Test V25.02.07

- Array and File Handling

- Q1. Create an Array and Display it?

In [44]:
```
import numpy as np
arr = np.array([0,1,2,3,4,5,6,7,8,9])
print(arr)
print(arr.ndim)
```

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

- Q2. Create a file "Coding.txt"

In [45]:
```
obj = open("Coading.txt","w")
obj.close()
```

- Q3. Write 5 lines to "Coding.txt"

In [49]:
```
obj = open("Coading.txt","w")
obj.write("Pratik")
obj.write("\nArchana")
obj.write("\nPragati")
obj.write("\nRamesh")
obj.write("\nMajage")
obj.close()
```

- Q4. Append 5 lines to "Coding.txt" and read it?

In [50]:
```
obj = open("Coading.txt","r")
data = obj.read()
print(data)
obj.close()
```

```
Pratik
Archana
Pragati
Ramesh
Majage
```

- Q5. Open the file "Coding.txt" and write two lines in the same code?

In [51]:
```
obj = open("Coading.txt","a")
obj.write("\nNupur")
obj = open("Coading.txt","r")
data = obj.read()
print(data)
```

```
Pratik
Archana
Pragati
Ramesh
Majage
Nupur
```

- Q6. Create an array of 2 elements and display them in reverse order.

In [55]:
```
import numpy as np
a = np.array([1,3])
b = np.sort(-a)
print(b)
#
```

```
[-3 -1]
```

- Q7. Display the array in assending order.

In [56]:
```
import numpy as np
a = np.array([5,1,4,3,2])
```

```
b = np.sort(a)
print(b)
```

```
[1 2 3 4 5]
```

- Q8. Perform positive and negative slicing and display entire array.

In [57]:
```
import numpy as np
a = np.array([1,2,3,4,5])
print(a[0:6])
print(a[-5:])
```

```
[1 2 3 4 5]
[1 2 3 4 5]
```

- Q9. Create a 3-Dimentional array with 3 arrays, each with 2-dimentional array and display the last element of last array.

In [58]:
```
import numpy as np
a = np.array([[[1,2],[3,4]],[[5,6],[7,8]],[[9,10],[11,12]]])
print(a)
print(a.ndim)
print(a[-1,-1,-1])
```

```
[[[ 1  2]
  [ 3  4]]

 [[ 5  6]
  [ 7  8]]

 [[ 9 10]
  [11 12]]]
3
12
```

- Regular Expression - Meta Chrecters

- Q10. Execute all Regular Expression Charecters.

- Search | Findall | Sub | Split

- start : ^

In [105…
```
import re
data = "Pandu is good boy"
a = re.search("^Pandu",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
<re.Match object; span=(0, 5), match='Pandu'>
Found.
```

- end : $

In [109…
```
import re
data = "Pandu is good boy"
a = re.search("boy$",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
<re.Match object; span=(14, 17), match='boy'>
Found.
```

- Remaining charecters : .*

In [108…
```
import re
data = "Pandu is good boy"
a = re.search("^Pandu.*boy$",data)
print(a)
if a:
```

```
    print("Found.")
else:
    print("Not Found.")
```

```
<re.Match object; span=(0, 17), match='Pandu is good boy'>
Found.
```

In [110…
```
import re
data = "No Pain No Gain"
a = re.findall("ain",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
['ain', 'ain']
Found.
```

- `No of Charecters : ...`

In [112…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("H...o",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
['Hello']
Found.
```

In [116…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("H..o",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
[]
Not Found.
```

In [117…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("H....o",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
[]
Not Found.
```

In [119…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("H....p",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
[]
Not Found.
```

- `Specific No of charecters : {3}`

In [113…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("H.{3}o",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
['Hello']
Found.
```

In [114…
```
import re
```

```
data = "Hello Pandu, How are you?"
a = re.findall("H.{4}o",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
[]
Not Found.
```

- `0 or More than 1 charecter: .+`

In [115…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("H.+o",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
['Hello Pandu, How are yo']
Found.
```

- `o or 1 charecter : ?`

In [130…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("H.?l",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
['Hel']
Found.
```

- `or : |`

In [121…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("Pandu|Pandi",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
['Pandu']
Found.
```

In [122…
```
import re
data = "Hello Pandu, How are you?"
a = re.findall("Bunty|Bandi",data)
print(a)
if a:
    print("Found.")
else:
    print("Not Found.")
```

```
[]
Not Found.
```

In [124…
```
data = "No Pain No Gain"
a = re.sub("Pain","Power",data)
print(a)
```

```
No  Power No Gain
```

In [126…
```
data = "No Pain No Gain"
a = re.split(" ",data)
print(a)
```

```
['No', 'Pain', 'No', 'Gain']
```

In [127…
```
data = "No Pain No Gain"
a = re.split("a",data)
print(a)
```

```
['No P', 'in No G', 'in']
```

```
```

In [128…
```
data = "No Pain No Gain"
a = re.split("",data)
print(a)
```

```
['', 'N', 'o', ' ', 'P', 'a', 'i', 'n', ' ', 'N', 'o', ' ', 'G', 'a', 'i', 'n', '']
```

- `DateTime`

- `Q11. Display all the Functions of Time()`

In [99]:
```
import datetime
a = datetime.datetime.now()
print(a)                # 2025-02-08 22:22:57.861960
print(a.strftime("%c")) # Sat Feb  8 22:22:57 2025
print("------------------------------")
print(a.strftime("%V")) # scequence of weeks Monday | Sunday
print(a.strftime("%w")) # 6
print(a.strftime("%W")) # 05
print("------------------------------")
print(a.strftime("%a")) # Sat
print(a.strftime("%A")) # Saturday
print("------------------------------")
print(a.strftime("%d")) # 08
print(a.strftime("%D")) # 02/08/25
print("------------------------------")
print(a.strftime("%b")) # Feb
print(a.strftime("%B")) # February
print("------------------------------")
print(a.strftime("%m")) # 02
print("------------------------------")
print(a.strftime("%y")) # 25
print(a.strftime("%Y")) # 2025
print("------------------------------")
print(a.strftime("%H")) # 22 Hours
print(a.strftime("%M")) # 27 Minutes
print(a.strftime("%S")) # 01 Seconds
print(a.strftime("%f")) # 861960 microseconds
print("------------------------------")
print(a.strftime("%x")) # date
print(a.strftime("%X")) # Time
print(a.strftime("%p")) # PM | AM
print(a.strftime("%I")) # 12 Hours Format
```

```
2025-02-08 22:37:26.486633
Sat Feb  8 22:37:26 2025
------------------------------
06
6
05
------------------------------
Sat
Saturday
------------------------------
08
02/08/25
------------------------------
Feb
February
------------------------------
02
------------------------------
25
2025
------------------------------
22
37
26
486633
------------------------------
02/08/25
22:37:26
PM
10
```

- `Polimorphism`

- `Q12. Create 4 classes each with same method called as DateMethod() execute 4 Methods of date in each Method of a Class.`

```
```

In [1]:
```
from datetime import datetime, timedelta

class Class1:
    def DateMethod(self):
        print("Class1:")
        print("Current Date and Time:", datetime.now())
        print("Today's Date:", datetime.today().date())
        print("Year:", datetime.now().year)
        print("Month:", datetime.now().month)
        print()

class Class2:
    def DateMethod(self):
        print("Class2:")
        today = datetime.today()
        print("Day of the Week:", today.strftime("%A"))
        print("Day of the Year:", today.timetuple().tm_yday)
        print("Week Number:", today.strftime("%U"))
        print("ISO Calendar (Year, Week, Weekday):", today.isocalendar())
        print()

class Class3:
    def DateMethod(self):
        print("Class3:")
        now = datetime.now()
        print("Current Hour:", now.hour)
        print("Current Minute:", now.minute)
        print("Current Second:", now.second)
        print("Current Microsecond:", now.microsecond)
        print()

class Class4:
    def DateMethod(self):
        print("Class4:")
        today = datetime.today()
        future_date = today + timedelta(days=10)
        past_date = today - timedelta(days=10)
        print("Date 10 Days Later:", future_date.date())
        print("Date 10 Days Ago:", past_date.date())
        print("Formatted Date:", today.strftime("%d-%m-%Y"))
        print("Time in 24-hour Format:", today.strftime("%H:%M:%S"))
        print()

# Creating objects of each class
obj1 = Class1()
obj2 = Class2()
obj3 = Class3()
obj4 = Class4()

# Executing DateMethod() of each class
obj1.DateMethod()
obj2.DateMethod()
obj3.DateMethod()
obj4.DateMethod()
```

```
Class1:
Current Date and Time: 2025-02-09 07:47:57.648482
Today's Date: 2025-02-09
Year: 2025
Month: 2

Class2:
Day of the Week: Sunday
Day of the Year: 40
Week Number: 06
ISO Calendar (Year, Week, Weekday): datetime.IsoCalendarDate(year=2025, week=6, weekday=7)

Class3:
Current Hour: 7
Current Minute: 47
Current Second: 57
Current Microsecond: 649482

Class4:
Date 10 Days Later: 2025-02-19
Date 10 Days Ago: 2025-01-30
Formatted Date: 09-02-2025
Time in 24-hour Format: 07:47:57
```

- `Polymorphism Example (Reusing Your Concept in Another Way):`

```
```

In [2]:

```python
# Defining a common interface
class BaseClass:
    def DateMethod(self):
        pass  # Abstract method

class Class1(BaseClass):
    def DateMethod(self):
        print("Class1 executing date methods...")

class Class2(BaseClass):
    def DateMethod(self):
        print("Class2 executing date methods...")

class Class3(BaseClass):
    def DateMethod(self):
        print("Class3 executing date methods...")

class Class4(BaseClass):
    def DateMethod(self):
        print("Class4 executing date methods...")

# Polymorphism in action
objects = [Class1(), Class2(), Class3(), Class4()]

for obj in objects:
    obj.DateMethod()  # Calls the respective class's implementation
```

```
Class1 executing date methods...
Class2 executing date methods...
Class3 executing date methods...
Class4 executing date methods...
```