Data Structures

Data Structure:- It is a meaningful way of arranging, and storing the data in the computer for efficient utilization and processing depending upon the situation.

Types of data Structure in python:-
There are two types of data structure

1. Built-in Data Structures
2. User-Defined Data Structures.

- **Built-in** data structures contains List, Dictionary, Tuple, Sets

- **User-Defined** data structure Contains Stack, Queue, Trees, Linked list, Graphs, Hashmap

Data Structure is one of the most fundamental topics of any programming language.

@programmer-girl--

Linear Vs Non-linear

Linear data-structure	Non-linear data structure
Element are arranged sequentially	Element are arranged hierarchically.
Easy to implement	Difficult to implement
Memory utilization is not efficient	Memory utilization is efficient
Single level data-structure	Multilevel data structure
Examples:- Array, Stack, Queue, Linkedlist	Example:- Graph, tree heap.

@programmer-girl--

Built-in Data Structures :-

1. **List :-** A list is a collection of an ordered, mutable of mixed datatype separated with comma and enclosed in square bracket on zero indexing method.
   - **Ordered :-** Elements have a defined order.
   - **Mutable :-** change even after creation.
   - **Mixed data type :-** list = [1, 2.5, "gir"]

→ **Method of lists :-** list(), append(), insert(), extand(), sort(), reverse(), remove(), pop(), clear() etc.

@programmer-girl--

2. **Dictionary :-** Dictionary is the collection of key-value pairs. which is created by using multiple key-value pairs enclosed with the curly brackets {}, and each key is separated from its value by the colon (:), one key value pair is separated by comma (,) from another
   - Dictionary is **mutable**.
   - Keys must be **immutable** and **unique**.
   - value can be mutable and immutable.

dictionary = {"key": 1}

3. **Tuple:-** A tuple is a collection of an ordered, immutable, of mixed datatype seperated with comma and enclosed in round brackets based on zero indexing method.

→ only difference between list and tuple is that list are mutable and tuple are immutable.

→ **Tuple methods:-** Count(), Sorted(), tuple(), min(), max(), len(), Del() etc.

4. **Sets:-** A set is a collection of unordered values or items. We can put heterogeneous type values inside the sets.

→ **Sets Method :-** set(), add(), update(), remove(), pop(), discard(), clear(), union() etc.

@programmer-girl--

Time complexity

→ **Time complexity :-** It is defined as the amount of time taken by an algorithm to execute each statement of code of an algorithm till its completion with respect to the function of the length of the input.

→ The three important **asymptotic notations** of Time Complexity are :-

- **O :** This notation defines if functions grow slower than or at the same rate with respect to the expression.

- **Ω :** This notation defines if functions grow faster than or at the same rate with respect to the expression.

- **θ :** This notations defines if functions' growth lie in both O and Ω.

→ The different types of Time complexity of sorting algorithms are:-

- Insertion Sort :-
  Best case :- $\Omega(n)$
  Wrost case :- $O(n^2)$

- Merge Sort :-
  Best case :- $\Omega(n \log n)$
  Wrost case :- $O(n \log n)$

- Quick Sort :-
  Best case :- $\Omega(n \log n)$
  Wrost case :- $O(n^2)$

- Bubble Sort :-
  Best case :- $\Omega(n)$
  Wrost case :- $O(n^2)$

@programmer-girl--

Stack

Stack:- A stack is an abstract data type that holds linear sequence of items. Stack follow Last in First out.

Real life Example:- Stack of paper

Stack operations:- @programmer-girl--

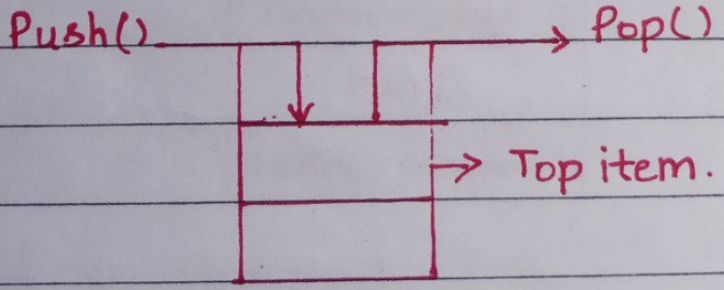
Push :- Add element

Pop :- Remove element

Peek :- Copy of the top element.

is-empty :- Check the stack empty or not.

is-full :- Check the stack is Full or not



Stack.

Let's discuss the most important concept →

Stack Implementation

```
stack = []
def push():
    element = input("Enter the element")
    stack.append(element)
    print(stack)
def pop-element()
    if not stack:
        print ("stack is empty")
    else:
        ele = stack.pop()
        print ("removed element:" ele)
        print (stack)
```

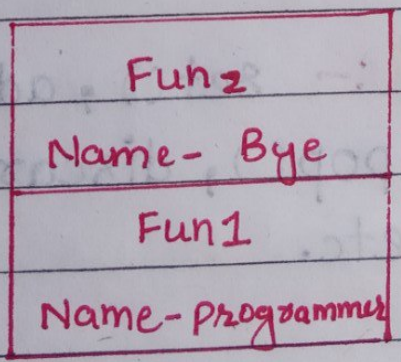
Call stack:

Call stack:- It is a stack which is used to save the variables of multiple functions is called the call stack

For Example:-

```
def Fun1(name):
    Print("Programmer")
    Fun2(name)
    Print("Bye")
    Fun3()
```

Fun1 calls two other Function Fun2, Fun3.

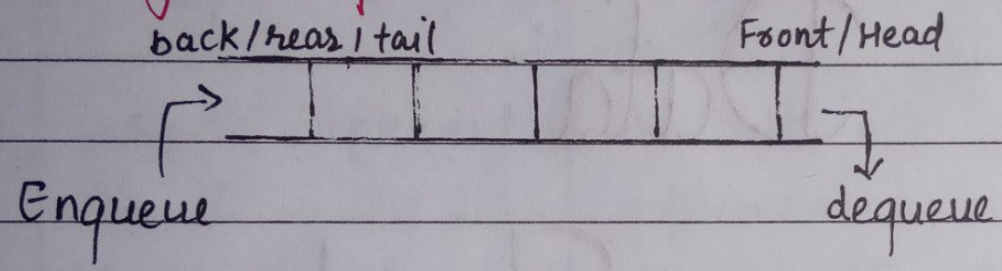


@programmer-girl--

Queue

Queue:- It is abstract data structure in which element inserted First will be removed first.

Real-life Example:- Ticket Queue



Queue Basic operation:-

Enqueue:- Add element to the rear end.

Dequeue:- Remove element from the front end.

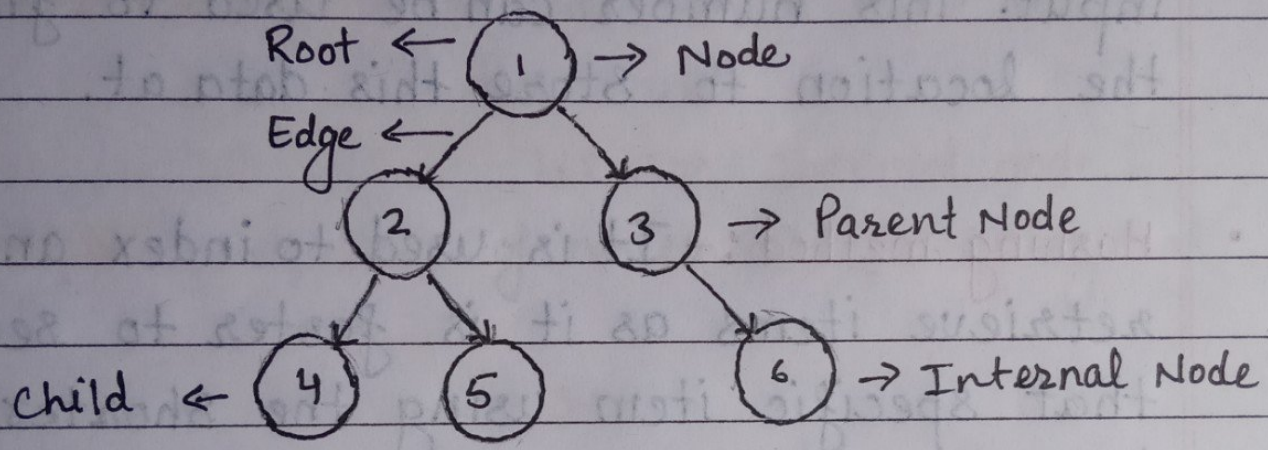
IsEmpty:- Check the queue is empty or not.

IsFull:- Check the queue is Full or not

Peek:- Get the value of front

Tree

Tree:- It is a non-linear data-structure which represents relationship between nodes.



Tree Terminologies:- @programmer_girl--

Root:- Origin node of tree.
1 is the root node

Edge:- Connection between two nodes.

Parent:- which has a child from it to any other node.
1, 2, 3 → Parent node

Child:- Any subnode of a given node.
Expect root node, Every node is child node.

Siblings:- Node with the same parent.

2 and 3, 4,5

Leaf:- Node with no child (External node)

4,5,6 → leaf node

Internal node:- A node which have atleast one child.

1,2,3 → Internal node.

Path:- Sequence of nodes along the edge of tree.

1 → 3 → 6

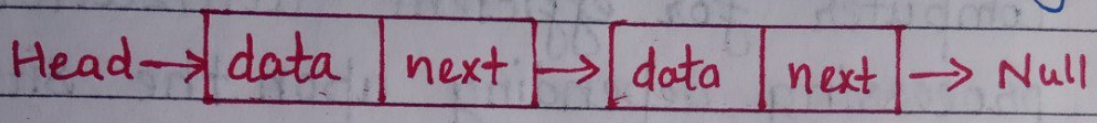
Real life Example:- A Family tree.

Types of Tree:-

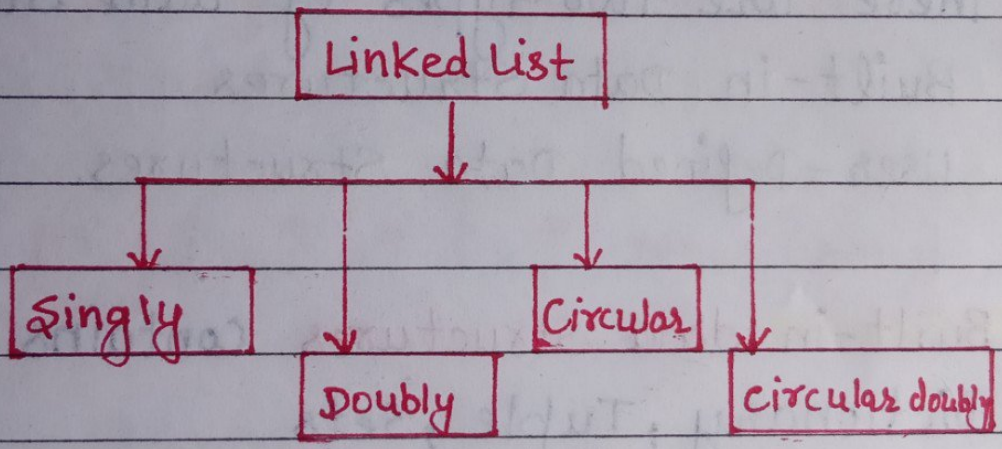
1. Binary Tree @programmer_girl--
2. Binary Search Tree
3. AVL Tree
4. B- Tree

Linked list

Linked list :- It is a linear datastructure, where elements are connected using pointers



Linked list allow fast inserts and deletes.



Memory allocation :- During Runtime.

Linked list stored elements randomly.
Linked list required more memory to store data in efficient way.

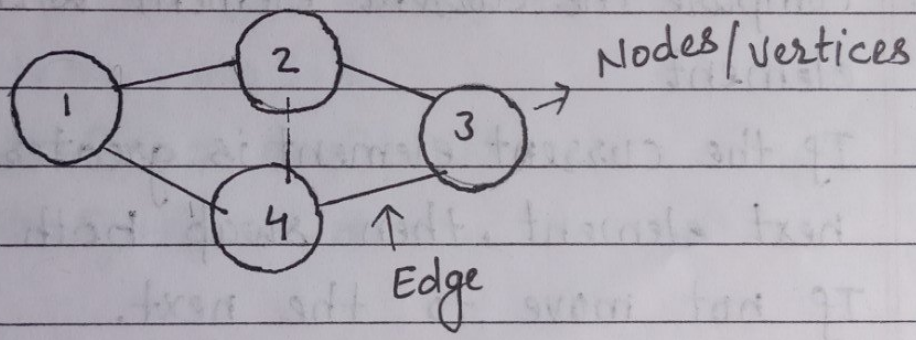
@programmer-girl--

Graph

@programmer-girl--

Graph:- It is a non-linear data structure made up of a finite number of nodes or vertices and the edges that connect them.

Real life Example:- GPS system, Social networks
Circuit networks



Every tree are graph But every graph are not tree.

Types of Graph :-

Null graph
Trivial graph
Non-directed graph
Directed graph
Connected graph
Disconnected graph
Cyclic graph
Acyclic graph
Finite graph
infinite graph
Planer graph
Simple graph

Hashing

Hashing :- It is a technique to find a small number from some data provided as an input. This number can be used to find the location to store this data at.

- Hashing method :- It is used to index and retrieve items as it is faster to search that specific item using the shorter hashed key insted of using its original value.

- Data bucket :- Key, Hash function, linear probing, Quadratic probing, Hash index, collisions are important terminologies used in hashing.

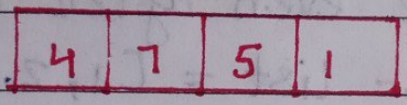
- Collision :- In case the resultant index for 2 different inputs comes to be the same, it's called a collision

Selection Sort

Selection Sort :- It takes the smallest item from an unsorted array and place at the beginning of array.

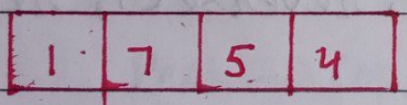
@programmer-girl--

Step :- I



Front item ↑ ——— ↑ Smallest item

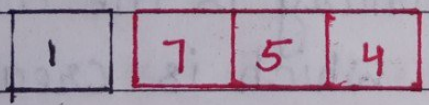
↓ Swap.



→ New array.

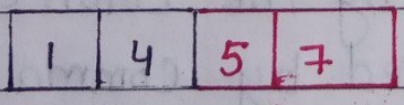
Sorted item ←

Step :- II



Front item ↑ ——— ↑ Smallest item

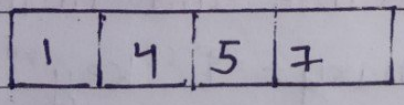
↓ Swap.



→ New array.

Sorted items ←┘ └→ Unsorted items

Step III



→ Already sorted.

Selection Sort

Selection sort :- It takes the smallest item in an unsorted array

Selection sort complexity :-

Time complexity :- $O(n^2)$

Space Complexity :- $O(1)$

→ Selection sort is easy to implement

→ Runnig time of selection sort is very poor.

Bubble Sort Algorithm

→ **Bubble Sort:-** It is also known as Sinking Sort. It works by repeatedly Comparing the adjacent elements, if they are in wrong order.

Steps to Sort:-

1. Start with the first element.
2. Compare the current element with the next element.
3. If the current element is greater than the next element, then swap both elements. If not move to the next.

For Example:- 4, 3, 0.

First Iteration:-

4, 3, 0.

3 4 0.

3 0 4

Second Iteration:-

3 0 4

0 3 4

0 3 4 → Sorted

Bubble sort algorithm is very slow.

Time Complexity :- $O(n^2)$

Space Complexity :- $O(1)$

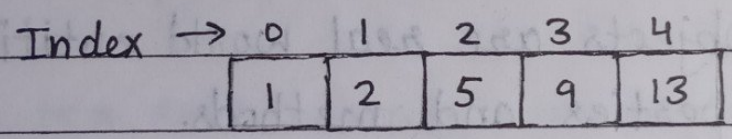
→ let's do code for Bubble Sort

```
def bubblesort(arr):
    for i in range(len(arr)):
        for j in range(0, len(arr)-(i-1));
            if arr[j] > arr[j+1]:
            tem = arr[j]
            arr[j] = arr[j+1]
            arr[j+1] = tem
```

Binary Search algorithms

Binary Search:- Binary Search is used to get the position of searched element from the sorted list or sorted array.

Real life Example:- Searching for a word in a dictionary.
Note the point, dictionary is sorted.



L-Index = 0

H-Index = len(list) − 1 = 4

In Binary Search, Each time, you check the middle Element.

mid = (L-Index + H-Index)/2 ⇒ 0+4/2 ⇒ 2

Guess = list[mid] ⇒ list[2] ⇒ 5

If Guess is too low :- ∴ Guess is too high.

Guess < item

low = mid+1

Guess > item

high = mid−1

Run Time of Binary Search :- O(log n)