```
#Inheritance (Miras)
"""
Mevcut (ilk oluşturduğumuz) sınıflara literatürde -> Super Class,
Parent Class, Base Class
Sonradan türettiğimiz sınıflara ise literatürde -> Sub Class, Child
Class, Derived Class
"IS A" ilişkisi vardır. Student is a person.
"""

'\nMevcut (ilk oluşturduğumuz) sınıflara literatürde -> Super Class,
Parent Class, Base Class\nSonradan türettiğimiz sınıflara ise
literatürde -> Sub Class, Child Class, Derived Class\n"IS A" ilişkisi
vardır. Student is a person. \n'

class person: #Base Class, Parent Class
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender

    def PersonInfo(self):
        print('Name :- {}'.format(self.name))
        print('Age :- {}'.format(self.age))
        print('Gender :- {}'.format(self.gender))


class student(person):
    def __init__(self, name, age, gender, studentid, fees):
        person.__init__(self, name, age, gender)
        self.studentid = studentid
        self.fees = fees

    def StudentInfo(self):
        print('Student ID :- {}'.format(self.studentid))
        print('Fees : {}'.format(self.fees))


class teacher(person):
    def __init__(self, name, age, gender, empid, salary):
        person.__init__(self, name, age, gender)
        self.empid = empid
        self.salary = salary

    def TeacherInfo(self):
        print('Employee ID :- {}'.format(self.empid))
        print('Salary : {}'.format(self.salary))


stud1 = student('Batı', 24, 'Male', 123, 500000)
print('Student Details')
```

```
print('---------------')
stud1.PersonInfo()
stud1.StudentInfo()
print()

teacher1 = teacher('Ayşe', 30, 'Female', 'TR1923', 1000000)
print('Employee Details')
print('----------------')
teacher1.PersonInfo()
teacher1.TeacherInfo()
```

```
Student Details
---------------
Name :- Batı
Age :- 24
Gender :- Male
Student ID :- 123
Fees : 500000

Employee Details
----------------
Name :- Ayşe
Age :- 30
Gender :- Female
Employee ID :- TR1923
Salary : 1000000
```

```
class person: #Base Class, Parent Class
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender

    def PersonInfo(self):
        print('Name :- {}'.format(self.name))
        print('Age :- {}'.format(self.age))
        print('Gender :- {}'.format(self.gender))

class student(person):
    def __init__(self, name, age, gender, studentid, fees):
        super().__init__(name, age, gender)
        self.studentid = studentid
        self.fees = fees

    def StudentInfo(self):
        print('Student ID :- {}'.format(self.studentid))
        print('Fees : {}'.format(self.fees))

stud = student('Hasan', 22, 'Male', 987, 450000)
print('Student Details')
```

```
print('---------------')
stud.PersonInfo()
stud.StudentInfo()

Student Details
---------------
Name :- Hasan
Age :- 22
Gender :- Male
Student ID :- 987
Fees : 450000

#Multi-Level Inheritance
class person: #Base Class, Parent Class
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender

    def PersonInfo(self):
        print('Name :- {}'.format(self.name))
        print('Age :- {}'.format(self.age))
        print('Gender :- {}'.format(self.gender))

class employee(person):
    def __init__(self, name, age, gender, empid, salary):
        person.__init__(self, name, age, gender)
        self.empid = empid
        self.salary = salary

    def EmployeeInfo(self):
        print('Employee ID :- {}'.format(self.empid))
        print('Salary : {}'.format(self.salary))

class fulltime(employee):
    def __init__(self, name, age, gender, empid, salary,
WorkExperience):
        employee.__init__(self, name, age, gender, empid, salary)
        self.WorkExperience = WorkExperience

    def FulltimeInfo(self):
        print("Work Experience :- {}".format(self.WorkExperience))


class contractual(employee):
    def __init__(self, name, age, gender, empid, salary,
ContractExpiry):
        employee.__init__(self, name, age, gender, empid, salary)
        self.ContractExpiry = ContractExpiry
```

```
    def ContractInfo(self):
        print("Contract Expiry :- {}".format(self.ContractExpiry))

print('Contractual Employee Details')
print('****************************')
contract1 = contractual('Fatma', 36, 'Female', 555, 1000000, '31-12-
2024')
contract1.PersonInfo()
contract1.EmployeeInfo()
contract1.ContractInfo()

print('\n\n')

print('Fulltime Employee Details')
print('*************************')
fulltim1 = fulltime('Melek', 29, 'Female', 999, 900000, 7)
fulltim1.PersonInfo()
fulltim1.EmployeeInfo()
fulltim1.FulltimeInfo()
```

```
Contractual Employee Details
****************************
Name :- Fatma
Age :- 36
Gender :- Female
Employee ID :- 555
Salary : 1000000
Contract Expiry :- 31-12-2024



Fulltime Employee Details
*************************
Name :- Melek
Age :- 29
Gender :- Female
Employee ID :- 999
Salary : 900000
Work Experience :- 7
```

```
#Multiple Inheritance

#Base Class
class Father:
    def __init__(self):
        self.fathername = str()

#2nd Base Class
class Mother:
    def __init__(self):
```

```
        self.mothername = str()

class Son(Father, Mother):
    name = str()
    def show(self):
        print('My Name :- ', self.name)
        print('Father :- ', self.fathername)
        print('Mother :- ', self.mothername)

s1 = Son() #instance (örnek)
s1.name = 'Bill'
s1.fathername = 'John'
s1.mothername = 'Anna'
s1.show()
```

```
My Name :-  Bill
Father :-  John
Mother :-  Anna
```

```
class Date:
    def __init__(self, date):
        self.date = date

class Time:
    def __init__(self, time):
        self.time = time

class timestamp(Date, Time):
    def __init__(self, date, time):
        Date.__init__(self, date)
        Time.__init__(self, time)
        DateTime = self.date + ' ' + self.time
        print(DateTime)

datetime1 = timestamp('2024-11-01', '11:09:50')
```

```
2024-11-01 11:09:50
```

```
#Method Overriding (Fonksiyonu Ezmek)

class person:
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender

    def greet(self):
        print("Hello Person")

class student(person):
```

```
    def __init__(self, name, age, gender, studentid, fees):
        person.__init__(self, name, age, gender)
        self.studentid = studentid
        self.fees = fees

    def greet(self):
        print("Hello Student not Person!")

stud = student('Gabriel', 20, 'Male', 45, 300000)
stud.greet()

person1 = person('Gabriel', 55, 'Male')
person1.greet()
```

```
Hello Student not Person!
Hello Person
```

```
#GUI Programlama / Tkinter
#PyGTK, PyQt, wxPython

#Pencere Oluşturma
from tkinter import *
pencere = Tk()
mainloop()

#OOP version
from tkinter import *

class Uygulama:
    def __init__(self):
        pass

pencere = Tk()
mainloop()

#Label (Etiket) oluşturma
from tkinter import *

pencere = Tk()

etiket = Label(text = 'Hello Python Lovers!')
etiket.pack()

mainloop()

#OOP versiyonu
from tkinter import *

class Uygulama:
    def __init__(self):
        self.etiket = Label(text = "Dosyayı silmek istediğinize emin
```

```
misiniz?")
        self.etiket.pack()

pencere = Tk()
uyg = Uygulama()
mainloop()

#Pencere başlığı (title)
from tkinter import *

pencere = Tk()

baslik = pencere.title("TBBPython - Day4")

etiket = Label(text = 'Hello Python Lovers!')
etiket.pack()

mainloop()

from tkinter import *

class Uygulama:
    def __init__(self):
        self.etiket = Label(text = "Dosyayı silmek istediğinize emin
misiniz?")
        self.etiket.pack()
        self.baslik = pencere.title("Çok Önemli Uyarı!")

pencere = Tk()
uyg = Uygulama()
mainloop()

#Renkler

#fg -> foreground

from tkinter import *

pencere = Tk()
pencere.title('Hata!')

etiket = Label(text = "Hata : Bilinmeyen hata 404", fg = '#FBCC2F')
etiket.pack()

mainloop()

#bg -> background

from tkinter import *

pencere = Tk()
```

```
pencere.title('Hata!')

etiket = Label(text = "Hata : Bilinmeyen hata 404", bg = 'blue', fg =
'white')
etiket.pack()

mainloop()

#Yazı Tipleri (Fonts)

from tkinter import *
pencere = Tk()
etiket = Label(text = "Hello Python Lovers", font = "Times 15
overstrike")
etiket.pack()
mainloop()

from tkinter import *
import tkinter.font as TkFont

pencere = Tk()
yazitipleri = list(TkFont.families())
yazitipleri.sort()
for i in yazitipleri:
    print(i)

@MS Gothic
@MS PGothic
@MS UI Gothic
@Malgun Gothic
@Malgun Gothic Semilight
@Microsoft JhengHei
@Microsoft JhengHei Light
@Microsoft JhengHei UI
@Microsoft JhengHei UI Light
@Microsoft YaHei
@Microsoft YaHei Light
@Microsoft YaHei UI
@Microsoft YaHei UI Light
@MingLiU-ExtB
@MingLiU_HKSCS-ExtB
@NSimSun
@PMingLiU-ExtB
@SimSun
@SimSun-ExtB
@Yu Gothic
@Yu Gothic Light
@Yu Gothic Medium
@Yu Gothic UI
@Yu Gothic UI Light
```

@Yu Gothic UI Semibold
@Yu Gothic UI Semilight
Agency FB
Algerian
Arabic Transparent
Arial
Arial Baltic
Arial Black
Arial CE
Arial CYR
Arial Greek
Arial Narrow
Arial Rounded MT Bold
Arial TUR
Arial Tur
Bahnschrift
Bahnschrift Condensed
Bahnschrift Light
Bahnschrift Light Condensed
Bahnschrift Light SemiCondensed
Bahnschrift SemiBold
Bahnschrift SemiBold Condensed
Bahnschrift SemiBold SemiConden
Bahnschrift SemiCondensed
Bahnschrift SemiLight
Bahnschrift SemiLight Condensed
Bahnschrift SemiLight SemiConde
Baskerville Old Face
Bauhaus 93
Bell MT
Berlin Sans FB
Berlin Sans FB Demi
Bernard MT Condensed
Blackadder ITC
Bodoni MT
Bodoni MT Black
Bodoni MT Condensed
Bodoni MT Poster Compressed
Book Antiqua
Bookman Old Style
Bookshelf Symbol 7
Bradley Hand ITC
Britannic Bold
Broadway
Brush Script MT
Calibri
Calibri Light
Californian FB
Calisto MT

```
Cambria
Cambria Math
Candara
Candara Light
Cascadia Code
Cascadia Code ExtraLight
Cascadia Code Light
Cascadia Code SemiBold
Cascadia Code SemiLight
Cascadia Mono
Cascadia Mono ExtraLight
Cascadia Mono Light
Cascadia Mono SemiBold
Cascadia Mono SemiLight
Castellar
Centaur
Century
Century Gothic
Century Schoolbook
Chiller
Colonna MT
Comic Sans MS
Consolas
Constantia
Cooper Black
Copperplate Gothic Bold
Copperplate Gothic Light
Corbel
Corbel Light
Courier
Courier
Courier New
Courier New Baltic
Courier New CE
Courier New CYR
Courier New Greek
Courier New TUR
Courier New Tur
Curlz MT
Dubai
Dubai Light
Dubai Medium
Ebrima
Edwardian Script ITC
Elephant
Engravers MT
Eras Bold ITC
Eras Demi ITC
Eras Light ITC
```

Eras Medium ITC
Felix Titling
Fixedsys
Footlight MT Light
Forte
Franklin Gothic Book
Franklin Gothic Demi
Franklin Gothic Demi Cond
Franklin Gothic Heavy
Franklin Gothic Medium
Franklin Gothic Medium Cond
Freestyle Script
French Script MT
Gabriola
Gadugi
Garamond
Georgia
Gigi
Gill Sans MT
Gill Sans MT Condensed
Gill Sans MT Ext Condensed Bold
Gill Sans Ultra Bold
Gill Sans Ultra Bold Condensed
Gloucester MT Extra Condensed
Goudy Old Style
Goudy Stout
Haettenschweiler
Harlow Solid Italic
Harrington
High Tower Text
HoloLens MDL2 Assets
Impact
Imprint MT Shadow
Informal Roman
Ink Free
Javanese Text
Jokerman
Juice ITC
Kristen ITC
Kunstler Script
Leelawadee UI
Leelawadee UI Semilight
Lucida Bright
Lucida Calligraphy
Lucida Console
Lucida Fax
Lucida Handwriting
Lucida Sans
Lucida Sans Typewriter

```
Lucida Sans Unicode
MS Gothic
MS Outlook
MS PGothic
MS Reference Sans Serif
MS Reference Specialty
MS Sans Serif
MS Serif
MS UI Gothic
MT Extra
MV Boli
Magneto
Maiandra GD
Malgun Gothic
Malgun Gothic Semilight
Marlett
Matura MT Script Capitals
Microsoft Himalaya
Microsoft JhengHei
Microsoft JhengHei Light
Microsoft JhengHei UI
Microsoft JhengHei UI Light
Microsoft New Tai Lue
Microsoft PhagsPa
Microsoft Sans Serif
Microsoft Tai Le
Microsoft YaHei
Microsoft YaHei Light
Microsoft YaHei UI
Microsoft YaHei UI Light
Microsoft Yi Baiti
MingLiU-ExtB
MingLiU_HKSCS-ExtB
Mistral
Modern
Modern No. 20
Mongolian Baiti
Monotype Corsiva
Myanmar Text
NSimSun
Niagara Engraved
Niagara Solid
Nirmala UI
Nirmala UI Semilight
OCR A Extended
Old English Text MT
Onyx
PMingLiU-ExtB
Palace Script MT
```

```
Palatino Linotype
Papyrus
Parchment
Perpetua
Perpetua Titling MT
Playbill
Poor Richard
Pristina
Rage Italic
Ravie
Rockwell
Rockwell Condensed
Rockwell Extra Bold
Roman
Script
Script MT Bold
Segoe MDL2 Assets
Segoe Print
Segoe Script
Segoe UI
Segoe UI Black
Segoe UI Emoji
Segoe UI Historic
Segoe UI Light
Segoe UI Semibold
Segoe UI Semilight
Segoe UI Symbol
Showcard Gothic
SimSun
SimSun-ExtB
Sitka Banner
Sitka Display
Sitka Heading
Sitka Small
Sitka Subheading
Sitka Text
Small Fonts
Snap ITC
Stencil
Sylfaen
Symbol
System
Tahoma
Tempus Sans ITC
Terminal
Times New Roman
Times New Roman Baltic
Times New Roman CE
Times New Roman CYR
```

```
Times New Roman Greek
Times New Roman TUR
Times New Roman Tur
Trebuchet MS
Tw Cen MT
Tw Cen MT Condensed
Tw Cen MT Condensed Extra Bold
Verdana
Viner Hand ITC
Vivaldi
Vladimir Script
Webdings
Wide Latin
Wingdings
Wingdings 2
Wingdings 3
Yu Gothic
Yu Gothic Light
Yu Gothic Medium
Yu Gothic UI
Yu Gothic UI Light
Yu Gothic UI Semibold
Yu Gothic UI Semilight

#Pencere Boyutu
from tkinter import *

pencere = Tk()
pencere.geometry("100x100+15+100")

etiket = Label(text = 'Hata!')
etiket.pack()

mainloop()

#resizable - OOP
from tkinter import *

class Uygulama:
    def __init__(self):
        self.etiket = Label(text = 'Hata!')
        self.etiket.pack()

pencere = Tk()
pencere.resizable(width = FALSE, height = FALSE)
uyg = Uygulama()
mainloop()

#Widgets (Pencere Araçları)

#2. Button (Düğme)
```

```
from tkinter import *

pencere = Tk()
dugme = Button(text = "TAMAM", command = pencere.destroy)
dugme.pack()

mainloop()

from tkinter import *

pencere = Tk()

def olustur():
    dosya = open("1101deneme.txt", "w")

dugme = Button(text = "olustur", command = olustur)
dugme.pack()

mainloop()

from tkinter import *

pencere = Tk()

def olustur():
    dosya = open("1101deneme1.txt", 'w')

dugme = Button(text = "olustur", command = olustur)
dugme.pack(side = LEFT)

dugme2 = Button(text = 'cikis', command = pencere.destroy)
dugme2.pack(side = RIGHT)

mainloop()

from tkinter import *
import random

pencere = Tk()
pencere.geometry("300x50+600+460")

def kodlar():
    liste = []
    for i in range(6):
        while len(liste) != 6:
            a = random.randint(1, 100)
            if a not in liste:
                liste.append(a)
    etiket["text"] = liste

etiket = Label(text = "Sayi üretmek icin dugmeye basiniz!", fg =
```

```
'white', bg = '#61380B', font = "Helvetica 12 bold")
etiket.pack()

dugme = Button(text = "Yeniden", command = kodlar)
dugme.pack()

mainloop()

#3. Entry Pencere Aracı

from tkinter import *

pencere = Tk()

giris = Entry()
giris.pack()

mainloop()

from tkinter import *
import random

pencere = Tk()

def bas():
    a = random.randint(1, 100)
    giris.delete(0, END)
    giris.insert(0, a)

giris = Entry(width = 10)
giris.pack()

dugme = Button(text = "bas", command = bas, width = 2, height = 0)
dugme.pack()

mainloop()

#4. Frame Pencere Aracı

from tkinter import *

pencere = Tk()

etiket = Label(text = "Aşağıdaki kutucuğa e-posta adresinizi
yazınız!")
etiket.pack()

giris = Entry()
giris.pack()

cerceve = Frame()
cerceve.pack(pady = 5)
```

```
dugme = Button(text = "Gönder", command = pencere.destroy)
dugme.pack()

mainloop()

#Geometri Yöneticileri: #pack, #grid, #place

from tkinter import *

pencere = Tk()
dolar = Label(text = "Dolar",
              fg = 'white',
              bg = 'red',
              font = 'Verdana 13 bold')
dolar.pack(side = TOP, expand = YES, fill = BOTH)

avro = Label(text = "Avro",
              fg = 'white',
              bg = 'blue',
              font = 'Verdana 13 bold')
avro.pack(side = TOP, expand = YES, fill = BOTH)

lira = Label(text = "Lira",
              fg = 'white',
              bg = 'green',
              font = 'Verdana 13 bold')
lira.pack(side = TOP, expand = YES, fill = BOTH)

mainloop()

#grid

from tkinter import *

dugme1 = Button(text = "Ürünler")
dugme1.grid()

dugme2 = Button(text = "Hizmetler")
dugme2.grid()

dugme3 = Button(text = "Ulaşım")
dugme3.grid()

dugme4 = Button(text = "Hakkında")
dugme4.grid()

mainloop()

from tkinter import *
```

```
dugme1 = Button(text = "Ürünler")
dugme1.grid(row = 0, column = 0)

dugme2 = Button(text = "Hizmetler")
dugme2.grid(row = 0, column = 1)

dugme3 = Button(text = "Ulaşım")
dugme3.grid(row = 1, column = 0)

dugme4 = Button(text = "Hakkında")
dugme4.grid(row = 1, column = 1)

mainloop()

#Hesap Makinesi

from tkinter import *

pencere = Tk()

pencere.resizable(width = FALSE, height = FALSE)

liste = [\
    "9", "8", "7",
    "6", "5", "4",
    "3", "2", "1",
    "0", "+", "-",
    "/", "*", "=",
    "C"]

sira = 1
sutun = 0

for i in liste:
    Button(text = i, width = 4, relief = GROOVE).grid(row = sira,
column = sutun)
    sutun += 1

    if sutun > 2:
        sutun = 0

        sira += 1

mainloop()

#place
from tkinter import *

pencere = Tk()

giris1 = Entry()
```

```
giris1.place(relx = 0.0, rely = 0.0, relheight = 0.15)

dugme1 = Button(text = "Düğme1")
dugme1.place(relx = 0.7, rely = 0.0, relheight = 0.16)

dugme2 = Button(text = "Düğme2")
dugme2.place(relx = 0.0, rely = 0.2, relwidth = 1)

giris2 = Entry()
giris2.place(relx = 0.0, rely = 0.35, relheight = 0.5, relwidth = 1)

mainloop()

#5. Checkbutton

from tkinter import *

pencere = Tk()

onay_el = Checkbutton(text = 'Elma')
onay_el.pack(side = LEFT)

onay_sa = Checkbutton(text = 'Salatalık')
onay_sa.pack(side = LEFT)

onay_do = Checkbutton(text = 'Domates')
onay_do.pack(side = LEFT)

onay_ka = Checkbutton(text = 'Karnıbahar')
onay_ka.pack(side = LEFT)

mainloop()

#6. Toplevel Pencere Aracı

from tkinter import *
pencere = Tk()

def ekle():
    pencere2 = Toplevel()
    btn_pen = Button(pencere2, text = 'çıkış', command =
pencere2.destroy)
    btn_pen.pack()

btn_pen2 = Button(pencere, text = 'ekle', command = ekle)
btn_pen2.pack()

mainloop()

#7. Listbox Pencere Aracı
```

```
from tkinter import *

pencere = Tk()

liste = Listbox(bg = 'white')
liste.pack()

linux_dagitimlari = ["Pardus", "Debian", "Ubuntu", "Backtrack"]

for i in linux_dagitimlari:
    liste.insert(END, i)

etiket = Label(text = "##################", fg = 'magenta', bg =
'light green')
etiket.pack()

btn = Button(text = "ekle", bg = 'orange', fg = 'navy')
btn.pack()

etiket2 = Label(text = "##################", fg = 'magenta', bg =
'light green')
etiket2.pack()

mainloop()

#8. Menü Pencere Aracı

from tkinter import *

pencere = Tk()

menu = Menu(pencere)
pencere.config(menu = menu)
dosya = Menu(menu, tearoff = 0)
menu.add_cascade(label = "Dosya", menu = dosya)
dosya.add_command(label = "Aç")
dosya.add_command(label = "Kaydet")
dosya.add_command(label = "Farklı Kaydet...")
dosya.add_command(label = "Çıkış", command = pencere.destroy)
yeni = Menu(dosya, tearoff = 0)
dosya.add_cascade(label = "Yeni", menu = yeni)
yeni.add_command(label = "Metin Belgesi")
yeni.add_command(label = "Resim Dosyası")
yeni.add_command(label = "pdf dökümanı")

mainloop()

#9. ScrollBar Pencere Aracı

from tkinter import *
```

```
pencere = Tk()

kaydirma = Scrollbar(pencere, orient = HORIZONTAL)
kaydirma.pack(side = BOTTOM, fill = X)

metin = Text(wrap = NONE, xscrollcommand = kaydirma.set)
metin.pack()

kaydirma.config(command = metin.xview)

mainloop()

#Importing multiple .csv files in a Pandas DataFrame

import pandas as pd
import glob

path = "C:\\Users\\ITU\\mydata\\COVID-19"
filenames = glob.glob(path + "/*.csv")
covid = pd.DataFrame()
for f in filenames:
    df = pd.read_csv(f)
    covid = covid.append(df, ignore_index = True, sort = True)

covid.head(10)
```

```
   Active  Admin2  Case_Fatality_Ratio         Combined_Key  Confirmed
\
0     NaN     NaN             4.605705          Afghanistan   160692.0

1     NaN     NaN             1.321435              Albania   251015.0

2     NaN     NaN             2.679334              Algeria   243568.0

3     NaN     NaN             0.413955              Andorra    35028.0

4     NaN     NaN             1.934323               Angola    97812.0

5     NaN     NaN             1.955914  Antigua and Barbuda     6442.0

6     NaN     NaN             1.476244            Argentina  8130023.0

7     NaN     NaN             2.258605              Armenia   355662.0

8     NaN     NaN                  NaN                  NaN        NaN

9     NaN     NaN                  NaN                  NaN        NaN


        Country_Region    Deaths  \
0          Afghanistan    7401.0
1              Albania    3317.0
```

```
2              Algeria    6526.0
3              Andorra     145.0
4               Angola    1892.0
5  Antigua and Barbuda     126.0
6            Argentina  120019.0
7              Armenia    8033.0
8                  NaN       NaN
9                  NaN       NaN

                                                  FIPS  Incident_Rate  \
0                                                  NaN     412.789232
1                                                  NaN    8722.461603
2                                                  NaN     555.444029
3                                                  NaN   45334.886430
4                                                  NaN     297.606044
5                                                  NaN    6578.302426
6                                                  NaN   17988.457196
7                                                  NaN   12002.494572
8  ,,Australian Capital Territory,Australia,2022-...             NaN
9  ,,New South Wales,Australia,2022-01-27 04:21:1...             NaN

           Last_Update       Lat      Long_  Province_State  Recovered

0  2022-01-27 04:21:19  33.93911  67.709953             NaN        NaN

1  2022-01-27 04:21:19  41.15330  20.168300             NaN        NaN

2  2022-01-27 04:21:19  28.03390   1.659600             NaN        NaN

3  2022-01-27 04:21:19  42.50630   1.521800             NaN        NaN

4  2022-01-27 04:21:19 -11.20270  17.873900             NaN        NaN

5  2022-01-27 04:21:19  17.06080 -61.796400             NaN        NaN

6  2022-01-27 04:21:19 -38.41610 -63.616700             NaN        NaN

7  2022-01-27 04:21:19  40.06910  45.038200             NaN        NaN

8                  NaN       NaN        NaN             NaN        NaN

9                  NaN       NaN        NaN             NaN        NaN


covid.tail(10)

       Active  Admin2  Case_Fatality_Ratio          Combined_Key
Confirmed  \
72245     NaN     NaN             0.788640               Uruguay
916388.0
72246     NaN     NaN             0.684866            Uzbekistan
```

```
239025.0
72247     NaN     NaN             0.154816               Vanuatu
9043.0
72248     NaN     NaN             1.092590             Venezuela
523618.0
72249     NaN     NaN             0.401908               Vietnam
10718369.0
72250     NaN     NaN             0.860741    West Bank and Gaza
657573.0
72251     NaN     NaN             0.000000  Winter Olympics 2022
535.0
72252     NaN     NaN            18.177973                 Yemen
11822.0
72253     NaN     NaN             1.239491                Zambia
321503.0
72254     NaN     NaN             2.181743              Zimbabwe
252092.0

             Country_Region   Deaths FIPS  Incident_Rate
Last_Update  \
72245               Uruguay   7227.0  NaN   26380.541706  2022-05-31
04:20:51
72246            Uzbekistan   1637.0  NaN     714.164089  2022-05-31
04:20:51
72247               Vanuatu     14.0  NaN    3089.722564  2022-05-31
04:20:51
72248             Venezuela   5721.0  NaN    1841.394885  2022-05-31
04:20:51
72249               Vietnam  43078.0  NaN   11011.429045  2022-05-31
04:20:51
72250    West Bank and Gaza   5660.0  NaN   12890.009362  2022-05-31
04:20:51
72251  Winter Olympics 2022      0.0  NaN            NaN  2022-05-31
04:20:51
72252                 Yemen   2149.0  NaN      39.636601  2022-05-31
04:20:51
72253                Zambia   3985.0  NaN    1748.823811  2022-05-31
04:20:51
72254              Zimbabwe   5500.0  NaN    1696.112751  2022-05-31
04:20:51

             Lat       Long_  Province_State  Recovered
72245 -32.522800  -55.765800             NaN        NaN
72246  41.377491   64.585262             NaN        NaN
72247 -15.376700  166.959200             NaN        NaN
72248   6.423800  -66.589700             NaN        NaN
72249  14.058324  108.277199             NaN        NaN
72250  31.952200   35.233200             NaN        NaN
72251  39.904200  116.407400             NaN        NaN
```

```
72252  15.552727   48.516388             NaN        NaN
72253 -13.133897   27.849332             NaN        NaN
72254 -19.015438   29.154857             NaN        NaN
```

```
covid['Country_Region'].unique()
```

```
array(['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola',
       'Antigua and Barbuda', 'Argentina', 'Armenia', nan, 'Austria',
       'Azerbaijan', 'Bahamas', 'Bahrain', 'Bangladesh', 'Barbados',
       'Belarus', 'Belize', 'Benin', 'Bhutan', 'Bolivia',
       'Bosnia and Herzegovina', 'Botswana', 'Brunei', 'Bulgaria',
       'Burkina Faso', 'Burma', 'Burundi', 'Cabo Verde', 'Cambodia',
       'Cameroon', 'Central African Republic', 'Chad', 'Comoros',
       'Congo (Brazzaville)', 'Congo (Kinshasa)', 'Costa Rica',
       "Cote d'Ivoire", 'Croatia', 'Cuba', 'Cyprus', 'Czechia',
'Denmark',
       'Diamond Princess', 'Djibouti', 'Dominica', 'Dominican
Republic',
       'Ecuador', 'Egypt', 'El Salvador', 'Equatorial Guinea',
'Eritrea',
       'Estonia', 'Eswatini', 'Ethiopia', 'Fiji', 'Finland', 'France',
       'Gabon', 'Gambia', 'Georgia', 'Ghana', 'Greece', 'Grenada',
       'Guatemala', 'Guinea', 'Guinea-Bissau', 'Guyana', 'Haiti',
       'Holy See', 'Honduras', 'Hungary', 'Iceland', 'Indonesia',
'Iran',
       'Iraq', 'Ireland', 'Israel', 'Jamaica', 'Jordan', 'Kazakhstan',
       'Kenya', 'Kiribati', 'Kosovo', 'Kuwait', 'Kyrgyzstan', 'Laos',
       'Latvia', 'Lebanon', 'Lesotho', 'Liberia', 'Libya',
       'Liechtenstein', 'Lithuania', 'Luxembourg', 'MS Zaandam',
       'Madagascar', 'Malawi', 'Maldives', 'Mali', 'Malta',
       'Marshall Islands', 'Mauritania', 'Mauritius', 'Micronesia',
       'Moldova', 'Monaco', 'Mongolia', 'Montenegro', 'Morocco',
       'Mozambique', 'Namibia', 'Nepal', 'New Zealand', 'Nicaragua',
       'Niger', 'Nigeria', 'North Macedonia', 'Norway', 'Oman',
'Palau',
       'Panama', 'Papua New Guinea', 'Paraguay', 'Philippines',
'Poland',
       'Portugal', 'Qatar', 'Romania', 'Rwanda', 'Saint Kitts and
Nevis',
       'Saint Lucia', 'Saint Vincent and the Grenadines', 'Samoa',
       'San Marino', 'Sao Tome and Principe', 'Saudi Arabia',
'Senegal',
       'Serbia', 'Seychelles', 'Sierra Leone', 'Singapore',
'Slovakia',
       'Slovenia', 'Solomon Islands', 'Somalia', 'South Africa',
       'South Sudan', 'Sri Lanka', 'Sudan', 'Summer Olympics 2020',
       'Suriname', 'Switzerland', 'Syria', 'Taiwan*', 'Tajikistan',
       'Tanzania', 'Thailand', 'Timor-Leste', 'Togo', 'Tonga',
       'Trinidad and Tobago', 'Tunisia', 'Turkey', 'Uganda',
       'United Arab Emirates', 'Uruguay', 'Uzbekistan', 'Vanuatu',
```

```
       'Venezuela', 'Vietnam', 'West Bank and Gaza', 'Yemen',
'Zambia',
       'Zimbabwe', 'Antarctica', 'Nauru', 'Tuvalu',
       'Winter Olympics 2022'], dtype=object)
```

```
covid['Country_Region'].nunique()
```

```
177
```

```
df1 = covid[['Country_Region', 'Confirmed', 'Deaths', 'Last_Update']]
df1.head()
```

```
  Country_Region  Confirmed  Deaths          Last_Update
0    Afghanistan   160692.0  7401.0  2022-01-27 04:21:19
1        Albania   251015.0  3317.0  2022-01-27 04:21:19
2        Algeria   243568.0  6526.0  2022-01-27 04:21:19
3        Andorra    35028.0   145.0  2022-01-27 04:21:19
4         Angola    97812.0  1892.0  2022-01-27 04:21:19
```