DAY 04

DATA STRUCTURES

Creating list with same data type and different data type

```
a=[1,2,3,4,5,6,7,8,9,10]
print(a)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
b=['Tahir',3.5,7*5,True,False]
print(b)
```

```
['Tahir', 3.5, 35, True, False]
```

List Operations

1.Accesing items

```
print(a[0])
print(b[0])
```

```
1
Tahir
```

2.Modifying items

```
a[5]=50
a
b[1]='Tahir Ahmed'
b
b[2]='HUMAN'
b
```

```
['Tahir', 'Tahir Ahmed', 'HUMAN', True, False]
```

ADDING ITEMS

```
# APPEND
a.append(12)
a
```

```
[1, 2, 3, 4, 5, 50, 7, 8, 9, 10, 12]

# Insert
b.insert(3,'Being')
b

['Tahir', 'Tahir Ahmed', 'HUMAN', 'Being', True, False]
```

Removing Items

```
# Remove by items
b.remove('Tahir Ahmed')
b

['Tahir', 'HUMAN', 'Being', True, False]

# pop by index
a.pop(5)
a

[1, 2, 3, 4, 5, 7, 8, 9, 10, 12]
```

OTHER OPERATIONS

```
# len gets the number of items
len(b)

5

# sort- sorts the list
a.sort()
a

[1, 2, 3, 4, 5, 7, 8, 9, 10, 12]

# reverse :reverses the list
b.reverse()
b

[False, True, 'Being', 'HUMAN', 'Tahir']
```

***Iterating Through a List**

```
for i in b:
 print(i)

False
True
Being
```

```
HUMAN
Tahir
```

TUPLES

```
NAMES=('Tahir','Ahmed','Tahir Ahmed')
NAMES
```

```
('Tahir', 'Ahmed', 'Tahir Ahmed')
```

```
# ACCESING TUPLES
NAMES[0]
NAMES[1]
NAMES[2]
print((NAMES[0],NAMES[1],NAMES[2]))
```

```
('Tahir', 'Ahmed', 'Tahir Ahmed')
```

DICTIONARIES

```
COLLEGE={
    'NAME':'OXFORD',
    'YEAR':2023    }
COLLEGE
```

```
{'NAME': 'OXFORD', 'YEAR': 2023}
```

ACCESSING AND MODIFYING ITEMS

ACCESSING

```
print(COLLEGE['NAME'])
COLLEGE['NAME']='OXFORD UNIVERSITY'
COLLEGE['NAME']
COLLEGE
```

```
OXFORD UNIVERSITY
```

```
{'NAME': 'OXFORD UNIVERSITY', 'YEAR': 2023}
```

MODIFYING

```
COLLEGE['YEAR']=2024
COLLEGE
```

```
{'NAME': 'OXFORD UNIVERSITY', 'YEAR': 2024}
```

ADDING

```
COLLEGE['BRANCH']='COMPUTER SCIENCE'
COLLEGE
```

```
{'NAME': 'OXFORD UNIVERSITY', 'YEAR': 2024, 'BRANCH': 'COMPUTER
SCIENCE'}
```

REMOVING

```
del COLLEGE['YEAR']
COLLEGE
```

```
{'NAME': 'OXFORD UNIVERSITY', 'BRANCH': 'COMPUTER SCIENCE'}
```

ITERATING THROUGH A DICTIONARY

```
for a,b in COLLEGE.items():
 print(a,b)
```

```
NAME OXFORD UNIVERSITY
BRANCH COMPUTER SCIENCE
```

SETS

```
x={45,'string',5.5,2.5,-9,True,False}
x
```

```
{-9, 2.5, 45, 5.5, False, True, 'string'}
```

Adding items

```
x.add('Tahir Ahmed')
x
```

```
{-9, 2.5, 45, 5.5, False, 'Tahir Ahmed', True, 'string'}
```

removing items

```
x.remove(5.5)
numbers={1,2,3,4,5,6,7,8,9,10}
numbers.remove(5)
print(numbers,x)
```

```
{1, 2, 3, 4, 6, 7, 8, 9, 10} {False, True, 'string', 'Tahir Ahmed', -
9, 45}
```

SET OPERATIONS

```
#Union : set1 | set2
print(numbers | x)

{False, 1, 2, 3, 4, 6, 7, 8, 9, 10, 45, 'string', 'Tahir Ahmed', -9}

# Intersection
u={1,2,3,4,5,6,7,8,9,10}
v={1,2,3,4,5,6,7,8,9,10,11,12,13,'Tahir Ahmed'}
u & v

{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

# DIFFERENCE
v-u

{11, 12, 13, 'Tahir Ahmed'}

u-v

set()
```

HANDS ON PRACTICE

1.MANIPULATING LISTS

```
fruits=['apple','banana','orange','mango','kiwi']
fruits.append('pineapple')
fruits

['apple', 'banana', 'orange', 'mango', 'kiwi', 'pineapple']
```

2.CREATING LISTS

```
BOOK={
    'TITLT':"PYTHON PROGRAMMING",
    'AUTHOR':"TAHIR AHMED",
    'YEAR':2023
}
BOOK

{'TITLT': 'PYTHON PROGRAMMING', 'AUTHOR': 'TAHIR AHMED', 'YEAR': 2023}
```

3.WORKING WITH SETS

```
set1={1,2,3,4,5,6,7,8,9,10}
set2={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}
print("Union:",set1 | set2)
print("Intersection:",set1 & set2)
print("Difference:",set1 - set2)

Union: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}
Intersection: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
Difference: set()
```

4.Merge two lists

```
list1=[1,2,3,4,5]
list2=[6,7,8,9,10]
merged_list=list1+list2
print(merged_list)

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

5.DICTIONARY OPERATIIONS

```
STUDENT={"name":"TAHIR AHMED","AGE":20,"MARKS":75}
print("NAME:",STUDENT["name"])
STUDENT["MARKS"]=90
print("UPDATED MARKS:",STUDENT["MARKS"])

NAME: TAHIR AHMED
UPDATED MARKS: 90
```

6.FIND THE MAXIMUM AND MINIMUM IN A LIST

```
numbers=[1,2,3,4,5,6,7,8,9,10]
maximum=max(numbers)
minimum=min(numbers)
print("Maximum:",maximum)
print("Minimum:",minimum)

Maximum: 10
Minimum: 1
```

7.COUNT Frequency of Elements in a list

```
numbers=[1,2,3,4,5,6,7,8,8,9,10]
frequency={}
for number in numbers:
  frequency[number]=frequency.get(number,0)+1
print("Frequency of elements:",frequency)

Frequency of elements: {1: 1, 2: 1, 3: 1, 4: 1, 5: 1, 6: 1, 7: 1, 8:
2, 9: 1, 10: 1}
```

8.Sort a list of tuples by the second Element

```
tuples=[('a',5),('b',2),('c',8),('d',1)]
sorted_tuples=sorted(tuples,key=lambda x:x[1])
print("Sorted Tuples:",sorted_tuples)
```

```
Sorted Tuples: [('d', 1), ('b', 2), ('a', 5), ('c', 8)]
```

PRACTICE QUESTIONS FROM CHAT GPT

Q1: What is the output of the following code?

lst = [1, 2, 3, 4, 5]

print(lst[2:4])

```
lst = [1, 2, 3, 4, 5]
print(lst[2:4])
```

```
[3, 4]
```

Q2:List comprehensions

```
squares = [x**2 for x in range(5)]
print(squares)
```

```
[0, 1, 4, 9, 16]
```

Q3:Checking if an item exists in a list

```
fruits = ["apple", "orange", "mango"]
print("apple" in fruits)
```

```
True
```

Q4:Tuple unpacking

```
a, b, c = (4, 5, 6)
print(a, b, c)
```

```
4 5 6
```

Q5:Finding an index in a tuple

```
vowels = ('a', 'e', 'i', 'o', 'u')
print(vowels.index('i'))
```

```
2
```

Q6:Updating a value in a dictionary

```
data = {"name": "Bob", "age": 25}
data["age"] = 26
print(data)
```

```
{'name': 'Bob', 'age': 26}
```

Q7:Checking for key existence in a Dictionary

```
user = {"username": "admin", "active": True}
print("password" in user)
```

```
False
```

PALINDROME NUMBER

```
number=int(input("Enter a Number:"))
reverse=0
temp=number
while temp>0:
  digit=temp%10
  reverse=reverse*10+digit
  temp=temp//10
if reverse==number:
  print(f"{number} is Palindrome")
else:
  print(f"{number} is Not Palindrome")
```

```
Enter a Number:121
121 is Palindrome
```

PALINDROME FOR STRING

```
NUMBER=input("Enter a Number:")
if NUMBER==NUMBER[::-1]:
  print(f"{NUMBER} is Palindrome")
else:
  print(f"{NUMBER} is Not Palindrome")
```

```
Enter a Number:SOS
SOS is Palindrome
```

PALINDROME FROM LEET CODE

```
    def isPalindrome(x):
        if x < 0 or (x % 10 == 0 and x != 0):
            return False
        reversed_half = 0
        while x > reversed_half:

            reversed_half = reversed_half * 10 + x % 10
            x //= 10


        return x == reversed_half or x == reversed_half // 10
    isPalindrome(121)
```

```
True
```