Important Points

for loop vs while loop

In [ ]:
```
for loop and while:
    for loop --> if we know the number of iterations.(Arrays)
    while loop --> if there is any condition based on that you want to execute a group of statement then you are
    going to use while loop(Linked list)
```

classes and objects

In [ ]:
```
"""classes and object
class --> blueprint of an object
object --> an instnace of a class"""
class className:
    def __init__(self,x):
        self.x=x

    def m1(self):
        print(self.x)
C=className(10)
C.m1()
```

What is Data Structures?

In [ ]:
```
--> Data Structure:is a way of storing the data so that we can use that data in an  efficient manner for our furthur use.
```

Types of Data Structures

In [ ]:
```
Types of Data Structures:

    1. Primitive Data Structure(Hold a single value) --> int,float,long,bool complex.....
    2. Non Primitive Data Structure(Hold multiple values)

    --> Based on Implemetaion
                        -->  Physical
                                    --> Array linkedlist
                        -->  logical(ADT)
                                    --> Stack and Queue,Graph and Trees

    --> Based on Storage
                        --> Linear Data Structures
                                    --> Array,linkedlist, Stack and Queue
                        --> Non Linear Data Structures
                                    --> Trees and Graphs
```

Abstract DataTypes(ADT)

In [ ]:
```
--> Abstract Data Type(ADT) is a data type, where only behavior is defined but not implementation.
--> Opposite of ADT is Concrete Data Type (CDT), where it contains an implementation of ADT
--> Abstract Datatype(Stack and Queue):like a set of operations that can be performed but without specifying how these
    operations are implemented.
--> Entities which are having partial implementation or data and method are known as ADT.

        Stack --> push,pop,peek --> Array and linkedlist
        Queue --> enqueue,dequeue --> array and linkedlist
```

Physical and Logical Data Structures

In [ ]:
```
Physical Data Structure --> are those data structures by which we can implement any other data structure.

Logical Data STructure --> are those data structure which are totally depend on physical Data Structure for implemenation.
```

Linear and Non Linear Data Structures

In [ ]:
```
--> Linear means data will be stored in sequential manner.--> Arrays linkedlist queue and stack
--> Non Linear means data will be store in non sequential manner. --> Trees and Graphs
```

Arrays

In [ ]:
```
Array --> Array is linear data structure which store similar kind of data. Elements in Array are stored in contigous memeory
location. Elements will be stored at continous memeory location.

List --> List is a collection of disimilar elements. List is an Object.

     --> In python for implementing Array we are going to consider list as an array.
     --> In python list are more feasible as compare to array.
```

Basic Operations of Any Data Structures

In [ ]:
```
Basic Operations of every data structures:
    1.Insertion --> Inserting an element in array(start,end,pos)
    2.Deletion --> Deletion of element im array(start,end,pos)
    3.Traversal  -->(Accessing) Visiting each and every element of the array.
```

Implementation of different operations of an array

In [1]:
```
#Implementation of different operations in array

class Array:
    def __init__(self):
        self.array=[]

    def insertion_at_begin(self,data):
        self.array.insert(0,data)
        return self.array

    def insertion_at_end(self,data):
        self.array.append(data)
        return self.array

    def insertion_at_pos(self,data,pos):
        if pos>len(self.array):
            return self.array
        self.array.insert(pos,data)
        return self.array

    def deletion_at_begin(self):  #[]
        if len(self.array)<1:
            return self.array

        self.array.pop(0)
        return self.array

    def deletion_at_end(self):
        if len(self.array)<1:
            return self.array
        self.array.pop()
        return self.array

    def deletion_at_pos(self,pos):
        if len(self.array)<1 or pos>len(self.array):
            return self.array
        self.array.pop(pos)
        return self.array

    def traversal(self):
        for i in self.array:
            print(i)

s=Array()
s.insertion_at_begin(10)
s.traversal()
print("**********************************************************************")
s.insertion_at_begin(20)
s.traversal()
print("**********************************************************************")
s.insertion_at_begin(30)
s.traversal()
print("**********************************************************************")
s.insertion_at_pos(50,0)
s.traversal()
print("**********************************************************************")
s.insertion_at_pos(50,4)
s.traversal()
print("**********************************************************************")
s.insertion_at_end(100)
s.traversal()
print("**********************************************************************")
s.deletion_at_begin()
s.traversal()
print("**********************************************************************")
s.deletion_at_end()
s.traversal()
print("**********************************************************************")
s.deletion_at_pos(2)
s.traversal()
```

```
10
**********************************************************************
20
10
**********************************************************************
30
20
10
**********************************************************************
50
30
20
10
**********************************************************************
50
30
20
10
50
**********************************************************************
50
30
20
10
50
100
**********************************************************************
30
20
10
50
100
**********************************************************************
30
20
10
50
**********************************************************************
30
20
50
```

List vs Array

In [ ]:
```
Python List:

    --> Lists are the inbuilt data structure of python.
    --> We can store elements of different types in the list.
    --> Items in the list are enclosed between square brackets and separated by commas.
    --> We can even nest lists with other data structures like lists, dictionaries, tuples, etc.

Python Array:

    --> Arrays are not the in-built data structure readily available in python.
    --> We must import the array from the 'array' or 'numpy' module.
    --> Array stores elements of similar types. If we try to store elements of different types, it will throw an error.
    --> We can only store elements of the same types in the array.
```

Examples

Traversal Using List

In [3]:
```
x=[10,20,30]
for i in x:
    print(i)
```

```
10
20
30
```

Traversal Using Array

In [2]:
```
import array
sample_array = array.array("i",[10,20,30])
for i in sample_array:
    print(i)
```

```
10
20
30
```

Applications of Arrays

In [ ]:
```
--> Arrays are used to implement data structures like a stack, queue, etc.
--> Arrays are used for matrices and other mathematical implementations.
--> Arrays are used in lookup tables in computers.
--> Arrays can be used for CPU scheduling.
--> Arrays are commonly used to store large amounts of data.
--> Arrays are used in the implementation of various graph and tree data structures.
--> Arrays can be used to dynamically allocate memory for storing data.
--> Arrays are used in computer graphics to store and manipulate images, pixel data, and other visual elements.

Real Time Application of Arrays:

    --> Contact lists on mobile phones.
    --> Matrices use arrays which are used in different fields like image processing, computer graphics, and many more.
    --> Arrays are used in online ticket booking portals.
    --> Pages of book.
    --> IoT applications use arrays as we know that the number of values in an array will remain constant, and also that the
        accessing will be faster.
    --> It is also utilised in speech processing, where each speech signal is represented by an array.
    --> The viewing screen of any desktop/laptop is also a multidimensional array of pixels.
```