TARİH VE ZAMAN

Tarih verisi yıl,ay,gün,saat,dakika,saniye ve microsaniye içerir.

In [1]:
```
# Tarih işlemlerini datetime kütüphanesi üzerinden yapıyoruz.
import datetime
from datetime import datetime
```

In [2]:
```
# Eğer bulunduğumuz anın tarihini kullanmak istiyorsak now() metodunu kullanıyoruz.
x = datetime.now()
print(x)
```

```
2023-09-26 23:11:27.122208
```

In [3]:
```
# Eğer bulunduğumuz anın tarihini kullanmak istiyorsak today() metodunu da kullanabiliriz
y = datetime.today()
print(y)
```

```
2023-09-26 23:11:27.136236
```

datetime kütüphanesini kullanarak datetime object'den bir çok detay bilgiyi çıkarmamızı sağlar.

In [4]:
```
# Tarih verisi oluşturmak
dates = [datetime(2016, 10, 7), datetime(2017, 6, 21)]
```

In [5]:
```
dates
```

Out[5]:
```
[datetime.datetime(2016, 10, 7, 0, 0), datetime.datetime(2017, 6, 21, 0, 0)]
```

In [6]:
```
A=datetime(2016,11,5)
```

In [7]:
```
type(A)
```

Out[7]:
```
datetime.datetime
```

In [8]:
```
# dates listesindeki tarihlerin yıl,gün,ay detaylarını görebiliriz.
print(dates[0].year)
print(dates[1].month)
print(dates[0].day)
print(dates[1].day)
```

```
2016
6
7
21
```

strftime() [string from time]tarih verisinin formatlanması için kullanılır.

In [9]:
```
from IPython.display import display, Image
display(Image(filename='strftime.jpg'))
```

Directive	Description	Example
%a	Weekday, short version	Wed
%A	Weekday, full version	Wednesday
%w	Weekday as a number 0-6, 0 is Sunday	3
%d	Day of month 01-31	31
%b	Month name, short version	Dec
%B	Month name, full version	December
%m	Month as a number 01-12	12
%y	Year, short version, without century	18
%Y	Year, full version	2018
%H	Hour 00-23	17
%I	Hour 00-12	05
%p	AM/PM	PM

strftime() datetime verisini string verisine çevirir. Dolayısıyla format

In [10]:
```
# Tarih formatındaki bilgiler.
print(dates[0])
print(dates[0].month)
print(dates[0].day)
```

```
2016-10-07 00:00:00
10
7
```

In [11]:
```
# strftime() datayı istediğimiz tarih formuna çevirecek ama data tipi string olacaktır.
string_date=dates[0].strftime("%Y-%m-%d")
print(string_date)
print(type(string_date))
```

```
2016-10-07
<class 'str'>
```

In [12]:
```
# Data tipi string'e dönüştüğü için datetime fonksiyonlarını kullanamayız.
string_date.month
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_8308\3838700564.py in <module>
      1 # Data tipi string'e dönüştüğü için datetime fonksiyonlarını kullanamayız.
----> 2 string_date.month

AttributeError: 'str' object has no attribute 'month'
```

In [13]:
```
# Tarih bilgisini sadece kısa gün olarak göstermek istiyorsak
print(dates[0].strftime("%a"))
```

```
Fri
```

In [14]:
```
# Tarih bilgisini sadece tam gün olarak göstermek istiyorsak
print(dates[0].strftime("%A"))
```

```
Friday
```

In [15]:
```
# Tarih bilgisini saedece yıl olarak göstermek istiyorsak
print(dates[0].strftime("%Y"))
```

```
2016
```

In [16]:
```
# Tarih bilgisini yıl-ay-gün şeklinde göstermek istiyorsak
print(dates[1].strftime("%Y-%m-%d"))
```

```
2017-06-21
```

In [17]:
```
datetime(2016, 10, 7)
```

Out[17]:
```
datetime.datetime(2016, 10, 7, 0, 0)
```

In [18]:
```
datetime(2016, 10, 7).month
```

Out[18]:
```
10
```

In [19]:
```
datetime(2016, 10, 7).strftime("%d-%m-%Y").month
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_8308\4245075100.py in <module>
----> 1 datetime(2016, 10, 7).strftime("%d-%m-%Y").month

AttributeError: 'str' object has no attribute 'month'
```

In [20]:
```
# Tarih bilgisini gün-ay-yıl olarak göstermek istiyorsak
print(dates[0].strftime("%d-%m-%Y"))
```

```
07-10-2016
```

In [21]:
```
# Tarih bilgisini ISO formatında göstermek istiyorsak
dates[0].isoformat()
```

Out[21]: '2016-10-07T00:00:00'

In [22]:
```
# ISO formatı sonrasında da datamız string olacaktır
type(dates[0].isoformat())
```

Out[22]: str

In [23]:
```
# Yerel tarih

today = datetime.now()

print(today.strftime("%x"))

```

```
09/26/23
```

In [24]:
```
# Yerel zaman

now = datetime.now()

print(now.strftime("%X"))
```

```
23:12:26
```

Haftanın günleri

0=Monday 1=Tuesday 2=Wednesday . . . 6=Sunday

In [25]:
```
dates[0]
```

Out[25]: datetime.datetime(2016, 10, 7, 0, 0)

In [26]:
```
print(dates[0].weekday())
```

```
4
```

In [ ]:
```

```

Tarih verisi ile matematiksel işlemler

In [27]:
```


# Tarihlerin tanımlanması
d1 = datetime(2017, 11, 5)
d2 = datetime(2019, 12, 4)

# Tanımlanmış tarihverilerinin list içine alınması
dates = [d1, d2]

```

In [28]:
```
min(dates)
```

Out[28]:
```
datetime.datetime(2017, 11, 5, 0, 0)
```

In [29]:
```
# En yakın tarihin seçilmesi
print(min(dates).strftime("%Y-%m-%d"))
```

```
2017-11-05
```

In [30]:
```
# En uzak tarihin seçilmesi
print(max(dates))
```

```
2019-12-04 00:00:00
```

In [31]:
```
# İki tarih arasındaki gün sayısı
delta = d2 - d1
print(delta)
print(delta.days)
```

```
759 days, 0:00:00
759
```

In [ ]:
```

```

In [32]:
```
# Tarihe gün sayısı eklemek
# Tarih verisine gün eklemek ve çıkarmak için timedelta kütüphanesini kullanıyoruz
from datetime import timedelta
# 29 gün timedelta tanımlanması
td = timedelta(days=29)
print(d1 + td)

```

```
2017-12-04 00:00:00
```

In [33]:
```
gun_1 = datetime(2022,10,14)
```

In [34]:
```
print(gun_1)
```

```
2022-10-14 00:00:00
```

relativedelta matematiksel işlemler için diğer bir kütüphane, bir çok açıdan timedelta'dan daha kullanışlıdır

In [35]:
```
# Tarih verisine ay eklemek istersek
# Import relative delta
from dateutil.relativedelta import relativedelta
# 2 ay için relative delta tanımlıyoruz
td = relativedelta(months=2)  # eklemek için pozitif, çıkarmak için negatif tamsayı kullanıyoruz

```

In [36]:
```
d1
```

Out[36]:
```
datetime.datetime(2017, 11, 5, 0, 0)
```

In [37]:
```
datetime.now()+relativedelta(months=2)
```

Out[37]:
```
datetime.datetime(2023, 11, 26, 23, 12, 55, 987717)
```

In [38]:
```
# iki ay sonrası
d1_after_2_months =(d1 + td)
print(d1_after_2_months)
print(d1_after_2_months.strftime("%Y-%m-%d"))
```

```
2018-01-05 00:00:00
2018-01-05
```

In [39]:
```
# iki ay öncesi
d1_before_2_months =(d1 - td)
print(d1_before_2_months)
print(d1_before_2_months.strftime("%Y-%m-%d"))
```

```
2017-09-05 00:00:00
2017-09-05
```

In [40]:
```
# Tarih verisine ay ve yılı beraber eklemek

# d1 tarih verisine 2 ay, bir yıl sonrası
td = relativedelta(months=2, years=1)
print(d1 + td)
```

```
2019-01-05 00:00:00
```

In [41]:
```
# Tarih verisinden ay ve yılı beraber çıkarmak

# d1 tarihinden 2 ay bir yıl öncesi
td = relativedelta(months=-2, years=-1)
print(d1 + td)
```

```
2016-09-05 00:00:00
```

In [42]:
```
# bir yıl iki ay sonrası
d1_after_1_year_2_months =(d1 + td)
print(d1_after_1_year_2_months)
print(d1_after_1_year_2_months.strftime("%Y-%m-%d"))
```

```
2016-09-05 00:00:00
2016-09-05
```

In [43]:
```
# bir yıl iki ay öncesi
d1_before_1_year_2_months =(d1 - td)
print(d1_before_1_year_2_months)
print(d1_before_1_year_2_months.strftime("%Y-%m-%d"))
```

```
2019-01-05 00:00:00
2019-01-05
```

In [44]:
```
# Tarih verisine yıl,ay ve gün sayısının beraber eklenmesi

# Eğer 3 yıl,2 ay ve 10 gün önce tarihin ne olduğunu bilmek istersek
td = relativedelta(days=10, months=2,years=-3)
```

In [45]:
```
print('3 yıl, 2 ay ve 10 gün öncesinin tarihi {} ve günlerden {}.'.format((d1+td).strftime("%Y-%m-%d"),(d1+td).st
```

```
3 yıl, 2 ay ve 10 gün öncesinin tarihi 2015-01-15 ve günlerden Thursday.
```

EĞER ISO FORMATINDA TARİHİ FORMAT DEĞİŞİMİ YAPMADAN GÖSTERMEK İSTERSEK

In [46]:
```
from datetime import date
# Örnek tarihimiz, 25-11-2017, datetime kütüphanesinden date datamızı görsel olarak ISO format: YYYY-MM-DD şeklin
# tarih fonksiyonu sağlayan datetime formatında kalmasını sağlar.
d = date(2017, 11, 25)
print(d)
print(type(d))
```

```
2017-11-25
<class 'datetime.date'>
```

In [47]:
```
# Eğer tarih verisini list içinde göstermek istersek köşeli parantezi kullanıyoruz
print( [d] )
```

```
[datetime.date(2017, 11, 25)]
```

In [48]:
```
# Eğer bir tarih verisini ISO 8601 formatında list içersinde göstermek istersek
print( [d.isoformat()] )

```

```
['2017-11-25']
```

In [49]:
```
# Formatın stringe dönüştüğünü unutmamalıyız
type([d.isoformat()][0] )
```

Out[49]:
```
str
```

STRING TARİH VERİSİNİ datetime FORMATINA ÇEVİRMEK

datetime.strptime() [string pass time]

In [50]:
```
# string olarak elimizde bulunan datamız
string_date ="12/30/2017 15:19:13"
print(type(string_date))
```

```
<class 'str'>
```

In [51]:
```
# Tarih fonksiyonlarını kullanmak istersek, mesela sadece yıl bilgisini görmek istersek veya matematiksel işlem y
# datamızı tarih formatına çevirmemiz gerekiyor
string_date_to_datetime = datetime.strptime("12/30/2017 15:19:13","%m/%d/%Y %H:%M:%S")
print(string_date_to_datetime)
print(type(string_date_to_datetime))
```

```
2017-12-30 15:19:13
<class 'datetime.datetime'>
```

In [52]:
```
# datetime kütüphanesinden datetime.strptime() datamızı date tipinde tutar ve ISO formatında gösterir.
from datetime import datetime
dt = datetime.strptime("12/30/2017 15:19:13", "%m/%d/%Y %H:%M:%S")
print(dt)

```

```
2017-12-30 15:19:13
```

KOŞULLU VE MANTIKSAL İŞLEMLER

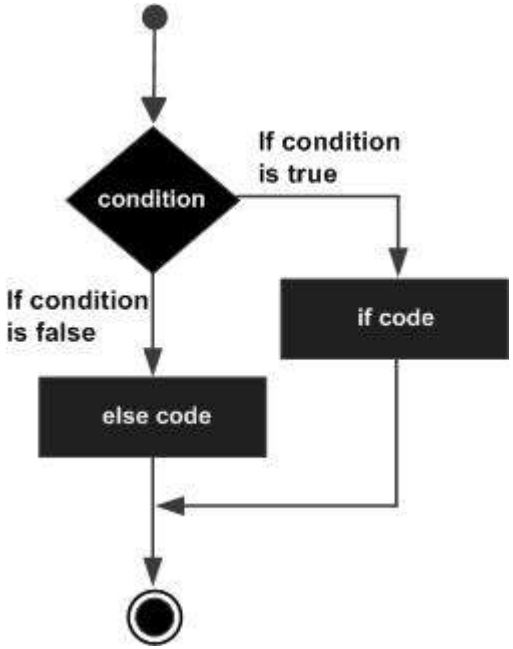
if-elif-else

In [ ]:
```

```

```
if expression: # if sonrasında koşul yer alıyor ve iki nokta ile bitiyor.
    statement(s) # koşul sonucu grintili yazılmalıdır.
else:            # else if ile aynı izada olmalı ve iki nokta ile bitmelidir.
    statement(s) # koşul sonucu grintili yazılmalıdır.
```

In [53]:
```
display(Image(filename='if_else_statement.jpg'))
```



In [54]:
```
# Ders geçme algoritması
final_sınav_sonucu = input("Lütfen final sınav sonucunu giriniz =")
proje_puanı = input("Lütfen proje puanınızı giriniz=")
midterm_sınav_sonucu = input("Lütfen midterm puanınızı giriniz=")
final_sınav_sonucu =int(final_sınav_sonucu)*0.60
proje_puanı = int(proje_puanı)*0.10
midterm_sınav_sonucu = int(midterm_sınav_sonucu)*0.30
total_point = final_sınav_sonucu + proje_puanı + midterm_sınav_sonucu


if total_point >= 70:

    print('Toplam puanınız  {} , Tebrikler geçtiniz'.format(round(total_point)))

else:

    print('Toplam puanınız {} , Üzgünüm kaldınız'.format(round(total_point)))


```

```
Lütfen final sınav sonucunu giriniz =45
Lütfen proje puanınızı giriniz=45
Lütfen midterm puanınızı giriniz=45
Toplam puanınız 45 , Üzgünüm kaldınız
```

```
1
```

In [ ]:
```
1
```

Eğer Birden Fazla Koşullu Loop Yazacaksak Her Koşulu Parantez İçinde Gösteriyoruz ve & (and) veya | (or) İle Bireştiriyoruz

In [55]:
```python
# Hangi Jenerasyondansınız?
doğum_tarihi = input('Lütfen Doğum Tarihinizi Yıl Olarak Giriniz=')

doğum_tarihi = int(doğum_tarihi)
if (doğum_tarihi < 1945):
    print('Kayip jenerasyondansın.')

elif (doğum_tarihi >= 1945) & (doğum_tarihi < 1965) :

    print('Baby boomer jenerasyondansın')

elif (doğum_tarihi > 1965) & (doğum_tarihi <= 1981) :

    print('X jenerasyondansın')

elif (doğum_tarihi > 1981) & (doğum_tarihi <= 1994 ):

    print('Y jenerasyondansın')

elif (doğum_tarihi > 1997) & (doğum_tarihi <= 2012):
    print('Z jenerasyondansın.')

else:
    print('Yaşın çok genç, muhtemelen yanlış okuldasın')

```

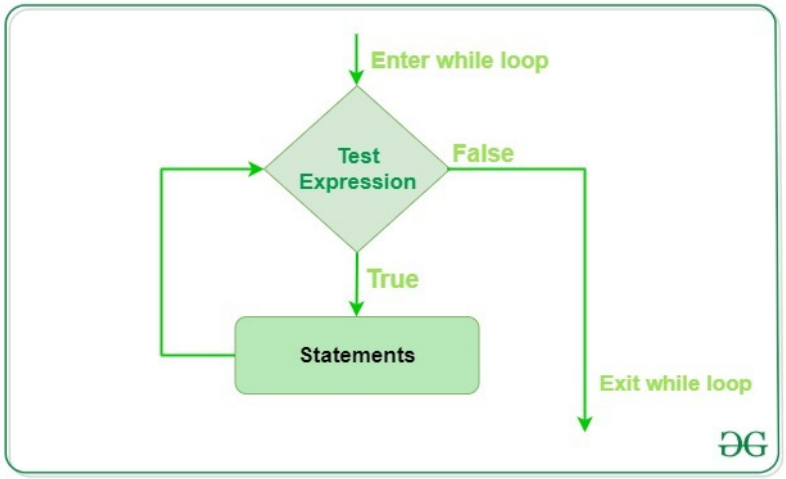
```
Lütfen Doğum Tarihinizi Yıl Olarak Giriniz=1976
X jenerasyondansın
```

LOOP(Döngü): SIRALANABİLİR HERHANGİ NESNEDE TEKRAR EDİLEBİLEN OPERASYONLAR YAPMAK.

WHILE LOOP: KOŞULUMUZUN SAĞLANDIĞIMI MÜDDETÇE ÇALIŞAN DÖNGÜLERDİR.

In [56]:

```
from IPython.display import display, Image
display(Image(filename='1.png'))
```



In [57]:
```
# Kullanıcıdan bir rakam iste
number = 2

# While loop için koşul oluştur, koşıl satırı herzaman : ile biter.
while number < 5 :
    print("Thank you") # İşlem funksiyonunu tanımla. İşlem satırı her zaman bir paragraf geride başlar.

    number = number+1 # while loop koşul gerçekleşinceye kadar devam edeceği için, koşulu bitiren satırı yazıyoru
```

```
Thank you
Thank you
Thank you
```

While loop koşul gerçekleşinceye kadar devam ettiği için, bazı durumlarda koşul gerçekleşse bile loopu durdurmak gerekiyor. bu durumlarda break komudunu kullanıyoruz.

In [58]:
```
#while loop rakam  altıdan küçük olduğu müddetçe yazırılmasını amaçlıyor, fakat aynı zamanda ikinci bir koşulla lo
#loop2un devamlılığını sağlıyan işlem ise verilen sayıya her defasında 1 eklenmesi.
i = 1
while i < 6:
  print(i)
  if i == 3:
    break
  i += 1
```

```
1
2
3
```

Continnue komudu loopu durdurup başka bir işlem sonrası devam edilmek istendiğinde kullanılır.

In [59]:
```
# Python kelimesindeki tüm harfleri yazdır h harfine gelirse döngüyü durdur ve atlayarak tekrar başla
for letter in 'Python':     # First Example
   if letter == 'h':
      continue
   print(letter)
```

```
P
y
t
o
n
```

In [60]:
```
# Verilen sayı 6' dan küçük olduğu müddetçe 1 ekle ve yazdır, sayi 3'e eşit olursa loopun başına dön.
i = 0
while i < 6:
  i += 1
  if i == 3:
    continue
  print(i)
```

```
1
2
4
5
6
```

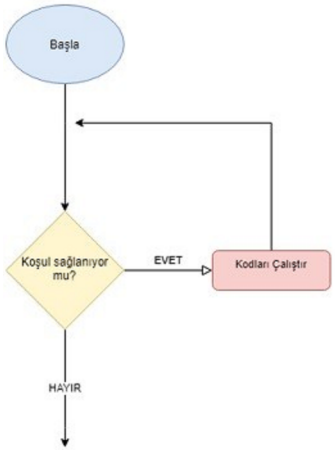
else koşulun oluşmadığı durumu ifade eder.

In [61]:
```
# Verilen sayı 6'dan küçük olduğu müddetçe 1 ekle ve yazdır, eğer koşul gerçekleşmezse yani 6 ya eşit veya büyük
# durumda "i artık 6'dan küçük değildir." şeklinde yazdır.
i = 1
while i < 6:
  print(i)
  i += 1
else:
  print("i artık 6'dan küçük değildir.")
```

```
1
2
3
4
5
i artık 6'dan küçük değildir.
```

for loop

In [62]:
```
display(Image(filename='2.png'))
```



In [63]:
```
#  1 den 10'a  kadar olan rakamları teker toplamını al ve sonunda yazdır
sum = 0
for i in range(1,11):
        sum+=i


print(sum)
```

```
55
```

In [64]:
```
sum = 0
for i in range(1,11):
    sum = sum+i
    print(sum)
```

```
1
3
6
10
15
21
28
36
45
55
```

In [65]:
```
# break döngüyü durdurur
isim_soyisim = input('Lütfen İsim ve Soyisminizi Giriniz=')
for letter in isim_soyisim:
    if letter in ('ç','ğ','ş','ü','ö','Ç','Ü','Ğ','Ş','Ş'):
        print ('Girdiğiniz İsim veya Soyisim Türkçe Karakter Taşıyor Lütfen ç,ğ,ş,ü,ö yerine c,g,s,u,o kullanınız
        break

```

```
Lütfen İsim ve Soyisminizi Giriniz=mehmet bentürk
Girdiğiniz İsim veya Soyisim Türkçe Karakter Taşıyor Lütfen ç,ğ,ş,ü,ö yerine c,g,s,u,o kullanınız.
```

In [66]:
```
# Girilen sayının kaç basamaklı olduğunu gösteren for loop
numara = input('Lütfen bir sayı giriniz=')

numara = str(numara) #İlk olarak girilen rakamı string formatına çeviriyorum

count=0

for i in numara:
        count += 1

print('Girdğiniz sayı '+ str(count)+ ' basamaklı.')
```

```
Lütfen bir sayı giriniz=123
Girdğiniz sayı 3 basamaklı.
```

In [ ]:
```

```

In [67]:
```
şifre = input("Lütfen 8 ile 10 karakterden oluşan ve en az bir tane harf içeren bir şifre giriniz=")
harf_sayısı =0
# Girilen şifre 8 ile 10 karakterden oluşmuyorsa, şifrenin kabul edilmediğini yaz, diğer durumda devam et.
if (len(şifre) < 8) | (len(şifre)>10):

    print("Şifre 8 veya 10 karakter arasında olmadığından dolayı kabul edilmedi")
else:
# şifrenin harf içerip içermediğinin kontrol edilmesi
    for i in şifre:
        if i.isalpha()== True:
            harf_sayısı = harf_sayısı +1
    if harf_sayısı<1:
        print("Şifreniz en az bir harf içermediğinden dolayı kabul edilmedi")
    else:
        print("Şifreniz kabul edildi")


```

```
Lütfen 8 ile 10 karakterden oluşan ve en az bir tane harf içeren bir şifre giriniz=123456789
Şifreniz en az bir harf içermediğinden dolayı kabul edilmedi
```

In [68]:
```
# Aylardaki gün sayısını gösteren döngü
aylar = ["Ocak", "Nisan", "Ağustos","Haziran","Gasım"]

# iterate through each mont in the list
for ay in aylar:
    if ay == "Şubat":
            print("Şubat ayı 28 ile 29 gündür.")
    elif ay in ("Nisan", "Haziran", "Eylül", "Kasım"):
            print(ay+" 30 gündür.")
    elif ay in ("Ocak", "Mart", "Mayıs", "Temmuz", "Ağustos", "Ekim", "Aralık"):
            print(ay+" 31 gündür.")
    else:
            print(ay,"Geçerli bir ay ismi değildir")
```

```
Ocak 31 gündür.
Nisan 30 gündür.
Ağustos 31 gündür.
Haziran 30 gündür.
Gasım Geçerli bir ay ismi değildir
```

In [ ]:
```

```

In [69]:
```
d={'hisse_senetleri':['AAPL','IBM','GOOGL','KO','FB','PG'],
   'kapanış_fiyatları':[160,115,220,65,115,25],
   'sahip_olunan_miktar':[10,15,21,0,5,11],
   'sahip_olunan_hisselerin_değeri':[]}

```

In [70]:
```
d
```

Out[70]:
```
{'hisse_senetleri': ['AAPL', 'IBM', 'GOOGL', 'KO', 'FB', 'PG'],
 'kapanış_fiyatları': [160, 115, 220, 65, 115, 25],
 'sahip_olunan_miktar': [10, 15, 21, 0, 5, 11],
 'sahip_olunan_hisselerin_değeri': []}
```

In [71]:
```
for i in range(len(d['hisse_senetleri'])):
    d['sahip_olunan_hisselerin_değeri'].append( d['kapanış_fiyatları'][i] *d['sahip_olunan_miktar'][i])
```

In [72]:
```
d
```

Out[72]:
```
{'hisse_senetleri': ['AAPL', 'IBM', 'GOOGL', 'KO', 'FB', 'PG'],
 'kapanış_fiyatları': [160, 115, 220, 65, 115, 25],
 'sahip_olunan_miktar': [10, 15, 21, 0, 5, 11],
 'sahip_olunan_hisselerin_değeri': [1600, 1725, 4620, 0, 575, 275]}
```

In [73]:
```
import numpy as np
d['hisselerin_toplam_portföydeki_payı'] = []
```

In [74]:
```
d
```

Out[74]:
```
{'hisse_senetleri': ['AAPL', 'IBM', 'GOOGL', 'KO', 'FB', 'PG'],
 'kapanış_fiyatları': [160, 115, 220, 65, 115, 25],
 'sahip_olunan_miktar': [10, 15, 21, 0, 5, 11],
 'sahip_olunan_hisselerin_değeri': [1600, 1725, 4620, 0, 575, 275],
 'hisselerin_toplam_portföydeki_payı': []}
```

In [75]:
```
np.sum(d['sahip_olunan_hisselerin_değeri'])
```

Out[75]:
```
8795
```

In [76]:
```
for i in range(len(d['hisse_senetleri'])):
    d['hisselerin_toplam_portföydeki_payı'].append((d['sahip_olunan_hisselerin_değeri'][i] / np.sum(d['sahip_olun
```

In [77]:
```
1 d
```

Out[77]:
```
{'hisse_senetleri': ['AAPL', 'IBM', 'GOOGL', 'KO', 'FB', 'PG'],
 'kapanış_fiyatları': [160, 115, 220, 65, 115, 25],
 'sahip_olunan_miktar': [10, 15, 21, 0, 5, 11],
 'sahip_olunan_hisselerin_değeri': [1600, 1725, 4620, 0, 575, 275],
 'hisselerin_toplam_portföydeki_payı': [18.192154633314384,
  19.61341671404207,
  52.529846503695275,
  0.0,
  6.537805571347357,
  3.1267765776009093]}
```

In [78]:
```
1 for i in range(len(d['kapanış_fiyatları'])):
2     for j in ['kapanış_fiyatları','sahip_olunan_miktar','hisselerin_toplam_portföydeki_payı']:
3         d[j][i] = round((d[j][i]),2)
```

In [79]:
```
1 d
```

Out[79]:
```
{'hisse_senetleri': ['AAPL', 'IBM', 'GOOGL', 'KO', 'FB', 'PG'],
 'kapanış_fiyatları': [160, 115, 220, 65, 115, 25],
 'sahip_olunan_miktar': [10, 15, 21, 0, 5, 11],
 'sahip_olunan_hisselerin_değeri': [1600, 1725, 4620, 0, 575, 275],
 'hisselerin_toplam_portföydeki_payı': [18.19, 19.61, 52.53, 0.0, 6.54, 3.13]}
```

for loop zip:EĞER İKİ İTERATOR (SIRALAYICI) AYNI UZUNLUKTAYSA

In [ ]:
```
1
```

In [80]:
```
# Porföyümüzde bulunan hisse senetlerinin payı %15'in üzerindeyse satış listesine, %5'in altındaysa alış listesine
# diğerlerini de elde tutlan listesine kayıt etmek istiyoeuz.
for i,j in zip(d['hisse_senetleri'][:],d['hisselerin_toplam_portföydeki_payı'][:]):
    if j > 15:
        print('Hisse senedi {} satış listesinde olmalı.'.format(i))
    elif (j < 15) & (j > 5):
        print('Hisse senedi {} elde tutulan listesinde olmalı.'.format(i))
    else:
            print('Hisse senedi {} satış listesinde olmalı.'.format(i))

```

```
Hisse senedi AAPL satış listesinde olmalı.
Hisse senedi IBM satış listesinde olmalı.
Hisse senedi GOOGL satış listesinde olmalı.
Hisse senedi KO satış listesinde olmalı.
Hisse senedi FB elde tutulan listesinde olmalı.
Hisse senedi PG satış listesinde olmalı.
```

In [81]:
```
faang_stocks_getileri = {'semboller':['FB','AMZN','AAPL','NFLX','GOOGL'],
                         'fiyatlar':[[338.54,160.03],[170.40,108.92],[181.26,136.53],[597.37,174.87],[145.07,109.3
                        'tarih':['01-03-2022','06-30-2022']}
```

In [82]:
```
for i,j in zip(faang_stocks_getileri['semboller'],faang_stocks_getileri['fiyatlar']):

        ret = ((j[1]-j[0])/j[0])*100
        print("{}  hissesinin getirisi  {}" .format(i,round(ret,2)))
```

```
FB  hissesinin getirisi  -52.73
AMZN  hissesinin getirisi  -36.08
AAPL  hissesinin getirisi  -24.68
NFLX  hissesinin getirisi  -70.73
GOOGL  hissesinin getirisi  -24.61
```

for loop İKİ İTERATOR (SIRALAYICI) AYNI UZUNLUKTA DEĞİL İSE

In [83]:
```
faang_stocks_getileri = {'sembol':['FB','AMZN','AAPL','NFLX','GOOGL'],
                         'fiyat':[[338.54,160.03],[170.40,108.92],[181.26,136.53],[597.37,174.87],[145.07,109.37]]
                        'tarih':['01-03-2022','06-30-2022']}
```

```
# hİSSE SENETLERİNİN TARİH BAZINDA FİYATLARINI YAZDIRMAK İSTİYORUZ
for tarih in faang_stocks_getileri['tarih']:
    for i,j in zip(faang_stocks_getileri['sembol'],faang_stocks_getileri['fiyat']):
        if tarih == faang_stocks_getileri['tarih'][0]:
            print("{} fiyatı {} tarihinde {}".format(i,j[0],tarih))
        else:
            print("{} fiyatı {} tarihindet {}".format(i,j[1],tarih))
```

EĞER İTERATİON SIRASINI DEĞİŞTİRİRSEK, AYNI SONUCU ELDE EDERMİYİZ?

In [85]:
```

for i,j in zip(faang_stocks_getileri['sembol'],faang_stocks_getileri['fiyat']):
    for tarih in faang_stocks_getileri['tarih']:
        if tarih == faang_stocks_getileri['tarih'][0]:
            print("{} fiyatı {} tarihinde {}".format(i,j[0],tarih))
        else:
            print("{} fiyatı {} tarihindet {}".format(i,j[1],tarih))
```

```
FB fiyatı 338.54 tarihinde 01-03-2022
FB fiyatı 160.03 tarihindet 06-30-2022
AMZN fiyatı 170.4 tarihinde 01-03-2022
AMZN fiyatı 108.92 tarihindet 06-30-2022
AAPL fiyatı 181.26 tarihinde 01-03-2022
AAPL fiyatı 136.53 tarihindet 06-30-2022
NFLX fiyatı 597.37 tarihinde 01-03-2022
NFLX fiyatı 174.87 tarihindet 06-30-2022
GOOGL fiyatı 145.07 tarihinde 01-03-2022
GOOGL fiyatı 109.37 tarihindet 06-30-2022
```

In [ ]:
```
1
```

enumerate

In [86]:
```
1 boy_ölçüleri = [1.73, 1.68, 1.71, 1.89]
2 for index,i in zip(range(len(boy_ölçüleri)),boy_ölçüleri):
3     print("index " + str(index) + ": " + str(i))
4
```

```
index 0: 1.73
index 1: 1.68
index 2: 1.71
index 3: 1.89
```

YUKARIDAKİ LOOP'U ENUMARATE İLE DAHA KOLAY VE HIZLI YAPABİLİRİZ

In [87]:
```
1 boy_ölçüleri = [1.73, 1.68, 1.71, 1.89]
2 for index, i in enumerate(boy_ölçüleri) :
3     print("index " + str(index) + ": " + str(i))
4
```

```
index 0: 1.73
index 1: 1.68
index 2: 1.71
index 3: 1.89
```

In [ ]:
```
1
```

FOKSİYONLAR(FUNCTİONS): FONSİYONLAR KULLANIMA ÖNCEDEN HAZIRLANMIŞ FORMÜLLERDİR, HIZLI VE AZ HATALI BİR ŞEKİLDE

SCOPE: LOCAL===>GLOBAL===>BULTIN

LOCAL: Fonksiyon içinde tanımlanan değişkendir ve funksiyon dışında bir anlamı yoktur.

GLOBAL :Fonksiyon dışında tanımlanmış(aynı sayfa (kernel) içinde) ama funksiyon içinde de kullanılan değişkenlerdir.

BULTIN : PYTHON'UN kendi içersinde tanımlanmış değişkenlerdir(Pi sayısı gibi).

fonksiyonu tanımlarken def ile başlıyoruz parantez içersinde kullanacağımız değişkenleri yazıyoruz.

In [88]:
```
# matematik işlemleri ni yapmak için math kütüphanesini yüklüyoruz
import math
```

In [89]:
```
# foo() fonksiyonu x,y ve pi sayısını yazdırır x,y,pi
def foo():
    global x_1
    x_1 = 36
    x_2=11
    print(x_1,x_2,math.pi)
```

In [90]:
```
# foo fonksiyonunu çalıştırırsak
foo()
```

```
36 11 3.141592653589793
```

In [91]:
```
# x'i yazdırabilir ve her türlü işlemde kullanabiliriz çünkü global değiken
print(x_1)
```

```
36
```

In [92]:
```
# y' yi yazdıramayız çünkü local değişken
print(x_2)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_8308\545417093.py in <module>
      1 # y' yi yazdıramayız çünkü local değişken
----> 2 print(x_2)

NameError: name 'x_2' is not defined
```

def func_name(var1,var2,...):

```
    operations

    return
```

In [93]:
```
#ders_geçme fonksiyonu
def ders_geçme(final_notu,ev_ödevi_notu,ara_sınav_notu):
    final = final_notu*0.60
    ev_ödevi = ev_ödevi_notu*0.10
    ara_sınav = ara_sınav_notu*0.30
    toplam = final + ev_ödevi + ara_sınav
    if toplam >= 70:

        print('Toplam not {} , Tebrikler dersi geçtin'.format(toplam))

    else:

        print('Toplam not {} , Üzgünüm dersi geçemedin'.format(toplam))

```

In [94]:
```
ders_geçme(final_notu=60,ev_ödevi_notu=60,ara_sınav_notu=60)
```

```
Toplam not 60.0 , Üzgünüm dersi geçemedin
```

In [95]:
```
ders_geçme(60,60,60)
```

```
Toplam not 60.0 , Üzgünüm dersi geçemedin
```

In [96]:
```
ders_geçme(90,60,60)
```

```
Toplam not 78.0 , Tebrikler dersi geçtin
```

In [97]:
```python
# Şifre doğrulama fonksiyonu
def şifre_doğrulama(şifre):
    şifre = str(şifre)
    harf_sayısı =0
    # higher or equal to 8 and lower or equal to 16
    if (len(şifre) < 8) | (len(şifre)>16):

        print("Şifreniz 8 ile 16 haneden oluşmadığı için kabul edilmedi")
    else:
        for i in şifre:
            if i.isalpha()== True:
                harf_sayısı = harf_sayısı +1
        if harf_sayısı!=1:
            print("Şifreniz en az bir harf içermediğinden dolayı kabul edilmedi")
        else:
            print("Şifre kabul edildi")
```

In [98]:
```python
şifre_doğrulama(1234567)
```

```
Şifreniz 8 ile 16 haneden oluşmadığı için kabul edilmedi
```

In [99]:
```python
şifre_doğrulama('12345678')
```

```
Şifreniz en az bir harf içermediğinden dolayı kabul edilmedi
```

In [100]:
```python
şifre_doğrulama('1234567A')
```

```
Şifre kabul edildi
```

EĞER FONSİYON SONUCUNUN GÖZÜKMESİNİ İSTİYORSAK RETURN KELİMESİ İLE GÖSTERMEK İSTEDİĞİMİZ SONUCU BELİRTMELİYİZ

In [101]:
```
# Eğer return ifadesini koymazsak fonksiyon işlemi yapar ama sonuç gözükmez ve işlemde kullanılamaz
def log_ret_A(P0,P1):
    ret = math.log(P1/P0)*100
    ret = round(ret,2)

```

In [102]:
```
# Eğer return ifadesini kullanırsak fonksiyon işlemi yapar ve sonuç gözükür ve işlemlerde kullanabiliriz
def log_ret(P0,P1):
    ret = math.log(P1/P0)*100
    ret = round(ret,2)
    return ret
```

In [103]:
```
# fonksiyon sonucu yazılmayacaktır
log_ret_A(100,110)
print(log_ret_A(100,110))
```

```
None
```

In [104]:
```
# İşlemlerde kullanılamıyacaktır
log_ret_A(100,110)+1
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_8308\2494037855.py in <module>
      1 # İşlemlerde kullanılamıyacaktır
----> 2 log_ret_A(100,110)+1

TypeError: unsupported operand type(s) for +: 'NoneType' and 'int'
```

In [105]:
```
# sonucu yazdırabiliriz
log_ret(100,110)
```

Out[105]:
```
9.53
```

In [106]:
```
# sonucu işlemlerde de kullanabiliriz
log_ret(100,110)+1
```

Out[106]: 10.53

FONSİYONLARI LOOP İÇERSİNDE KULLANABİLİRİZ

In [107]:
```
# portföyümüzdeki hisse isimleri,fiyat bilgileri ve fiyat bilgilerin tarihleri
faang_stocks_getileri = {'sembol':['FB','AMZN','AAPL','NFLX','GOOGL'],
                         'fiyat':[[338.54,160.03],[170.40,108.92],[181.26,136.53],[597.37,174.87],[145.07,109.37]]
                        'tarih':['01-03-2022','06-30-2022']}
```

In [108]:
```
# log_ret() fonksiyonu döngü içinde kullanılabilir
for i,j in zip(faang_stocks_getileri['sembol'],faang_stocks_getileri['fiyat']):

        print("{}  hisse senedi getirisi  {}" .format(i,round(log_ret(j[0],j[1]),2)))
```

```
FB  hisse senedi getirisi  -74.93
AMZN  hisse senedi getirisi  -44.75
AAPL  hisse senedi getirisi  -28.34
NFLX  hisse senedi getirisi  -122.85
GOOGL  hisse senedi getirisi  -28.25
```

In [ ]:
```

```