## Python Classes and Objects

```
#Create a class named My_Coolor, with a property named White:

class My_Color:
  White = "Red + Blue + Green_Light"
print(My_Color)
```

```
#Pint Class Properties:
class My_Color:
  White = "Red + Blue + Green_Light"
C1 = My_Color()
print(C1.White)
```

```
#__init__? // A reseved method in python classes.
#C++ constructor in an object-oriented approach.
#Initialize the object's attributes
class F_Prime_Minister:
    def __init__(self, name, age):
        self.name = name
        self.age = age

p1 = F_Prime_Minister("Saiful Islam Ratan", "22+/-")

print("Name of the Future Prime Minister:", p1.name)
print("And his age:", p1.age)

```

```
#Self?
#'self' is always passed in its argument.//allowing you to access its attributes and call its
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def myfunc(self):
    print("Hello my name is " + self.name)

p1 = Person("Iqra", 25)
p1.myfunc()

```

## Python Inheritance

```
#Inheritance: Parent class <----Child class (Inherits all the methods and properties from anoth
class Person:
  def __init__(self, fname, lname):
    self.firstname = fname
    self.lastname = lname

  def printname(self):
    print(self.firstname, self.lastname)

```

```
class Student(Person):
  def __init__(self, fname, lname, year):
    super().__init__(fname, lname)
    #super() function->child class inherit all the methods and properties from its parent:

    self.graduationyear = year

  def welcome(self):
    print("Welcome", self.firstname, self.lastname, "to the AI lab class of", self.graduationye

x = Student("Iqra", "Islam", "Summer 2023 Trimester")
x.welcome()
```

## NumPy -->Numerical Python --> Arrays

```
#Create a NumPy array:
import numpy as np
#print(np.__version__)
Array = np.array([1, 2, 3, 4, 5,6])

print(Array)

print(type(Array))

```

```
[1 2 3 4 5 6]
<class 'numpy.ndarray'>
```

```
#0-D Arrays:
import numpy as np

Array = np.array(99)

print(Array)
```

```
99
```

```
#1-D Arrays:
import numpy as np

Array = np.array([9,8,7,6,5,4,3,2,1,0])

print(Array)
```

```
[9 8 7 6 5 4 3 2 1 0]
```

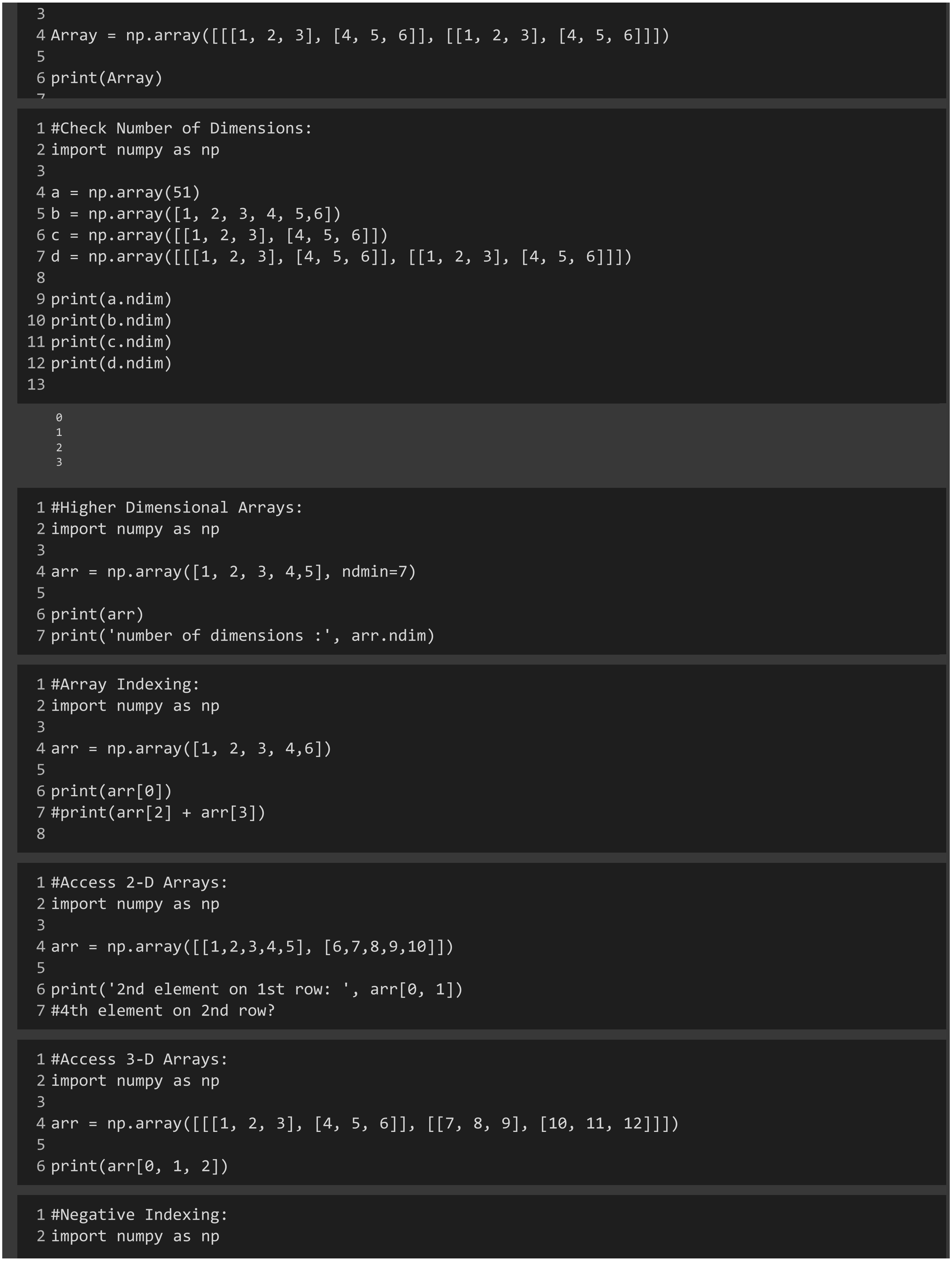
```
#2-D arrays:
import numpy as np

Array = np.array([[1, 2, 3,4], [0,7, 5, 6]])

print(Array)
```

```
[[1 2 3 4]
 [0 7 5 6]]
```

```
#3-D arrays
import numpy as np
```



```

Array = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(Array)
```

```
#Check Number of Dimensions:
import numpy as np

a = np.array(51)
b = np.array([1, 2, 3, 4, 5,6])
c = np.array([[1, 2, 3], [4, 5, 6]])
d = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(a.ndim)
print(b.ndim)
print(c.ndim)
print(d.ndim)

```

```
0
1
2
3
```

```
#Higher Dimensional Arrays:
import numpy as np

arr = np.array([1, 2, 3, 4,5], ndmin=7)

print(arr)
print('number of dimensions :', arr.ndim)
```

```
#Array Indexing:
import numpy as np

arr = np.array([1, 2, 3, 4,6])

print(arr[0])
#print(arr[2] + arr[3])

```

```
#Access 2-D Arrays:
import numpy as np

arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])

print('2nd element on 1st row: ', arr[0, 1])
#4th element on 2nd row?
```

```
#Access 3-D Arrays:
import numpy as np

arr = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])

print(arr[0, 1, 2])
```

```
#Negative Indexing:
import numpy as np
```

```

arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])

print('Last element from 2nd dim: ', arr[1, -1])
```

```
#Array Slicing:
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[1:5])
```

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[4:])

```

```
[5 6 7]
```

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[:7])
#print(arr[:])
```

```
#Negative Slicing:
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[-3:-1])

```

```
#STEP
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[1:5:2])
```

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[::2])
```

```
#Slicing 2-D Arrays:
import numpy as np

arr = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])

print(arr[1, 1:4])
```

```
import numpy as np

arr = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])

print(arr[0:2, 2])
```

```
import numpy as np

arr = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])

print(arr[0:2, 1:4])
```

```
#Data Types:
import numpy as np

arr = np.array([1, 0, 3])

newarr = arr.astype(bool)

print(newarr)
print(newarr.dtype)

```

```
#Array Reshaping:
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7, 8])

newarr = arr.reshape(2, 2, -1)

print(newarr)
```

## Pandas: ->analyzing, cleaning, exploring, and manipulating data.

```
import pandas as pd
#print(pd.__version__)
mydataset = {
  'cars': ["BMW", "Volvo", "Ford"],
  'passings': [3, 7, 2]
}

myvar = pd.DataFrame(mydataset)

print(myvar)
```

```
#Data Frame:
import pandas as pd

data = {
  "Duration":{
    "0":60,
    "1":60,
    "2":60,
    "3":45,
    "4":45,
    "5":60
  },
  "Pulse":{
    "0":110,
    "1":117,
    "2":103,
    "3":109,
    "4":117,
    "5":102
  },
  "Maxpulse":{
    "0":130,
    "1":145,
    "2":135,
    "3":175,
    "4":148,
    "5":127
  },
  "Calories":{
    "0":409.1,
    "1":479.0,
    "2":340.0,
    "3":282.4,
    "4":406.0,
    "5":300.5
  }
}

df = pd.DataFrame(data)

print(df)

```

## Pandas Read CSV

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df)
```

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.to_string())#to_string() to print the entire DataFrame.
```

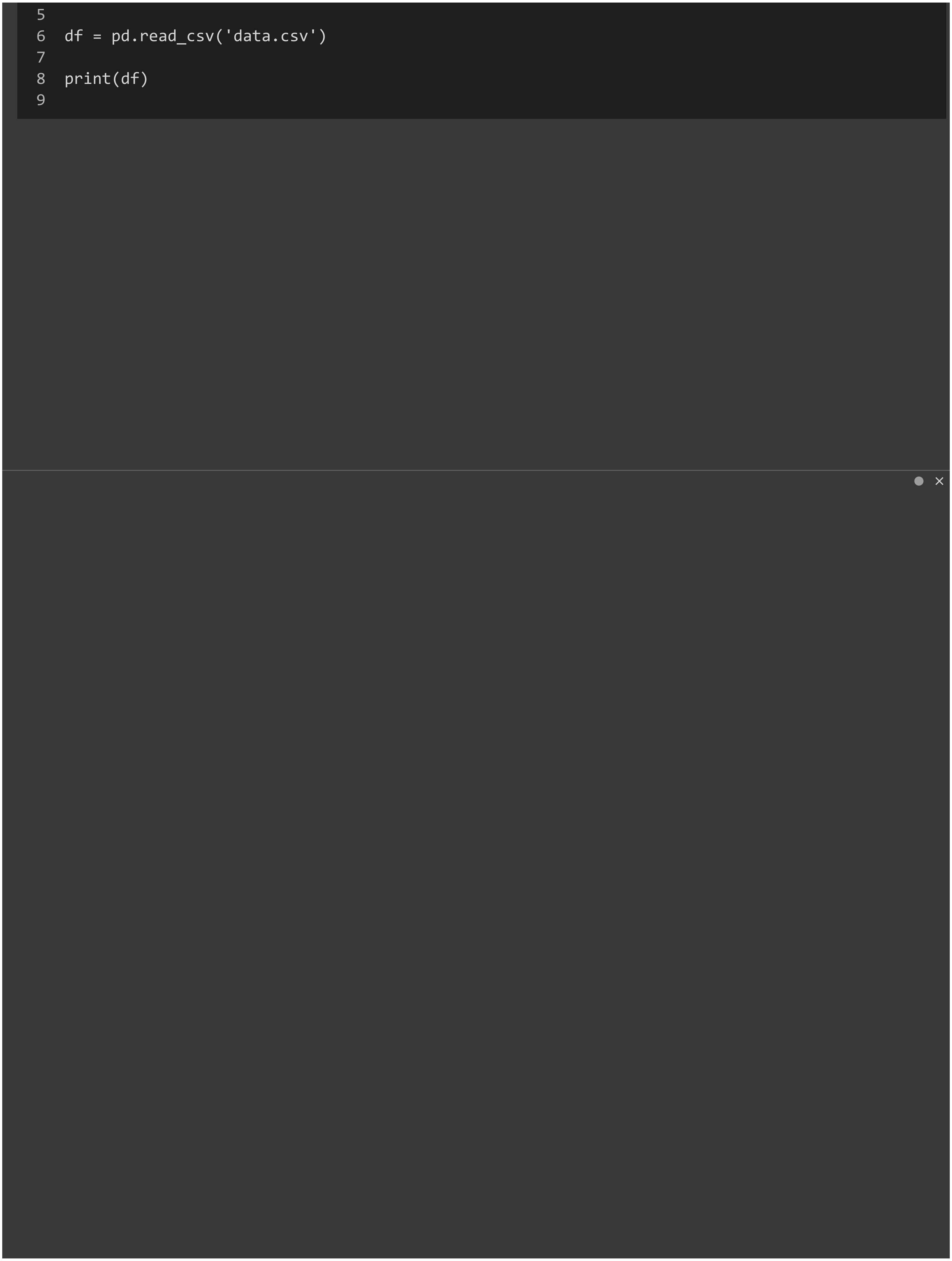
```
#Max_Rows:
import pandas as pd

print(pd.options.display.max_rows)
```

```
60
```

```
#Increase the maximum number of rows:
import pandas as pd

pd.options.display.max_rows = 9999
```



```python

df = pd.read_csv('data.csv')

print(df)

```