



**JMIETI, RADAUR**

JAI PARKASH MUKAND LAL INNOVATIVE ENGG. & TECH. INSTITUTE

# Practical File Of AI&ML workshop-I

**Submitted By:**

**Kanan Sharma**

**8522222**

**Submitted to:**

**Mr. Sumit Kumar Mahana**

**AP (CSE Deptt.)**

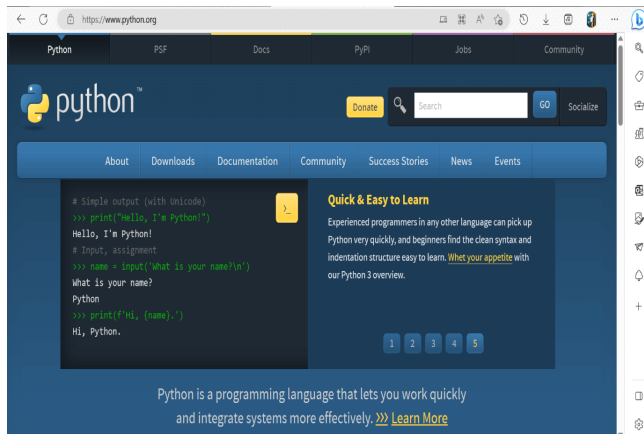
**JMIETI, Radaur**

Sr.No.	List of Practical	Date	Sign
1.	Installation of Python, and learning interactively at command prompt and write simple programs.		
2.	Program to calculate a five-function calculator.		
3.	Program to count uppercase and lowercase letters in an inputted string.		
4.	Program to find maximum, minimum and mean value from the list.		
•	Learning the conditions and iterations in python by writing and running simple programs.		
5.	Write a program to check student's grade. Your program must fulfil the following conditions: <ol style="list-style-type: none"> <li>1. Grade A –Outstanding.</li> <li>2. Grade B –Excellent.</li> <li>3. Grade C –Very Good</li> <li>4. Grade D –Good</li> <li>5. Grade E –Satisfactory.</li> <li>6. Grade F – Unrecognized.</li> </ol>		
6.	Program to find sum of N natural numbers (for loop implementation).		
7.	Write a program to calculate the factorial of an inputted number (using while loop).		
•	Random numbers generations, and problems based on random numbers.		
8.	WAP to print random numbers.		
9.	WAP to print random items from given list.		
10.	WAP to print random number with in-built library.		
•	Handling tuples and exercises based on tuples.		
11.	Write a program to print Fibonacci series by using tuples.		
•	Text Processing using python.		
12.	WAP to count the number of lines from a text file “story.txt” which is not starting from “T”		
13.	WAP using NumPy to capitalize the first letter, lowercase, uppercase, swap-case, title-case of given array => ['python' 'PHP' 'java' 'c++']		
14.	WAP to create data frames using Pandas Package and implement its features, how to read csvfile		

# #1

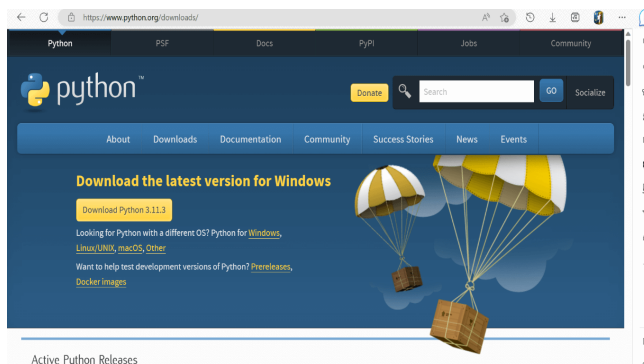
## Installation of Python, and learning interactively at command prompt and write simple programs.

Visit the link <https://www.python.org> to download the latest release of Python. In this process, we will install Python 3.11.3 on our Windows operating system. When we click on the above link, it will bring us the following page.



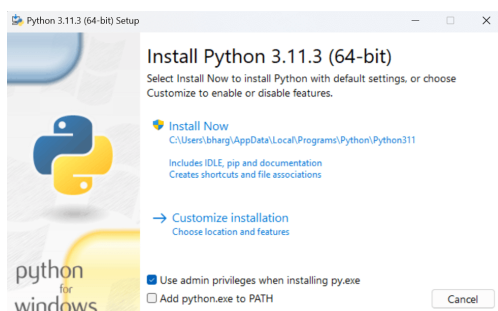
### Step - 1: Select the Python's version to download.

Click on the download button to download the exe file of Python.



### Step - 2: Click on the Install Now

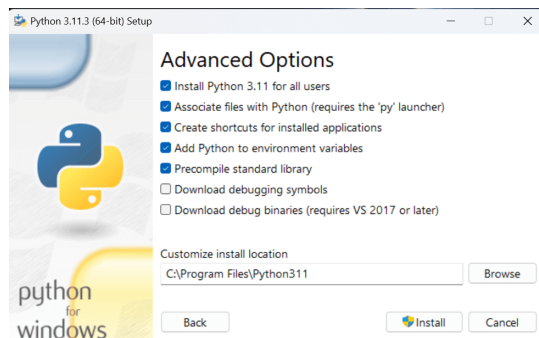
Double-click the executable file, which is downloaded.



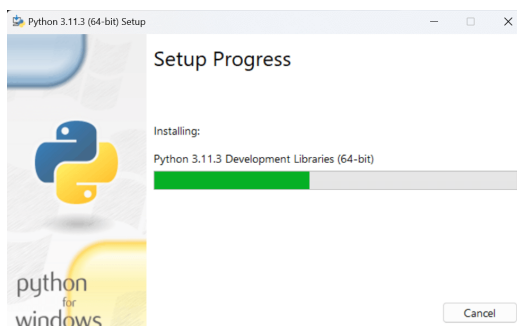
The following window will open. Click on the Add Path check box, it will set the Python path automatically.

Now, Select Customize installation and proceed. We can also click on the customize installation to choose desired location and features. Other important thing is install launcher for the all user must be checked.

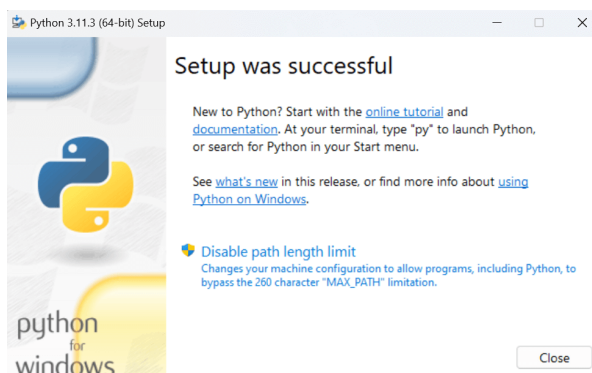
Here, under the advanced options, click on the checkboxes of " Install Python 3.11 for all users ", which is previously not checked in. This will check the other option " Precompile standard library " automatically. And the location of the installation will also be changed. We can change it later, so we leave the install location default. Then, click on the install button to finally install.



### Step - 3 Installation in Process



The set up is in progress. All the python libraries, packages, and other python default files will be installed in our system. Once the installation is successful, the following page will appear saying " Setup was successful ".

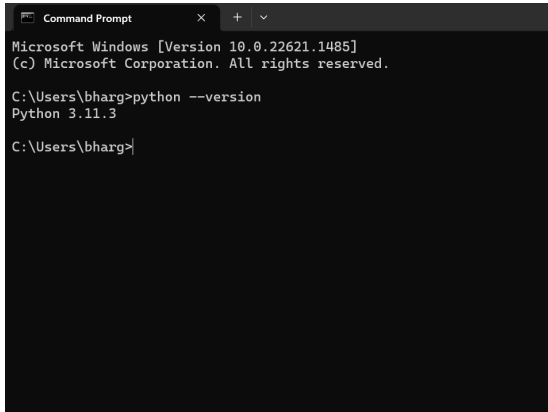


### Step - 4: Verifying the Python Installation

To verify whether the python is installed or not in our system, we have to do the following.

- Go to "Start" button, and search " cmd ".
- Then type, " **python - - version** ".

- If python is successfully installed, then we can see the version of the python installed.
- If not installed, then it will print the error as " **'python' is not recognized as an internal or external command, operable program or batch file.** ".



```
Microsoft Windows [Version 10.0.22621.1485]
(c) Microsoft Corporation. All rights reserved.

C:\Users\bharg>python --version
Python 3.11.3

C:\Users\bharg>
```

**We are ready to work with the Python.**

## The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

```
C:\Users\Your Name>python
Or, if the "python" command did not work, you can try "py":
C:\Users\Your Name>py
```

From there you can write any python, including our hello world example from earlier in the tutorial:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
```

Which will write "Hello, World!" in the command line:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
Hello, World!
```

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

```
exit()
```

## #2

### Program to calculate a five-function calculator.

```
def math(a,b,method):
    if method == "add":
        return a+b
    elif method == "sub":
        return a-b
    elif method == "mul":
        return a*b
    elif method == "div":
        return a/b

x = str(input("choose add,sub,multiply,divide= "))
m = int(input("enter the first number "))
n = int(input("enter the second number "))
b = math

if x == "add":
    print("after addition",b(m,n,"add"))
elif x == "sub":
    print("after subtraction",b(m,n,"sub"))
if x == "multiply":
    print("after multiplication",b(m,n,"mul"))
elif x=="divide":
    print("after division",b(m,n,"div"))
```

### output

```
● PS E:\project\python> python .\PJT.py
choose add,sub,multiply,divide= add
enter the first number 55
enter the second number 45
after addition 100
○ PS E:\project\python> █
```

### #3

**Program to count uppercase and lowercase letters in an inputted string.**

```
str = str(input("enter the String "))
print("Original strings : ",str)
upr, lwr, num, spl = 0, 0, 0, 0
for i in range(len(str)):
    if str[i] >= 'A' and str[i] <= 'Z':
        upr += 1
    elif str[i] >= 'a' and str[i] <= 'z':
        lwr += 1
    elif str[i] >= '0' and str[i] <= '9':
        num += 1
    else:
        spl += 1
print("UpperCase : ",upr)
print("LowerCase : ",lwr)
print("NumberCase : ",num)
print("SpecialCase : ",spl)
```

### Output

```
● PS E:\project\python> python .\test.py
○ enter the String Hello World!
Original strings : Hello World!
UpperCase : 2
LowerCase : 8
NumberCase : 0
SpecialCase : 2
PS E:\project\python> █
```

#### #4

**Program to find maximum, minimum and mean value from the list.**

```
n = []
x = int(input("Enter number of elements : "))
for i in range(0, x):
    a = int(input())
    n.append(a)
print(n)
max_value = max(n)
min_value = min(n)
mean_value = sum(n) / len(n)
print("Maximum value:", max_value)
print("Minimum value:", min_value)
print("Mean value:", mean_value)
```

output

```
Enter number of elements : 5
10
11
12
13
14
[10, 11, 12, 13, 14]
Maximum value: 14
Minimum value: 10
Mean value: 12.0
PS E:\project\python>
```



## #5

**Write a program to check student's grade.**

```
x=int(input('enter marks 1 '))
y=int(input('enter marks 2 '))
z=int(input('enter marks 3 '))
a = (x+y+z)/3
print (a, "is average")
if a>= 90:
    print("Grade = A Outstanding")
elif 80 <=a< 90:
    print("Grade = B Excellent")
elif 70 <=a< 80:
    print("Grade = C Very Good")
elif 50<=a< 70:
    print("Grade = D Good")
elif 33<=a< 50:
    print("Grade = E Satisfactory")
elif a<33:
    print("Grade = F Unrecognized")
```

## output

```
● PS E:\project\python> python .\project.py
enter marks 55
enter marks 55
enter marks 40
50.0 is average
Grade = D Good
```

## #6

**Program to find sum of N natural numbers (for loop implementation).**

```
N = int(input("Enter a positive integer N: "))
n = 0
for i in range(1, N + 1):
    n += i
print(n)
```

**output**

```
● PS E:\project\python> python .\project.py
Enter a positive integer N: 5
15
```

**#7**

**Write a program to calculate the factorial of an inputted number (using while loop).**

```
n = int(input("Enter a number: "))
f = 1
if n<0:
    print("Factorial is not defined for negative numbers.")
elif n==0:
    print("The factorial of 0 is 1")
else:
    while n>0:
        f *= n
        n-=1
print(f)
```

**Output**

```
● PS E:\project\python> python .\project.py
Enter a number: 5
120
○ PS E:\project\python> █
```

**#8**

**WAP to print random numbers.**

```
import random
b = random.randint(1, 100)
print(b)
```

**Output**

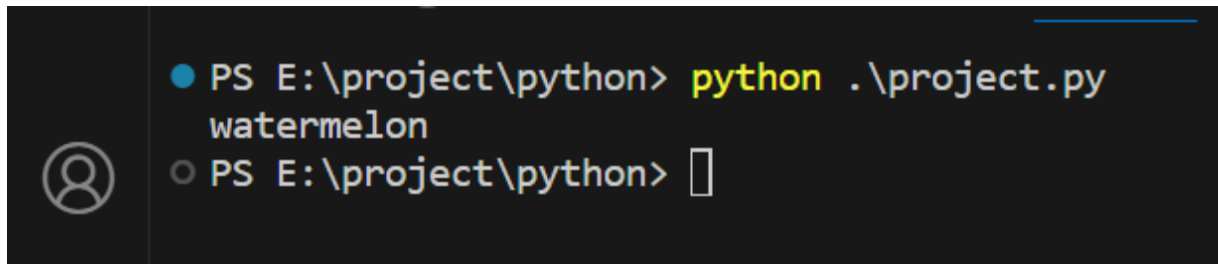
```
● PS E:\project\python> python .\project.py
37
```

**#9**

**WAP to print random items from given list.**

```
import random
my_list=["apple","banana","cherry","watermelon","grape
s","guawa","muskmelon"]
b=random.choice(my_list)
print(b)
```

**Output**

A screenshot of a Windows PowerShell terminal window. The prompt is 'PS E:\project\python>'. The user has entered 'python .\project.py' and the output is 'watermelon'. The prompt is now 'PS E:\project\python>'.

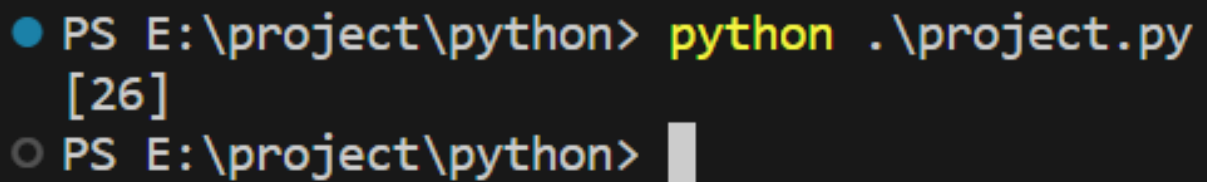
```
● PS E:\project\python> python .\project.py
watermelon
○ PS E:\project\python> 
```

## #10

**WAP to print random number with in-built library.**

```
import random
b = random.randint(10, 100)
list=[]
list.append(b)
print(list)
```

## Output



The screenshot shows a Windows command prompt window with a dark background. At the top, there are several tabs labeled 'PROBLEMS', 'PYTHON', 'DEBUG CONSOLE', and 'TERMINAL'. The active tab is 'PYTHON'. The command prompt shows the following sequence of events: a blue cursor icon followed by the command 'PS E:\project\python> python .\project.py', which is followed by the output '[26]'. Below this, the prompt returns to 'PS E:\project\python>' with a white cursor icon.

```
● PS E:\project\python> python .\project.py
  [26]
○ PS E:\project\python> █
```

## #11

Write a program to print Fibonacci series by using tuples.

```
def fibonacci(n):  
    fib = ()  
    a, b = 0, 1  
    for x in range(n):  
        fib += (a,)   
        a, b = b, a + b  
    return fib  
num = int(input("Enter the number of terms in the  
Fibonacci series: "))  
fib=fibonacci(num)  
print("Fibonacci Series:")  
print(fib)
```

## Output

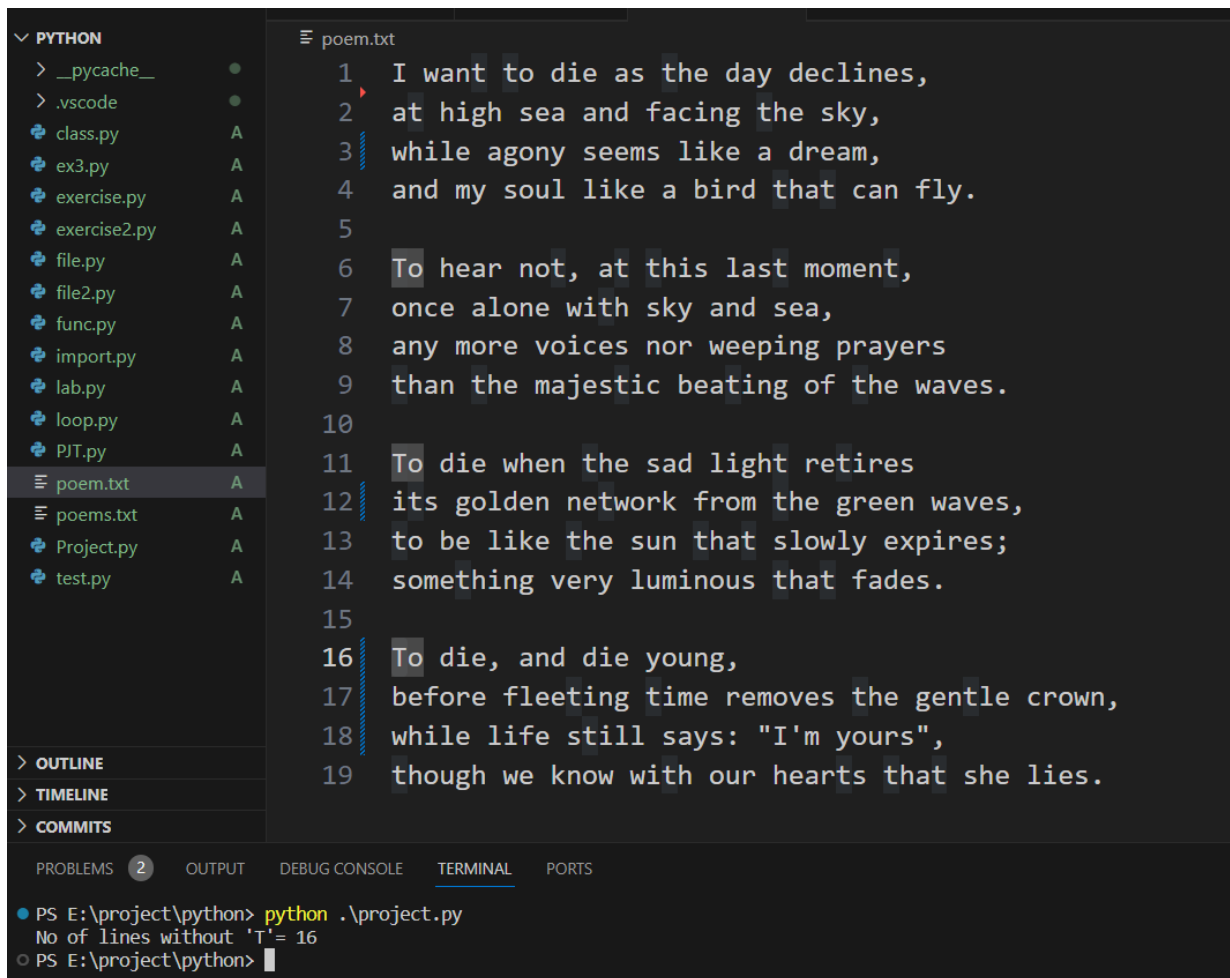
```
● PS E:\project\python> python .\project.py  
Enter the number of terms in the Fibonacci series: 10  
Fibonacci Series:  
(0, 1, 1, 2, 3, 5, 8, 13, 21, 34)  
○ PS E:\project\python> █
```

## #12

**WAP to count the number of lines from a text file "story.txt" which is not starting from "T"**

```
def count():  
    f = open("poem.txt","r")  
    lines = f.readlines()  
    c = sum(1 for line in lines if not line.strip().startswith('T'))  
    f.close()  
    print("No of lines without 'T'=",c)  
count()
```

## output



The screenshot shows a code editor with a file explorer on the left, a code editor in the center, and a terminal at the bottom. The file explorer shows a list of files including `__pycache__`, `.vscode`, `class.py`, `ex3.py`, `exercise.py`, `exercise2.py`, `file.py`, `file2.py`, `func.py`, `import.py`, `lab.py`, `loop.py`, `PJT.py`, `poem.txt`, `poems.txt`, `Project.py`, and `test.py`. The code editor shows the content of `poem.txt`, which contains 19 lines of text. The terminal shows the output of running the Python script, which is "No of lines without 'T'= 16".

```
1 I want to die as the day declines,  
2 at high sea and facing the sky,  
3 while agony seems like a dream,  
4 and my soul like a bird that can fly.  
5  
6 To hear not, at this last moment,  
7 once alone with sky and sea,  
8 any more voices nor weeping prayers  
9 than the majestic beating of the waves.  
10  
11 To die when the sad light retires  
12 its golden network from the green waves,  
13 to be like the sun that slowly expires;  
14 something very luminous that fades.  
15  
16 To die, and die young,  
17 before fleeting time removes the gentle crown,  
18 while life still says: "I'm yours",  
19 though we know with our hearts that she lies.
```

```
PS E:\project\python> python .\project.py  
No of lines without 'T'= 16  
PS E:\project\python>
```



### #13

**WAP using numpy to capitalize the first letter, lowercase, uppercase, swap-case, title-case of given array => ['python' 'PHP' 'java' 'c++']**

```
import numpy as n
x = n.array(['python', 'PHP', 'java', 'c++'])
print("Original Array:",x)
a = n.char.capitalize(x)
print("Capitalize: ",a)
b = n.char.lower(x)
print("Lower: ", b)
c = n.char.upper(x)
print("Upper: ",c)
d = n.char.swapcase(x)
print("Swapcase: ",d)
e = n.char.title(x)
print("Title-case: ",e)
```

### Output

```
● PS E:\project\python> python .\project.py
Original Array: ['python' 'PHP' 'java' 'c++']
Capitalize:  ['Python' 'Php' 'Java' 'C++']
Lower:  ['python' 'php' 'java' 'c++']
Upper:  ['PYTHON' 'PHP' 'JAVA' 'C++']
Swapcase:  ['PYTHON' 'php' 'JAVA' 'C++']
Title-case:  ['Python' 'Php' 'Java' 'C++']
```

## #14

### WAP to create data frames using Pandas Package and implement its features, how to read csvfiles

```
import pandas as pd
data = {
    'Name': ['John', 'Alice', 'Bob', 'Eva'],
    'Age': [25, 30, 22, 28],
    'City': ['New York', 'San Francisco', 'Los Angeles', 'Chicago']}
df = pd.DataFrame(data)
print("DataFrame from dictionary:")
print(df)
print("\n")
print("Accessing 'Name' column:")
print(df['Name'])
print("\n")
df['Occupation'] = ['Engineer', 'Doctor', 'Artist', 'Teacher']
print("DataFrame with new 'Occupation' column:")
print(df)
print("\n")
print("Filtering rows where Age is greater than 25:")
filtered_df = df[df['Age'] > 25]
print(filtered_df)
print("\n")
```

### output

```
PS E:\project\python> python .\project.py
DataFrame from dictionary:
   Name  Age      City
0  John   25  New York
1 Alice   30 San Francisco
2  Bob   22  Los Angeles
3  Eva   28   Chicago
```

```
Accessing 'Name' column:
```

```
0    John
1    Alice
2     Bob
3     Eva
```

```
Name: Name, dtype: object
```

```
DataFrame with new 'Occupation' column:
```

	Name	Age	City	Occupation
0	John	25	New York	Engineer
1	Alice	30	San Francisco	Doctor
2	Bob	22	Los Angeles	Artist
3	Eva	28	Chicago	Teacher

```
Filtering rows where Age is greater than 25:
```

	Name	Age	City	Occupation
1	Alice	30	San Francisco	Doctor
3	Eva	28	Chicago	Teacher

## How to read csvfiles

Reading a CSV file

Let's assume you have a CSV file named 'sample.csv' with the same columns (Name, Age, City, Occupation)

```
csv_file_path = 'sample.csv'
```

Reading CSV file into a DataFrame

```
csv_df = pd.read_csv(csv_file_path)
print("DataFrame from CSV file:")
print(csv_df)
```