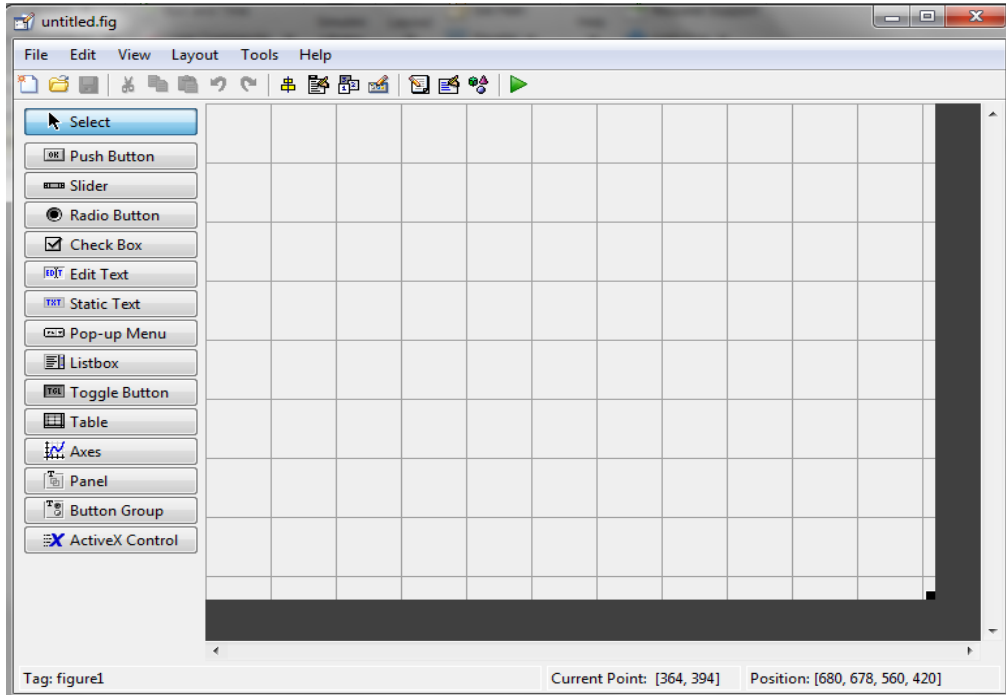
Bu pencereden eğer yeni bir GUI tasarımı yapacak isek Blank GUI seçeneğini seçeriz. Şayet önceden yapılmış bir tasarımı açmak istiyor isek Open Existing GUI sekmesinden sonra istenilen dosyayı seçeriz.

Burada yeni bir tasarım oluşturulacağı için Blank GUI seçeneği seçilerek OK düğmesi tıklanılır. Böylelikle Şekil 2’deki GUIDE LAYOUT Editor (GUIDE Çalışma Alanı) penceresine ulaşırız.



Şekil 2.

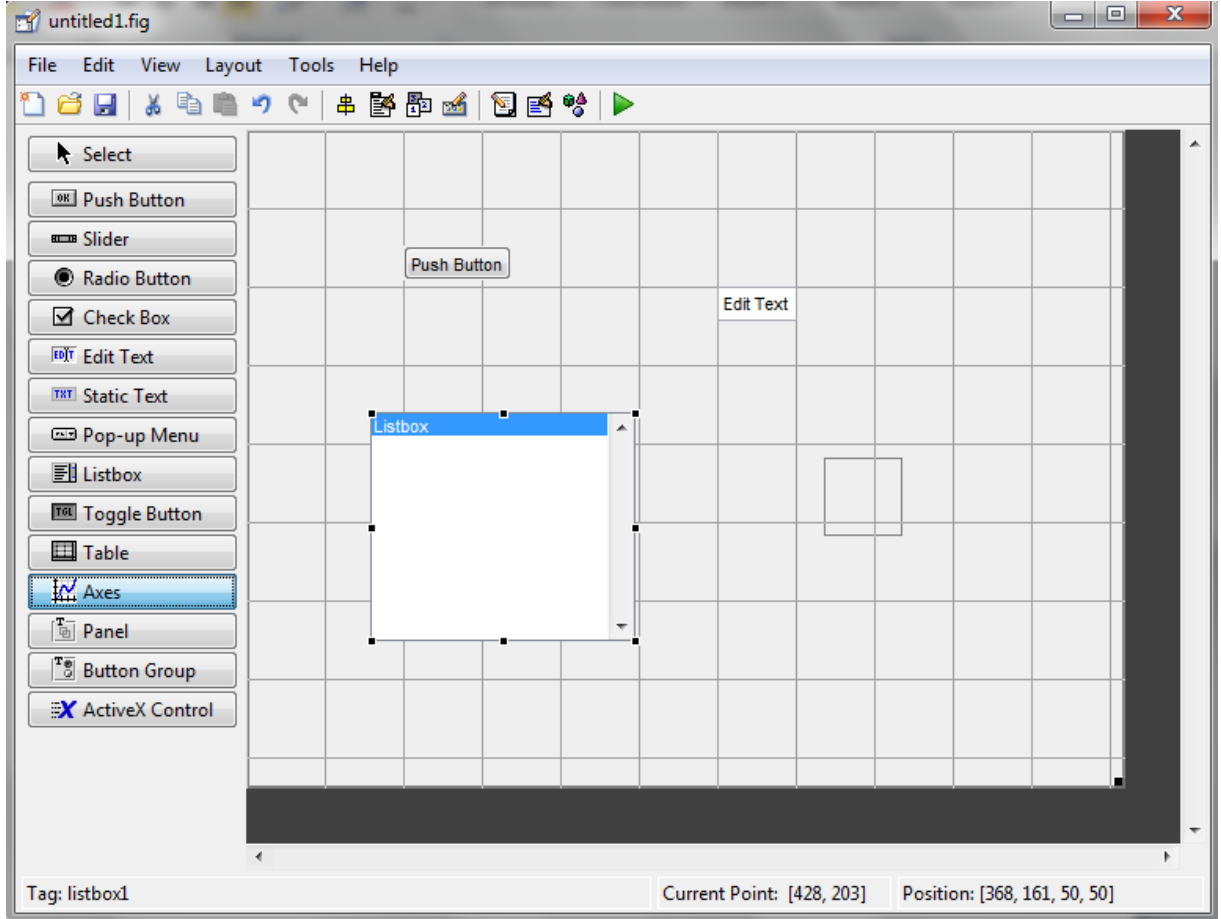
Bu adımdan sonra File/Prefences/Guide yolu kullanılarak gelen pencerede “Show names in component palette” seçeneği tıklanarak OK düğmesine basılır ise karşımıza Şekil 3’teki gibi bir pencere gelecektir.



Şekil 3.

KOMPONENTLERİ ÇALIŞMA ALANINA EKLEME

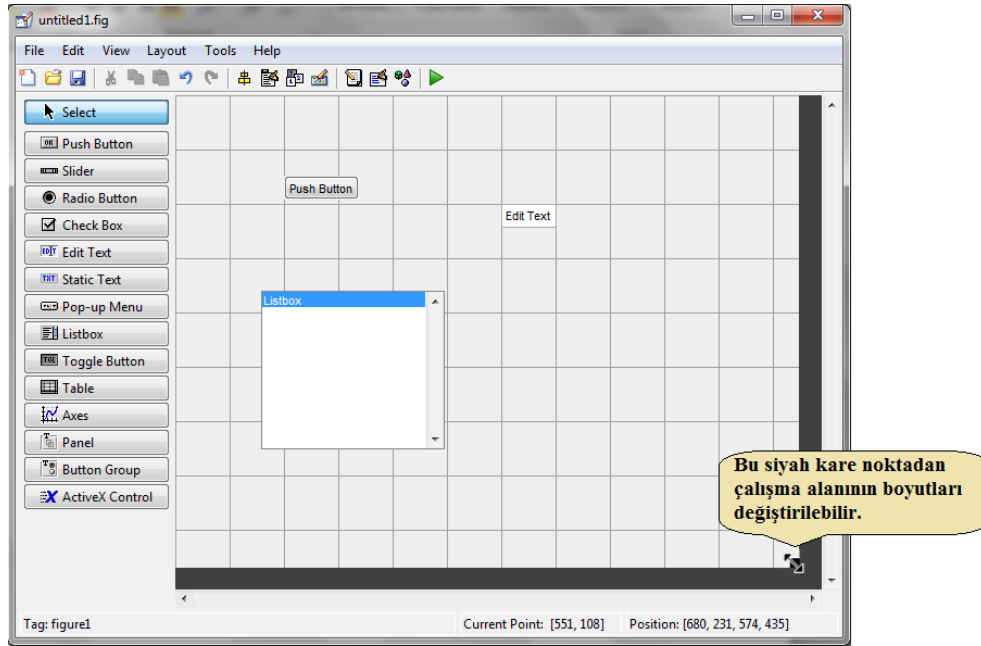
Bunun için sol tarafta bulunan nesne butonlarından istenilen nesneye ait buton tıklanır ve daha sonra çalışma alanında uygun görülen bir noktaya tıklandığında o noktaya ilgili nesne eklenir. İstenirse çalışma alanındaki bir nesne farenin sol tuşu ile tıklanıp bırakılmadan çalışma alanının herhangi bir yerine sürüklenebilir. Bu durum Şekil 4'te de görülmektedir.



Şekil 4.

ÇALIŞMA ALANININ BOYUTLARINI DEĞİŞTİRMEK

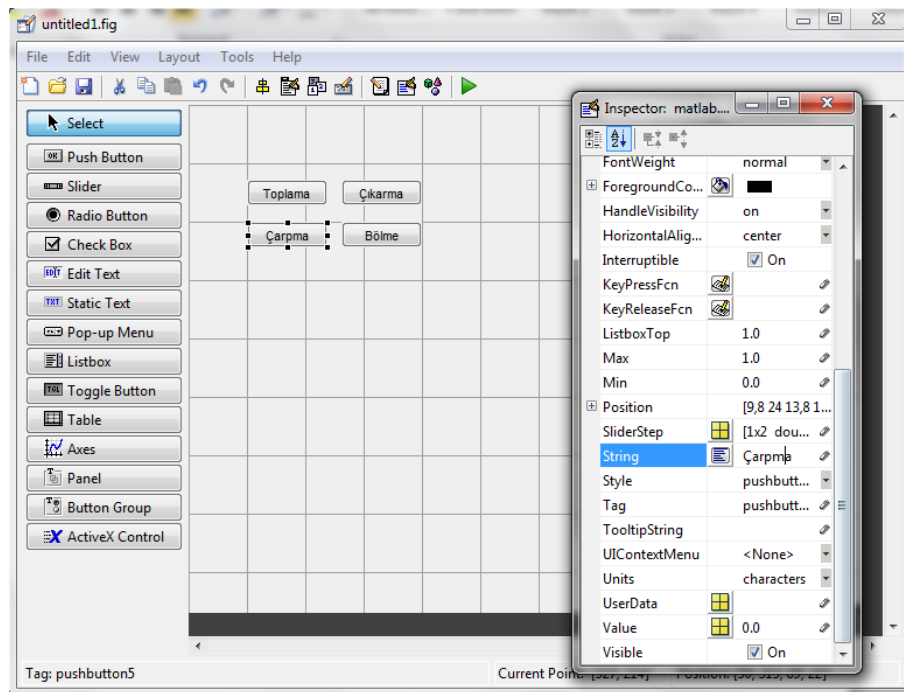
Burada da çalışma alanının sağ alt tarafında bulunan siyah karenin üzerine fare işaretçisi getirilir ve fare işaretçisi konum değiştirdiğinde farenin sol tuşu basılı tutularak çalışma alanı istenilen boyutlarda olacak şekilde düzenleme yapılabilir. Bu durum Şekil 5'te gösterilmektedir.



Şekil 5.

NESNELERE YAZI EKLEME VE ÖZELLİKLERİNİ DEĞİŞTİRME

Nesnelerin özelliklerini değiştirmek istersek ilgili nesne fare ile çift tıklanır veya ilgili nesnenin üzerine gelip fare ile sağ tuş ile tıklanarak Property Inspector seçilir ya da ilgili nesne önce seçilip daha sonra View/Property Inspector komutu ile özellikler penceresi açılır. Burada GUI arayüzüne eklenen pushbutton'ların String değerleri Toplama, Çıkarma, Çarpma ve Bölme olarak değiştirilecektir. GUI arayüzü penceresi Şekil 6'daki gibi görünecektir.



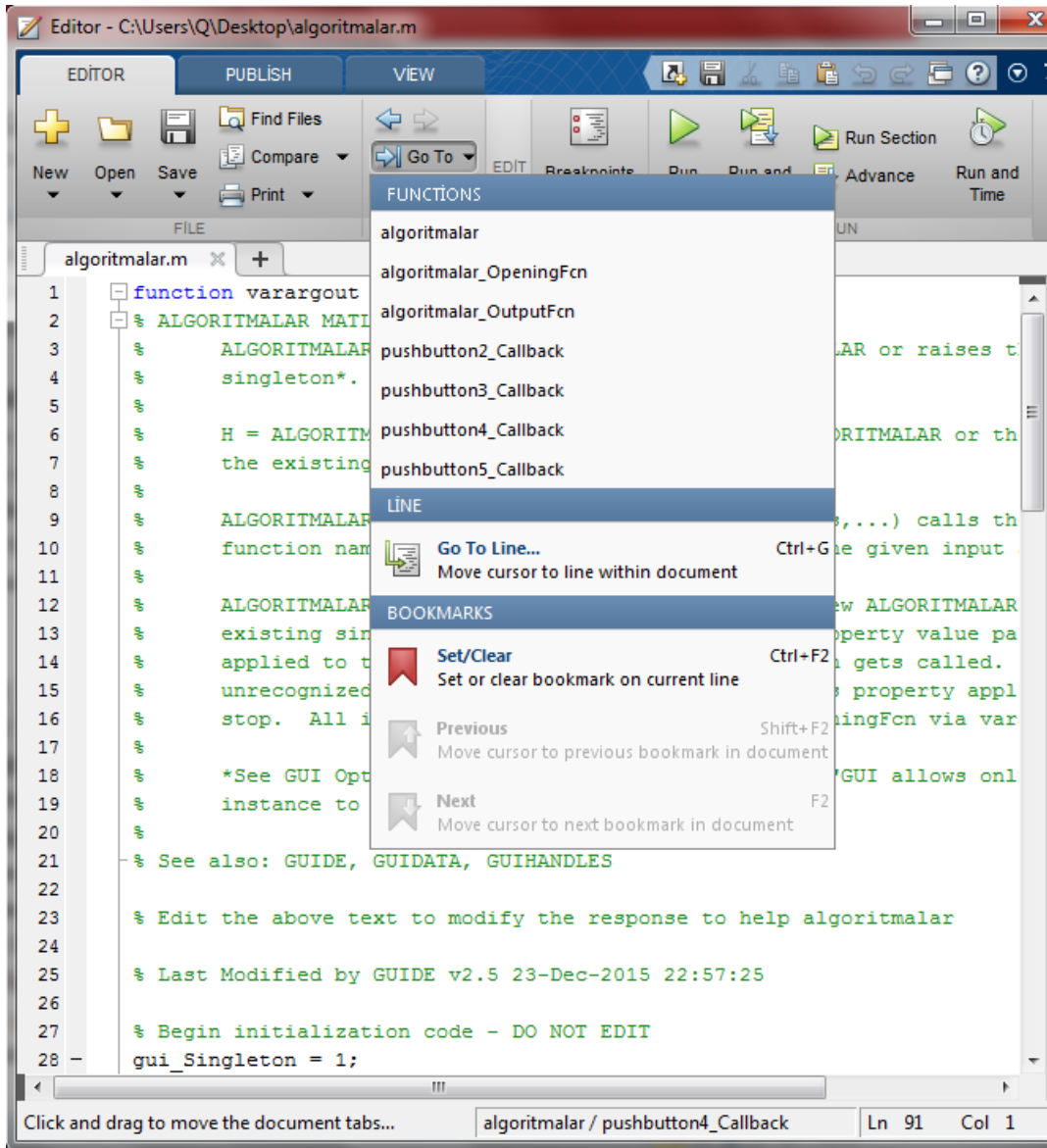
Şekil 6.

GUI TASARIMINI KAYDETME VE ÇALIŞTIRMA

Bitmiş olan bir GUI arayüzünü çalıştırarak görmek için öncelikle Tools/Run yolundan Run (Çalıştır) komutu verilir. Burada çalışmanın Run edilebilmesi için kaydedilmesi gerektiğini bildiren bir pencere çıkar. Yes butonuna tıklayarak çalışmanın kaydedilmesini seçeriz. Bu adımdan sonra MATLAB GUIDE bize tasarımın kaydedileceği dosya ismini soran bir pencere getirir. Bu pencereden çalışmamıza bir isim vererek tasarımı kaydetmiş oluruz.

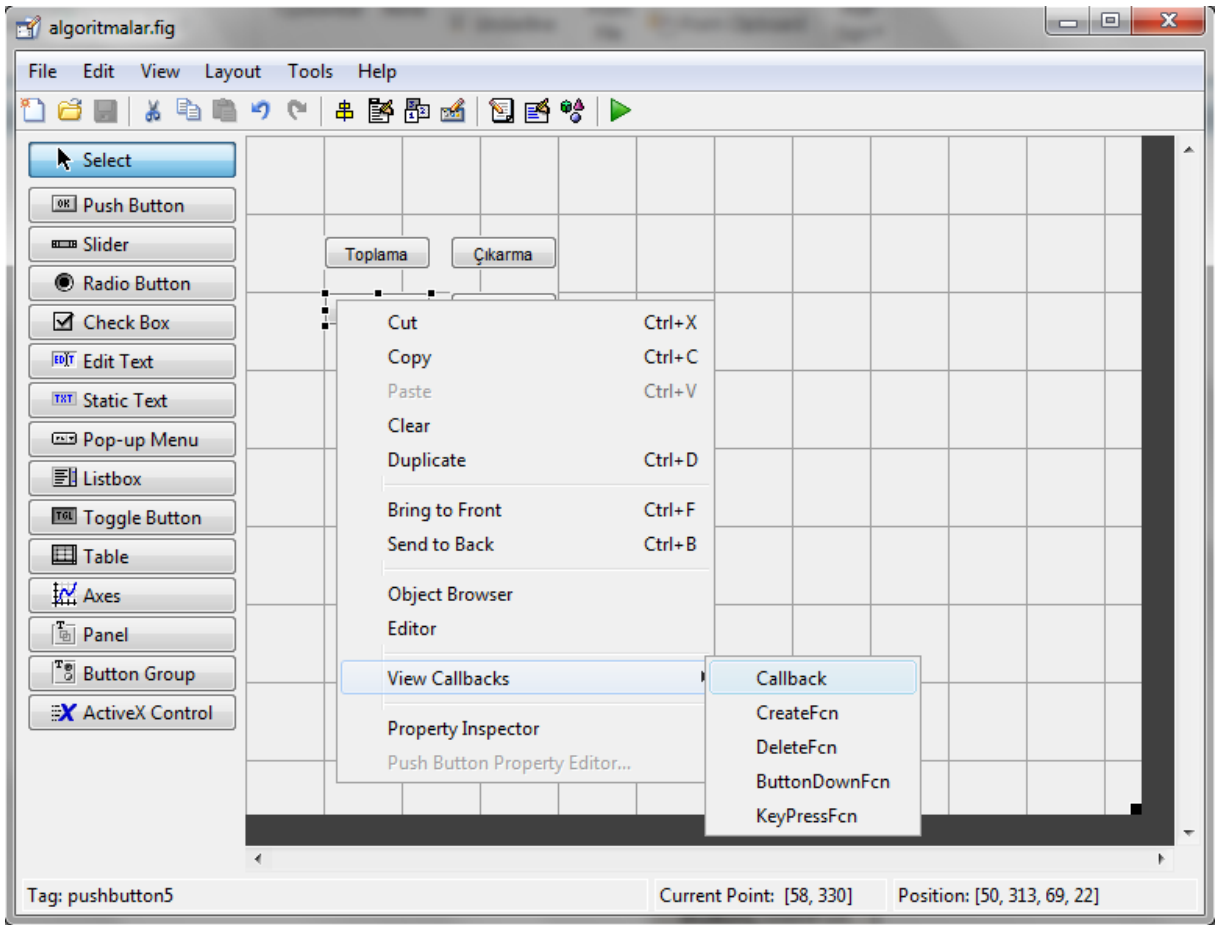
GUI ARAYÜZÜNÜN PROGRAMLANMASI

Bir GUI arayüzünün programlanması demek o çalışmanın kaydedildiği isimle aynı zamanla oluşturulan .m uzantılı dosya içerisine kodlama satırlarının eklenmesi demektir. Bu dosyanın içeriğini görebilmek, değişiklik yapabilmek için GUIDE çalışma ekranı penceresinden View/Editor komutu işletilebilir veya herhangi bir nesnenin üzerine gelerek farenin sağ tuşu ile tıklanarak Editor seçeneği seçilebilir. Karşımıza Şekil 7’deki gibi bir pencere gelecektir.



Şekil 7.

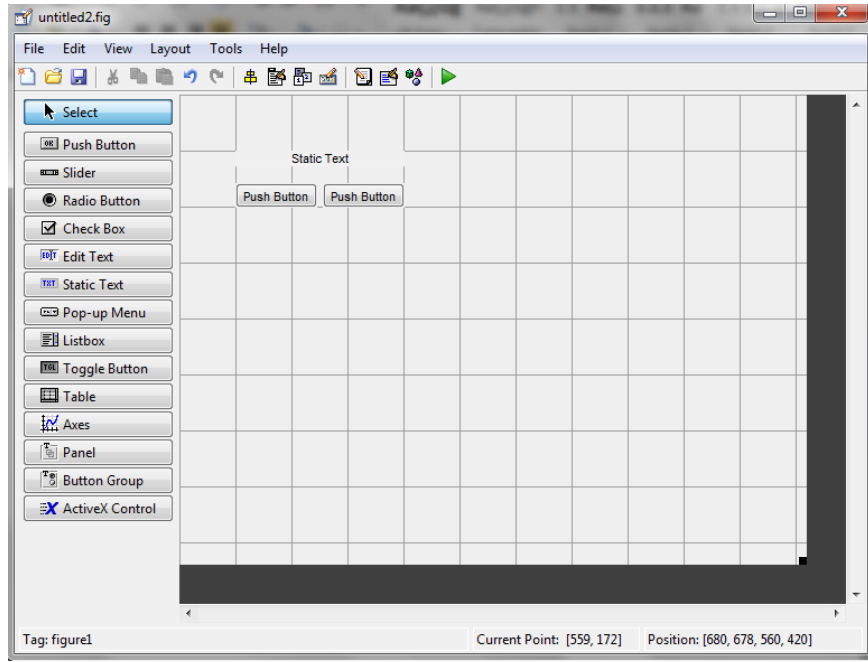
Şekil 7’deki pencerede hazırlamış olduğumuz GUI tasarımına ait kodlar gözükmemektedir. Burada pek çok kodun hazır eklenmiş olduğu görülecektir. Bu kodlar otomatik olarak MATLAB GUIDE tarafından eklenmiştir. Biz burada ilgili butonlara ve liste kutularına ya da istenilen bir nesneye ait callback isimli alt program parçalarına ilgili kodları yazacağız. Bir nesneye ait callback in bulunduğu satıra gitmek için araç çubuğunda yer alan “Go To” butonuna tıklanır ve açılan listeden ilgili nesneye ait callback’ın ismi seçilir. Bu durum Şekil 7’de de görülmektedir. Ayrıca, GUIDE çalışma ekranından da direkt istenilen bir callback satırına gidilebilir. Bunun için ilgili nesne üzerinde sağ tıklanır ve açılan pencereden View Callbacks/Callback menüsünden ilgili callback tıklanması ya da ilgili nesne seçilip View/View Callbacks/Callback yolu üzerinden gidilebilir. Bu durum Şekil 8’de görülmektedir.



Şekil 8.

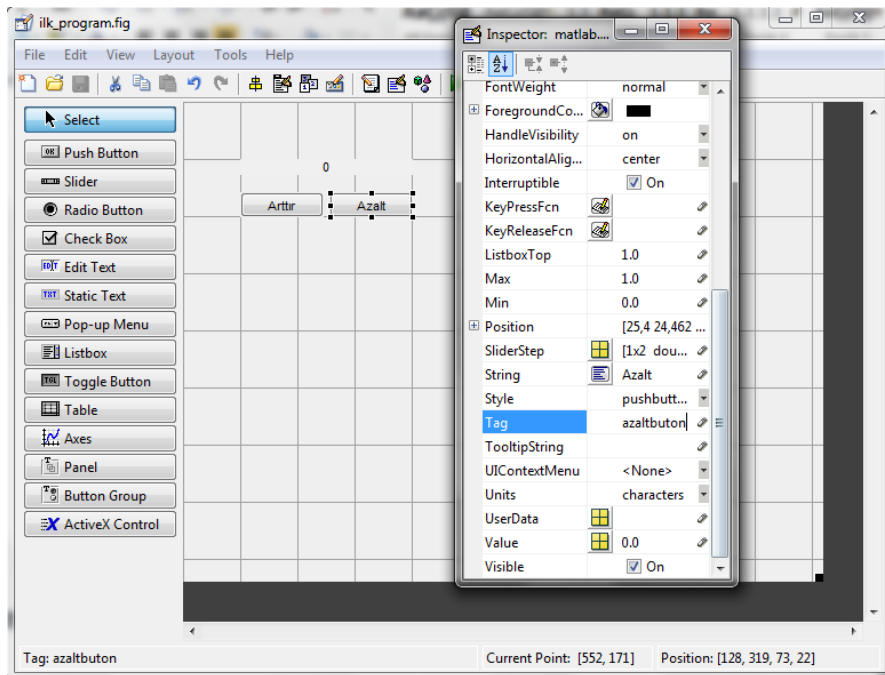
MATLAB GUI’de İLK PROGRAM

İlk olarak MATLAB GUI’de buton kontrollü bir sayıcı programı yazalım. Bunun için sayıcıyı arttırmak ve azaltmak için 2 pushbutton ve o anda elde edilen sayıyı göstermek için bir Static Text’e ihtiyaç duymaktayız. Nesneleri eklediğimiz GUIDE çalışma ekranı Şekil 9’da görülmektedir.



Şekil 9.

Pushbutton'ların String özellikleri nesneler çift tıklanarak açılan Property Inspector kısmında Arttır ve Azalt olarak, Static Text ise sayıcı 0 değerinde başladığı için 0 olarak belirlenir. Benzer şekilde daha anlaşılır bir programlama yapılabilmesi için pushbutton'ların "Tag" özellikleri Property Inspector penceresinde sırası ile arttırbuton ve azaltbuton olarak ayarlanır. Nesnenin Tag özelliğinin değiştirilmesi ile .m uzantılı programda yer alan callback'ler bizim verdiğimiz isimler ile gösterilmesi sağlanır. Böylelikle GUIDE çalışma ekranında figure kısmı tamamlanır ve figure kaydedilir. Şekil 10'da gösterilmektedir.



Şekil 10.

Figure'nin kaydedilmesi ile açılan editör penceresine kendi program satırlarımızı ekliyoruz. Bunun için callback'lerin bulunduğu komut satırlarına gidiyoruz. Tag özelliğini arttırbuton olarak belirlediğimiz callback'e giderek sayıcı değerini 1 arttıran program parçasığını callback'in içine yazıyoruz. Program kodları aşağıda verilmektedir.

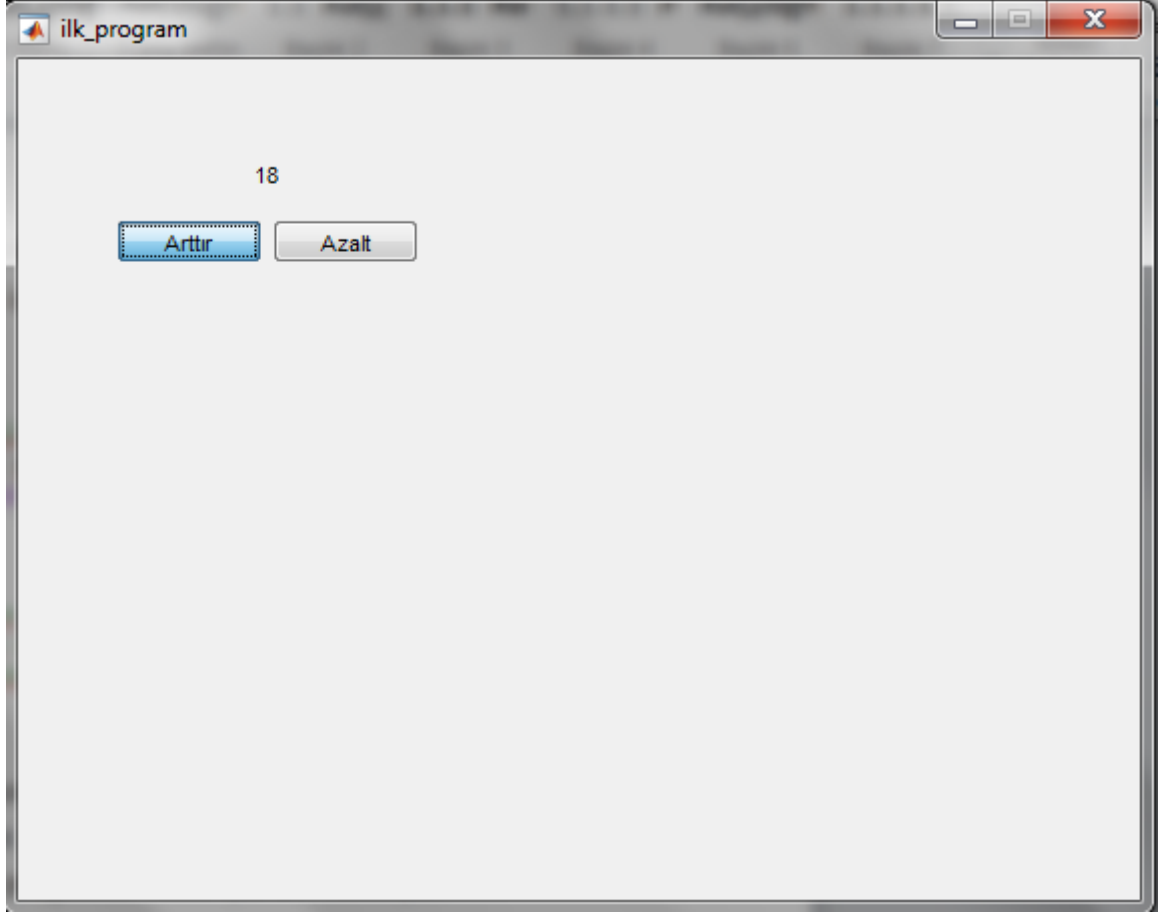
```
function arttirbuton_Callback(hObject, eventdata, handles)
str_sayici = get(handles.text2, 'String');
sayici = str2double(str_sayici);
sayici = sayici + 1;
set(handles.text2, 'String', sayici);
% hObject    handle to arttirbuton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
```

Burada ilk olarak Static Text'te yazılı değer str_sayici adlı değişkene aktarılıyor. Bunun için MATLAB GUIDE'da yer alan get() komutu kullanılmaktadır. Bu komut ile Property Inspector bölümünde yer alan nesneye ait tüm özellikler okunabilir ve bir değişkene aktarılabilir. Biz de bu komutu kullanarak Static Text'in String özelliğini str_sayici değişkenine aktardık. Burada dikkat edilmesi gereken nokta ise hangi nesnenin özelliğini okuyacağımızı handles komutu ile belirlememizdir. Daha sonra string olan bu değeri double türüne çevirerek 1 arttırıyoruz. 1 arttırılan değeri son olarak yine Static Text'e yazdırıyoruz. Bunun için ise set() komutunu kullanıyoruz. Yine get() komutuna benzer olarak set() komutunu kullanarak Property Inspector bölümündeki özellikleri program satırını kullanarak değiştirebiliriz. Bu programda da 1 arttırılmış değeri Static Text'in String özelliğine set() komutunu kullanarak yazdırıyoruz.

Bu kez Tag özelliğini azaltbuton olarak belirlediğimiz callback'e giderek sayıcı değerini 1 azaltan program parçasığını callback'in içine yazıyoruz. Program kodları aşağıda verilmektedir.

```
% --- Executes on button press in azaltbuton.
function azaltbuton_Callback(hObject, eventdata, handles)
str_sayici = get(handles.text2, 'String');
sayici = str2double(str_sayici);
sayici = sayici - 1;
set(handles.text2, 'String', sayici);
% hObject    handle to azaltbuton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
```

Bu program parçasığı da arttırma işlemine benzer mantık ile oluşturulmuştur. Böylelikle sayıcı programımız tamamlanmıştır. Editör penceresinde yer alan Run butonuna tıklanarak program çalıştırılabilir. Şekil 11'de programın ekran çıktısı verilmiştir.



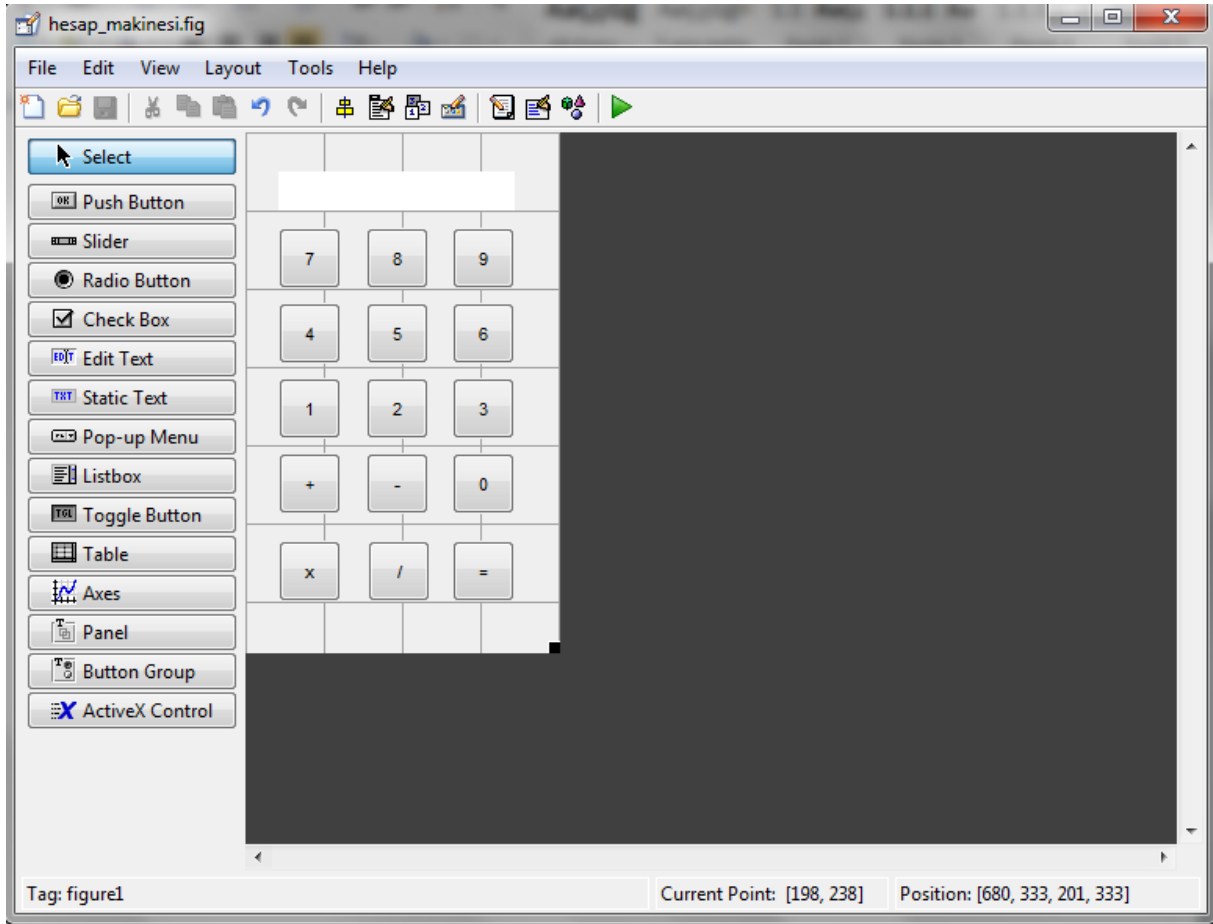
Şekil 11.

MATLAB GUI'de HESAP MAKİNESİ TASARIMI

MATLAB GUI'de hesap makinesi tasarımı için takip edilecek işlem adımları şöyle olacaktır.

1. Komut satırına guide yazılır ve açılan pencerede Blank GUI seçeneği seçilerek boş bir proje açılır.
2. GUIDE penceresinde yer alan nesneler (Push Button, Static Text, Edit Text vb.) kullanılarak program arayüzü hazırlanır.
3. Her bir nesneye ayrı ayrı çift tıklanarak Property Inspector bölümünden String ve Tag özellikleri değiştirilir. İsteğe bağlı olarak diğer özellikleri de değiştirilebilir.
4. Figure kaydedilerek editör penceresi açılır.
5. Her bir callback için gerekli kod parçacıkları yazılır ve program çalıştırılır.

Şekil 12'de GUIDE penceresinde hazırlanmış hesap makinesi arayüzü yer almaktadır.



Şekil 12.

Burada “ = “(eşittir) butonu dışındaki tüm butonların callback’ine her bir sayıya ve işleme uygun olacak şekilde aşağıdaki kod parçacığını yazıyoruz.

```
function birbuton_Callback(hObject, eventdata, handles)
str_sayi = get(handles.ekran, 'String');
str_sayi = strcat(str_sayi, '1');
set(handles.ekran, 'String', str_sayi);
% hObject    handle to birbuton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
```

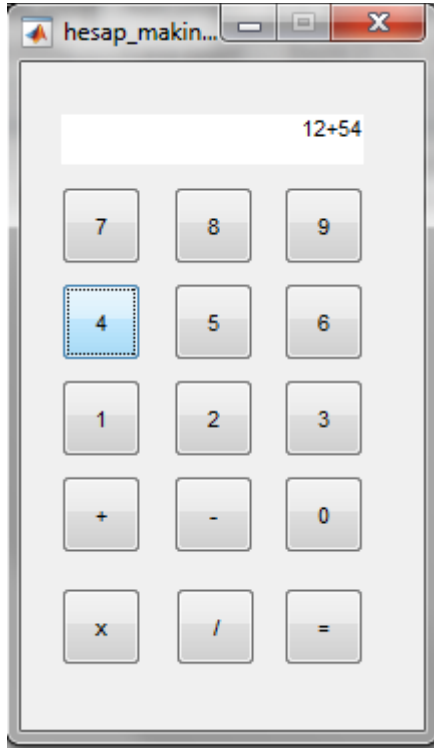
İlk olarak get() komutu ile ekrandaki değer str_sayi değişkenine aktarılıyor ve str_sayi değişkeni 1 butonuna basıldığı için strcat() komutu ile 1 sayısı ile birleştiriliyor. Son olarak set() komutu ile ekranda gösteriliyor.

“=” (eşittir) butonuna ise aşağıdaki kod parçacığını yazıyoruz.

```
function sonucbuton_Callback(hObject, eventdata, handles)
str_sayi = get(handles.ekran, 'String');
str_sayi = eval(str_sayi);
set(handles.ekran, 'String', str_sayi);
% hObject    handle to sonucbuton (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
```

Burada ise diğer butonlardan farklı olarak 2. Satırda birleştirme işlemi yapmak yerine MATLAB kütüphanesinden yararlanılarak eval komutu ile ekrana girilen işlemler gerçekleştirilmektedir. Programın ekran çıktısı Şekil 13 ve Şekil 14’de gösterilmektedir.



Şekil 13.



Şekil 14.

SORULAR

1. Şekil 13’te verilen hesap makinesini gerçekleyiniz.
2. 1. Soruda gerçeklediğiniz hesap makinesine ekranı temizleme butonu ekleyiniz.
3. Bu hesap makinesine “ (“ ve “) “ butonları ekleyerek işlemlere öncelik atanmasını sağlayınız.
4. Hesap makinesine girilen sayının 10 tabanında logaritmasını alan, derece cinsinden cosinus değerini hesaplayan ve faktöriyelini alan 3 adet buton ekleyiniz.

MATLAB’de DOSYA İŞLEMLERİ

Program sonuçlarının otomatik olarak farklı bir dosyaya yazdırılması veya bir dosyadaki bilgilerin okunarak program içerisinde kullanılması, programcılıkta oldukça sık başvurulmuş çıktı alma ve veri girişi yöntemleridir. Dosya yazdırma, çıktı almaya; Dosya okuma ise veri girişine karşılık olan işlemlerdir. Bu işlemler için MATLAB’da kullanılan bazı fonksiyonlar aşağıda sırasıyla verilmiştir.

Matlab’de dosya yazdırma, en basit biçimde, **diary** komutuyla gerçekleştirilir. Kullanımı ise aşağıdaki biçimdedir;

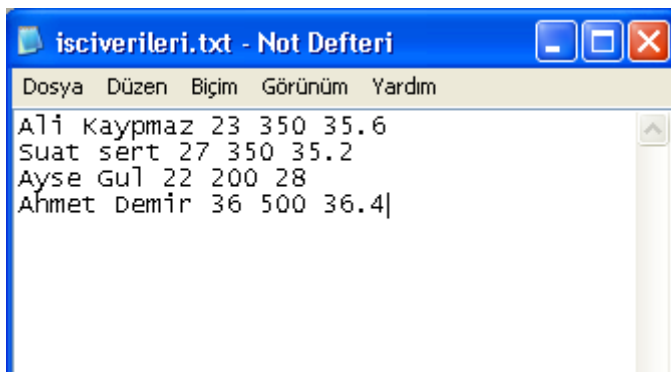
```
clear all;  
for a=1:100;  
    x(a)=a^2;  
end  
diary sonuc.txt  
    disp('-----');  
    disp('Kare Alma İşlemi');  
    disp(x);  
diary end
```

MATLAB’da kullanılan diğer bir fonksiyon textread fonksiyonudur.

textread: Bu fonksiyon ASCII dosyaları okumak için kullanılır. Genel yazım biçimi;

[a, b, c, ...]=textread(‘dosyaadi’, ‘format’, N)

“dosyaadi” açılmak istenen dosyanın ismidir. “format”, herbir sütundaki verinin açıklamasını içeren bir katardır. N ise okunacak satır sayısıdır. N belirtilmemişse dosya sonuna kadar okunur. İlk olarak Şekil 15’de verilen “isciverileri.txt” dosyasını not defteri programını kullanarak oluşturunuz ve MATLAB’da yazdığınız programları çalıştırdığınız klasörün altına kaydediniz.



Şekil 15.

Komut satırında aşağıdaki komutu çalıştırınız.

```
>> [Adi, Soyadi, Yasi, Maasi, Vergiorani]=textread('isciverileri.txt', '%s %s %f %f %f')
```

Soru: Aynı veri dosyasını kullanarak, işçilerin aldığı maaşların ortalamasını hesaplayan ve işçinin aldığı maaşın ortalama maaş miktarından fazla ve az olduğunu belirleyen ve ekrana yazdıran programı yazınız.

Daha gelişmiş dosya yazdırma, fopen, fprintf ve fclose fonksiyonlarının kullanımı ile gerçekleştirilir. fopen, program çıktılarının yazdırılacağı dosyayı açar, fprintf yazdırır ve fclose ise yazdırma işlemini sonlandırır. Genel yazım biçimi;

```
ifade=fopen('dosya_adi', 'izin')  
fprintf(ifade,'aciklama',değişken)  
fclose(ifade)
```

fopen fonksiyonundaki “dosyaadi” açılmak istenen dosyanın ismidir. “izin” ise erişim modunu belirten bir kataradır (string). İzin modları Çizelge-1’de verilmiştir.

Çizelge-1

İzin Formatı	Açıklama
‘r’	Sadece okuma için varolan bir dosyayı açar.
‘rt’	Okuma ve yazma için varolan bir dosyayı açar.
‘w’	Yazma modunda dosya açar; dosya yoksa oluşturulur ve daha önce böyle bir dosya varsa ondaki bilgiler yitirilir.
‘wt’	Varolan bir dosyanın içindekilerini siler ve okuma ve yazma işlemi için açar
‘a’	Varolan bir dosyayı yalnızca yazma işlemi için açar ve yeni yazılanları dosyanın sonuna ekler.
‘a+’	Varolan bir dosyayı, okuma ve yazma işlemi için açar ve yeni verileri dosyanın sonuna ekler.

Fahrenheit ve santigrat değerleri arasında istenilen bir aralıkta dönüşüm yapan ve sonuçları sıcaklikdonusumu.txt dosyasına yazdıran aşağıdaki kodu yazınız ve elde edilen sıcaklikdonusumu.txt dosyasını inceleyiniz.

```
Tbasla=input('Ilk sıcaklık degerinin yaziniz:');  
Tson=input('Son sıcaklık degerinin yaziniz:');  
nTemp=input('Kac deger istediginizi giriniz:');  
santigrat=linspace(Tbasla,Tson,nTemp);  
fahrenheit=1.8*santigrat+32;  
fid=fopen('sicaklikdonusum.txt','wt');  
fprintf(fid,'sicaklik donusum tablosu\n');  
fprintf(fid,'-----\n');
```

```
fprintf(fid,' santigrat fahrenheit\n');
for k=1:nTemp
    fprintf(fid, '%f %f \n', santigrat(k), fahrenheit(k) );
end
fclose(fid);
```

Örnek: a=[3.12356 4.12456 1;5.8463 6.45111 2;4 5 6] biçiminde verilen bir a matrisini, elemanları virgülden sonra 4 hane olacak biçimde, mat.out dosyasına yazdıran aşağıdaki programı yazınız.

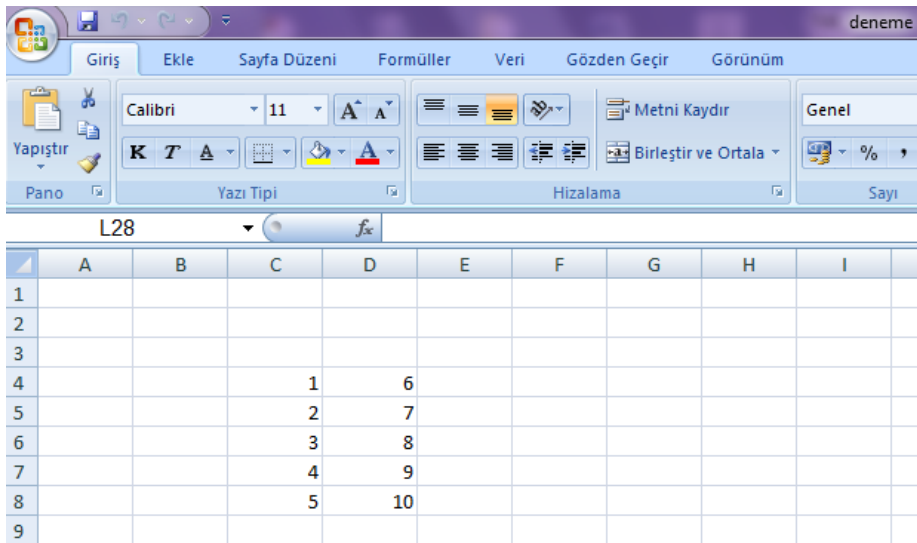
```
a=[3.12356 4.12456 1;5.8463 6.45111 2;4 5 6]
fid = fopen('mat.out','w');
fprintf(fid,'%1.4f %10.4f %10.4f\n',a);
fclose(fid);
```

Veri dosyalarının okunması amacı ile fscanf fonksiyonu kullanılabilir.

[dizi,sayı]=fscanf(ifade,'format',alan) şeklinde kullanımı vardır. Bir önceki uygulamada mat.out dosyasına yazdırılan matrisi tekrar Matlab programında okutulması için aşağıdaki kodu yazınız.

```
fid=fopen('mat.out','rt');
[dizi,sayı]=fscanf(fid,'%f',inf)
```

Excel’ den veri okutmak amacıyla “xlsread” fonksiyonu kullanılır. Bu fonksiyonun incelemesi için Şekil 16’da verilen şekilde bir “deneme.xlsx” dosyasını oluşturunuz ve aşağıdaki kodu yazınız.



	A	B	C	D	E	F	G	H	I
1									
2									
3									
4			1	6					
5			2	7					
6			3	8					
7			4	9					
8			5	10					
9									

Şekil 16.

```
>>A = xlsread('deneme.xlsx', 1, 'C4:D7')
```

SEMBOLİK ARAÇ KUTUSU

MATLAB’da sembolik olarak da çeşitli matematiksel işlemleri gerçekleştirmek için sembolik toolbox (araç kutusu) oluşturulmuştur. Bu araç kutusunda tanımlanan çeşitli fonksiyonlar aşağıda verilmiştir. Sembolik olarak işlem yapmak için ilk olarak mutlaka değişkenlerin sembolik ifade edileceğini göstermek için “syms” fonksiyonu kullanılmaktadır. Aşağıdaki kodu MATLAB’de yazınız ve elde edilen sonuçları inceleyiniz.

```
syms a b c x
s=a*x^2+b*x+c
sonuc=solve(s)
pretty(s)
pretty(sonuc)
```

diff komutu ve int komutu ile türev ve integral işlemlerinin gerçekleştirilmesi

Aşağıdaki komutları MATLAB’de yazınız ve sonuçları inceleyiniz.

Diff fonksiyonu örneği:

```
syms a x
f=(a^2)*(x^3)
sonuc1=diff(f)
sonuc2=diff(f,a)
pretty(sonuc1)
pretty(sonuc2)
```

int fonksiyonu örneği:

```
syms a x
sonuc=int(x^a,x)
pretty(sonuc)
```

Örnek: $f(t) = e^t \cos(3t)$ fonksiyonunun 3. türevini sembolik olarak bulunması ve türevinin $x=-2$ için değerini hesaplanması için aşağıdaki programı yazınız ve sonuçları inceleyiniz.

```
syms t
f=exp(t)*cos(3*t)
f_3u_turev=diff(f,3)
pretty(f_3u_turev)
%türev işlemi ile elde edilen fonksiyonda x=-2
değerinin yerine
%konulması için "subs" fonksiyonu kullanılır.
sonuc=subs(f_3u_turev,t,-2)
```

SIRALAMA ALGORİTMALARI

Verilerin küçükten büyüğe veya tam tersi şekilde sıralama işleminin yapılması önemli veri uygulamalarından bir tanesidir.

Çeşitli sıralama algoritmaları aşağıda verilmiştir;

1. Kabarcık Sıralama (Bubble Sort) Algoritması
2. Araya Sokma (Insertion) Sıralama Algoritması
3. Seçmeli (Selection) Sıralama Algoritması
4. Birleşmeli (Merge) Sıralama Algoritması
5. Hızlı (Quick) Sıralama Algoritması

Kabarcık Sıralama (Bubble Sort) Algoritması

Sıralacak olan veriler üzerinden bir yönden diğer yöne doğru ilerlerken komşu iki verinin yer değiştirilmesi esasına dayanır. Örneğin küçükten büyüğe doğru sıralama işlemi gerçekleştiriliyor olsun. Bu durumda komşu iki veriden sağdaki soldakinden küçük ise yer değiştirilecektir. Böylece komşu iki veriden küçük olan önceki sıraya alınmış olacaktır. Bu işlem n elemanlı bir dizi için $n*(n-1)$ kadar devam edecektir. Kabarcık sıralama algoritmasının kodu aşağıda verilmiştir. Kodu inceleyiniz ve farklı A dizileri için sıralama işlemini gerçekleştiriniz. Bu algorithmada veri boyutu arttıkça sıralama işleminin süresi de üstel olarak artmaktadır.

```
clear all;  
A=[18 7 6 15 4 13]  
n=length(A);  
  
for j=1:1:(n-1)  
    for i=1:1:(n-1)  
        if A(i)>A(i+1);  
            temp=A(i);  
            A(i)=A(i+1);  
            A(i+1)=temp;  
        end  
    end  
end  
disp(A)
```


A=[18 7 6 15 4 13] İşlem adımları sonucunda elde edilecek olan sonuçların bazıları aşağıda verilmiştir.

(Koda göre j=1 için)

1.Adım: [7 18 6 15 4 13] % i=1 iken

2.Adım: [7 6 18 5 4 13] % i=2 iken

3.Adım: [7 6 5 18 4 13] % i=3 iken

4.Adım: [7 6 5 4 18 13] % i=4 iken

5.Adım: [7 6 5 4 13 18] % i=5 iken

(Koda göre j=2 için)

1.Adım: [6 7 5 4 13 18] % i=1 iken

2.Adım: [6 5 7 4 13 18] % i=2 iken

3.Adım: [6 5 4 7 13 18] % i=3 iken

4.Adım: [6 5 4 7 13 18] % i=4 iken

5.Adım: [6 5 4 7 13 18] % i=5 iken

İşlem adımları bu şekilde devam eder..

Araya Sokma Algoritması

Sıralanacak olan dizinin ilk elemanını yerine bırakarak ondan sonraki elemanları alt diziye sırayla alarak sıralamaya uygun olacak olan yere sokma işlemi gerçekleştirilir. İşlem bu şekilde devam eder. Bu işlem adımlarını gösteren çizelge aşağıda verilmiştir.

Sıralanacak dizi: [7 3 5 8 2 9 4 15 6]

Başlangıç durumu: [7][3 5 8 2 9 4 15 6]

Adım	İşlem Öncesi	İşlem Sonrası
1.Adım	[7 3][5 8 2 9 4 15 6]	[3 7][5 8 2 9 4 15 6]
2.Adım	[3 7 5][8 2 9 4 15 6]	[3 5 7][8 2 9 4 15 6]
3.Adım	[3 5 7 8][2 9 4 15 6]	[3 5 7 8][2 9 4 15 6]
...		
8.Adım		[2 3 4 5 6 7 8 9 15][]

Aşağıdaki kodu çalıştırınız.

```

clear all;
a=[7 3 5 8 2 9 4 15 6];
n=length(a);
for i=2:n
    ekle=a(i);
    k=i-1;
    while k>=1 && ekle<=a(k)
        a(k+1)=a(k);
        k=k-1;
    end
    a(k+1)=ekle;
end
a

```

SEÇMELİ SIRALAMA ALGORİTMASI

Bu algoritmada dizi içerisindeki ilk eleman alınır ve daha sonra dizi içerisindeki en küçük eleman aranır. En küçük eleman bulunduğu zaman ilk eleman ile yer değiştirilir. İşlem adımları aşağıda verilmiştir.

Sıralanacak dizi: [7 3 5 8 2 9 4 15 6]

Adım	İşlem Öncesi	İşlem Sonrası
1.Adım	[] [7 3 5 8 2 9 4 15 6]	[2] [3 5 8 7 9 4 15 6]
2.Adım	[2] [3 5 8 7 9 4 15 6]	[2 3] [5 8 7 9 4 15 6]
3.Adım	[2 3] [5 8 7 9 4 15 6]	[2 3 4] [8 7 9 5 15 6]
...		
8.Adım		[2 3 4 5 6 7 8 9 15][]

```

clear all;
a=[7 3 5 8 2 9 4 15 6];
n=length(a);
for i=1:n
    enkucuk=a(n);
    index=n;
    for j=i+1:n
        if a(j)<enkucuk
            enkucuk=a(j);
            index=j;
        end
    end
    a(index)=a(i);
    a(i)=enkucuk;
end
a

```

```

        index=j;
    end
end
a(index)=a(i);
a(i)=enkucuk;
end
a

```

Soru: N kişilik bir sınıfın M tane dersine ait notları klavyeden giriniz.

- a) Her bir dersten sınıf ortalamasını bulan programın MATLAB kodunu yazınız.
- b) Her bir dersten en düşük ve en yüksek not ile bu notu alan öğrenci numaralarını veren programın MATLAB kodunu yazınız.
- c) Kabarcık sıralama algoritmasını kullanarak her bir dersin notlarını büyükten küçüğe doğru sıralayan ve sıralı notları “notlar.txt” dosyasına kaydeden programın MATLAB kodunu yazınız.

Not: N kişilik bir sınıfın M tane dersine ait notları, klavyeden girilerek bellekte $M \times N$ boyutunda A matrisine aktarılmaktadır. Bu matrisin satırlarında dersler, sütunlarında ise öğrenciler yer alır. Ders isimleri ve öğrenci numaraları 1, 2, 3, ... şeklinde adlandırılmaktadır.