Data Structures

<u>*Topics to be covered*</u>

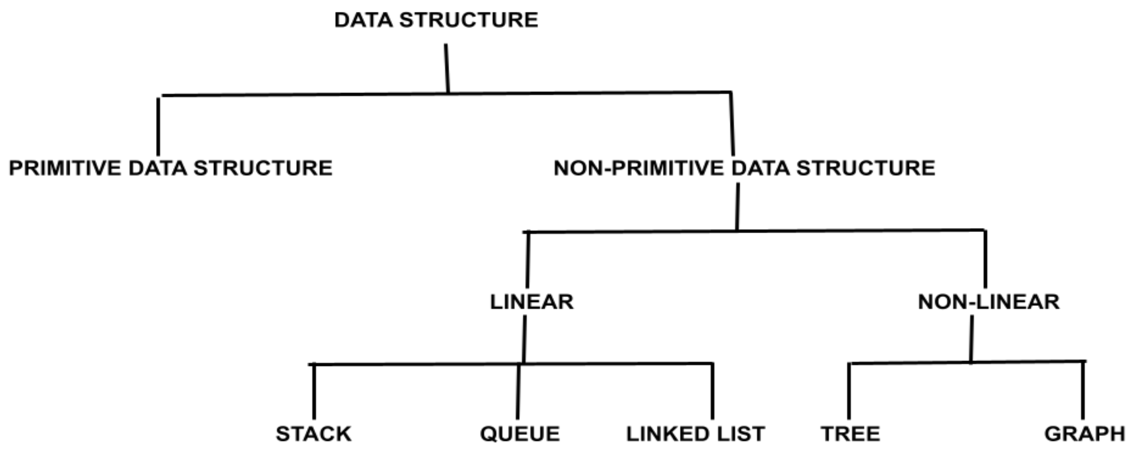
1. *Stack*
2. *Operations on stack (push, pop, peek, display)*
3. *Implementation of stack using list*
4. *Applications of stack*

DATA STRUCTURE

Data structure can be defined as a set of rules and operations to organize and store data in an efficient manner. We can also say that it is a way to store data in a structured way. We can apply different operations like reversal, slicing, counting etc. of different data structures. Hence, Data Structure is a way to organize multiple elements so that certain operations can be performed easily on whole data as a single unit as well as individually on each element also.

In Python, Users are allowed to create their own Data Structures which enable them to define the functionality of created data structures. Examples of User Defined data structures in Python are Stack, Queue, Tree, Linked List etc. There are some built-in data structures also available in Python like List, Tuple, Dictionary and Set.

Types of Data Structure:



Primitive Data Structures

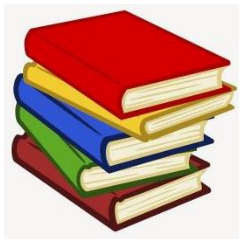
Primitive Data Structures contain simplified data values and are directly operated by machine level instructions. For example, integer, real, character etc. are primitive data structures.

Non-Primitive Data Structures

Non-Primitive Data Structures are derived from the primitive data structures. These are of two types: **Linear data structures,** that are single level data structures having their elements in a sequence like stack, queue and linked list while **Non-linear data structures** that are multilevel data structures like tree and graph.

STACK:

A Stack is a Linear data structure which works in LIFO (Last In First Out) manner (or we can say FILO i.e. First In Last Out manner). It means that Insertion and Deletion of elements will be done only from one end generally known as TOP only. In Python, we can use List data structure to implement Stack.



Application of Stack:

1. Expression Evaluation
2. String Reversal
3. Function Call
4. Browser History
5. Undo/Redo Operations

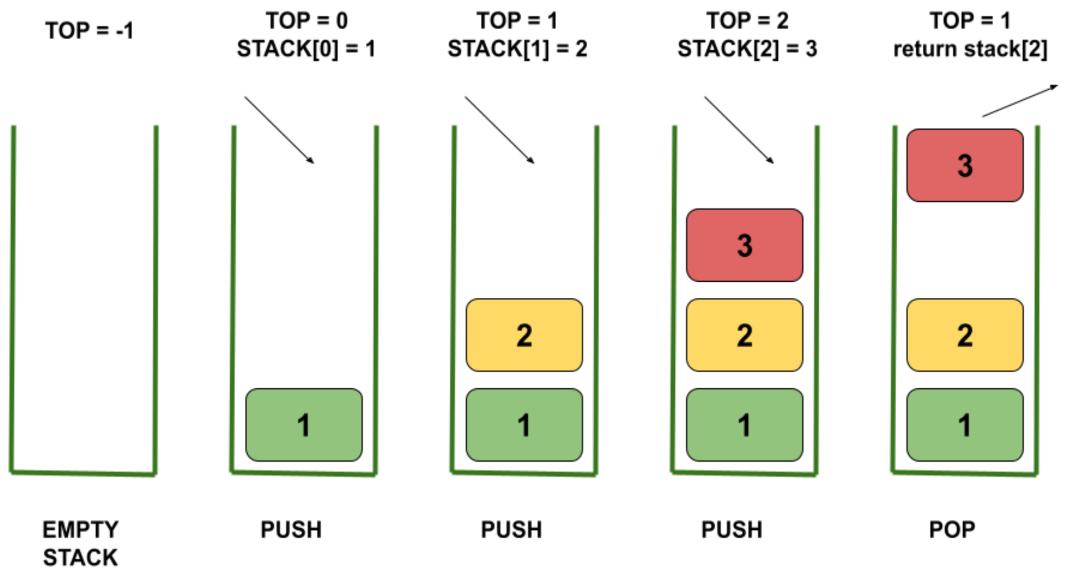
Operations of Stack:

The Stack supports following operations:

1. **Push:** It adds an element to the TOP of the Stack.
2. **Pop:** It removes an element from the TOP of the Stack.
3. **Peek:** It is used to know/display the value of TOP without removing it.
4. **isEmpty:** It is used to check whether Stack is empty.

OVERFLOW: It refers to the condition in which we try to PUSH an item in a Stack which is already FULL.

UNDERFLOW: It refers to the condition in which we are trying to POP an item from an empty Stack.



Implementation of Stack:

In Python, List data structure is used to implement Stack. For PUSH operation we use **append()** method of List while for POP operation we use **pop()** method of List.

PROGRAM: To illustrate the basic operations of Stack:

```
def Push(emp,el): #method to add an element at TOP of stack
    top = len(emp) - 1
    emp.append(el)
    print(top)
    print("Data inserted successfully")
    return top
def Pop(emp):  #method to delete last element (TOP) of stack
    if isEmpty(emp):
        print("Stack is empty... Underflow Case")
    else:
        print("Deleted data is: ",emp.pop())
def Peek(emp): #method to display element at TOP
    if isEmpty(emp):
        print("Stack is empty...Nothing to Display")
    else:
        top = len(emp) - 1
        print("The last data added is: ",emp[top])

def Display(emp): #methjod to display stack
    if isEmpty(emp):
        print("Stack is empty... Nothing to Display")
    else:
        top = len(emp)
        d=emp[::-1]   #reversal of list done to display the TOP element first
        print("Data in stack is as follows: ")
        for i in d:
            print(i)

def isEmpty(emp):  #method to check stack is empty or not.
    if len(emp)==0:
        return True
    else:
        return False
```

```
emp=[]
top=None
while True:
    print("Stack operations")
    print("1. Add employee data.")
    print("2. Delete employee data.")
    print("3. Display employee data.")
    print("4. Display last added data.")
    print("5. Exit")
    ch=int(input("Choose operation on stack:"))
    if ch==1:
        e_no=int(input("Enter employee number: "))
        e_name=input("Enter employee name: ")
        data=[e_no,e_name]
        Push(emp,data) #method calling
    elif ch==2:
        Pop(emp)
    elif ch==3:
        Display(emp)
    elif ch==4:
        Peek(emp)
    elif ch==5:
        break
    else:
        print("Invalid choice")
```

2. A list contains following record of customer: [CBSE EXAM 2022-23]

   [Customer_name, Room_type]

   Write the following user defined functions to perform given operations on the stack

named “Hotel”:

i. Push_Cust() - To Push customer’s names of those customers who are staying in ‘Delux’ Room Type.

ii. Pop_Cust() - To Pop the names of Customers from the stack and display them. Also, display “Underflow” when there are no customers in the stack.

For example: If the list with customer details are as follows:

[“Siddarth”, “Delux”]

[“Rahul”, ”Standard”]

[“Jerry”, “Delux”]

The stack should contain:

Jerry

Siddharth

The output should be:

Jerry

Siiddharth

Underflow

```
Hotel = []
Customer = [['Siddarth', 'Delux'], ['Rahul', 'Standard'], ['Jerry', 'Delux']]
def Push_Cust():
    for rec in Customer:
        if rec[1] == 'Delux':
            Hotel.append(rec[0])

def Pop_Cust():
    while len(Hotel) > 0:
        print(Hotel.pop())
    else:
        print("Underflow")

Push_Cust()
Pop_Cust()
```

3. Write a function, Push (Vehicle) where, Vehicle is a dictionary containing details of vehicles – {Car_Name: Maker}. The function should push the name of car manufactured by ‘TATA’ (including all the possible cases like Tata, TaTa, etc.) to the stack. For example:

If the dictionary contains the following data:

Vehicle = {‘Santro’: ‘Hyundai’, ‘Nexon’: ‘TATA’, ‘Safari’ : ‘Tata’}

The stack should contain:

Safari

Nexon [CBSE EXAM 2022-23]

```
stack=[]
Vehicle = {'Santro':'Hyundai', 'Nexon':'TATA', 'Safari':'Tata'}
def Push(Vehicle):
    for v_name in Vehicle:
        if Vehicle[v_name].upper() == 'TATA':
            stack.append(v_name)
Push(Vehicle)
x = stack[::-1]    #reversal to print stack
for i in x:
    print(i)

                            OR

stack=[]
Vehicle = {'Santro':'Hyundai', 'Nexon':'TATA', 'Safari':'Tata'}
def Push(Vehicle):
    for v_name in Vehicle:
        if Vehicle[v_name] in ('TATA', 'TaTa', 'tata', 'Tata')
            stack.append(v_name)
Push(Vehicle)
x = stack[::-1]    #reversal to print stack
for i in x:
    print(i)
```

QUES: A list, NList, contains following record as list elements:

[City, Country, distance from Delhi]

Each of these records are nested together to form a nested list. Write the following user defined functions in Python to perform the specified operations on the stack named travel.

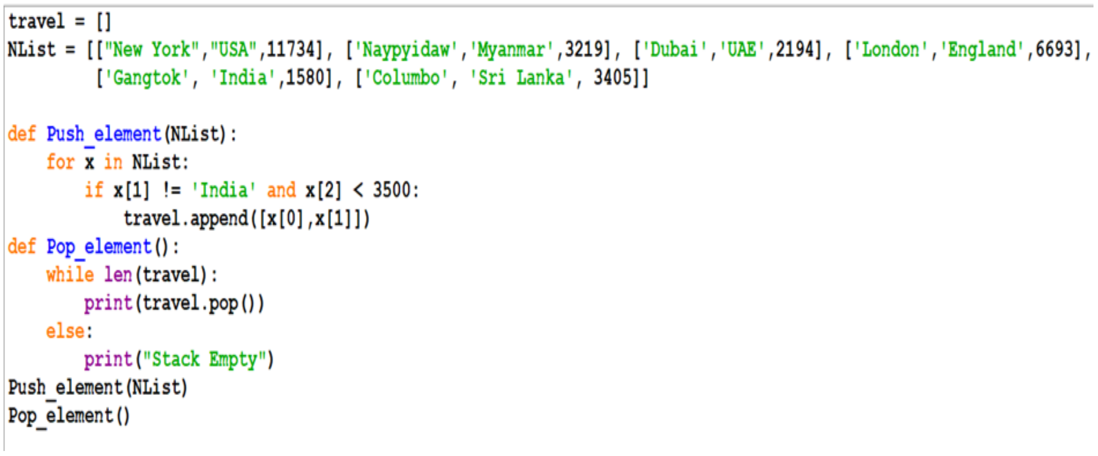
i. **Push_element(NList):** It takes the nested list as an argument and pushes a list object containing name of the city and country, which are not in India and distance is less than 3500 km from Delhi.

ii. **Pop_element():** It pops the objects from the stack and displays them. Also, the function should display “Stack Empty” when there are no elements in the stack.

For example: If the nested list contains the following data:
NList=[[“Newyork”,”USA”,11734], [“Naypyidaw”, “Myanmar”,3219], [“Dubai”,”UAE”,2194], [“London”,”England”,6693], [“Gangtok”, “India”,1580], [“Columbo”, ”Sri Lanka”, 3405]]

The stack should contain:
[“Naypyidaw”, “Myanmar”,3219], [“Dubai”,”UAE”,2194], [“Columbo”, ”Sri Lanka”, 3405]

The output should be:
[“Columbo”, ”Sri Lanka”, 3405]
[“Dubai”,”UAE”,2194]
[“Naypyidaw”, “Myanmar”,3219]
Stack Empty [CBSE SQP 2023]



```
travel = []
NList = [["New York","USA",11734], ['Naypyidaw','Myanmar',3219], ['Dubai','UAE',2194], ['London','England',6693],
        ['Gangtok', 'India',1580], ['Columbo', 'Sri Lanka', 3405]]

def Push_element(NList):
    for x in NList:
        if x[1] != 'India' and x[2] < 3500:
            travel.append([x[0],x[1]])
def Pop_element():
    while len(travel):
        print(travel.pop())
    else:
        print("Stack Empty")
Push_element(NList)
Pop_element()
```

Ques: Write a function in Python, Push(SItem) where, SItem is a dictionary containing the details of stationary items—{Sname:price}. The function should Push the names of those items in the stack who have price greater than 75. Also display the count of elements pushed into the stack.

For example:

If th dictionary contains the following data:

SItem: {'Pen': 106, 'Pencil': 59, 'Notebook': 80, 'Eraser':25}

The stack should contain:

Notebook

Pen

The output should be : The count of elements in the stack is 2. [CBSE SQP 2022]

```
stackItem = []
SItem = {'Pen': 106, 'Pencil': 59, 'Notebook': 80, 'Eraser':25}
def Push(SItem):
    count = 0
    for k in SItem:
        if (SItem[k] >= 75):
            stackItem.append(k)
            count = count + 1
    print("The Count of elements in the stack is ", count)

Push(SItem)
```

Assignment

1 Sanya wants to remove an element from empty stack. Which of the following term is related to this?

(a) Empty Stack (b) Overflow (c) Underflow (d) Clear Stack

2 In Stack, all operations takes place at____________.

1. Top
2. Front
3. Rear
4. Any

3 Insertion and Deletion operations of Stack are known as ___________ respectively.

A. Insertion and Deletion
B. Push and Pop
C. Pop and Push
D. Enqueue and Dequeue

4 When a Stack is empty, the TOP will reset to:

- NONE
- 1
- -1
- None of the above.

5 Which of the following is not an inherent application of Stack?

- Reversing a String.
- Evaluation of postfix expression.
- Implementation of recursion.
- Job Scheduling.

6 Data structures are:

- Network structures.
- Group of data
- Different types of data
- Different operations on data

ASSERTION & REASON:

(A): Both A and R are true and R is the correct explanation for A.
(B): Both A and R are true and R is not correct explanation for A.
(C): A is true but R is false.
(D): A is false but R is true.

- **ASSERTION (A):** Using append(), many elements can be added at a time.
  REASON(R): For adding more than one element, extend() method can be used.

- **ASSERTION (A):** A data structure is a named group of data types.
  REASON(R): A data structure has a well-defined operations, behaviour and properties.

- **ASSERTION (A):** LIFO is a technique to access data from queues.
  REASON(R): LIFO stands for Last In First Out.

- **ASSERTION (A):** A Stack is a Linear Data Structure, that stores the elements in FIFO order.
  REASON(R): New element is added at one end and element is removed from that end only.

- **ASSERTION (A):** An error occurs when one tries to delete an element from an empty stack.
  REASON(R): This situation is called an Inspection.

- **ASSERTION (A):** A stack is a LIFO structure.

  REASON (R): Any new element pushed into the stack always gets positioned at the index after the last existing element in the stack.

8 Write a function in Python POPSTACK (L) where L is a stack implemented by a list of numbers. The function returns the value deleted from the stack.

9 A list contains following record of a customer:
[Customer_name, Phone_number, City]
Write the following user defined functions to perform given operations on the stack named 'status':
(i) Push_element() - To Push an object containing name and Phone number of customers who live in Goa to the stack
(ii) Pop_element() - To Pop the objects from the stack and display them. Also, display "Stack Empty" when there are no elements in the stack.
For example: If the lists of customer details are:
["Gurdas", "99999999999","Goa"]
["Julee", "8888888888","Mumbai"]
["Murugan","77777777777","Cochin"]
["Ashmit", "1010101010","Goa"]
he stack should contain
["Ashmit","1010101010"]
["Gurdas","9999999999"]
The output should be:
["Ashmit","1010101010"]
["Gurdas","9999999999"] Stack Empty

10 Write a function in Python, Push(SItem) where , SItem is a dictionary containing the details of stationery items– {Sname:price}.
The function should push the names of those items in the stack who have price greater than 75.
Also display the count of elements pushed into the stack.
For example: If the dictionary contains the following data: Ditem={"Pen":106,"Pencil":59,"Notebook":80,"Eraser":25}
The stack should contain:
Notebook
Pen
The output should be:

11 Create a stack that stores dictionaries as elements. Each dictionary represents a person's information
(name, age, city).
- Implement a `push_dict` method to push a dictionary onto the stack of person above age 20
- Implement a `pop_dict` method to pop the top dictionary from the stack.

12 - Create a stack to manage student records. Each student record is represented as a dictionary containing attributes like student ID, name, and GPA.

- Implement a `push_student` method to push student records onto the stack that GPA above 60.

- Implement a `pop_student` method to pop the top student record from the stack.

13 Assume a nested dictionary .Each dictionary can contain other dictionaries as values.the format of the dictionary is as follows:

{1:{‘a’:’one,’b’:’two},2:{‘x’:10},3:{‘y’:100,’z’:200}....}

Create a stack that stores dictionaries as elements.

- Implement a `push_nested_dict` method to push a values of nested dictionary onto the stack.

- Implement a `pop_nested_dict` method to pop the top element from the stack.

for example :
after implementing push_nested_dict() , the stack becomes:

{‘y’:100,’z’:200}

{‘x’:10}

{‘a’:’one,’b’:’two}

14 Write the definition of a function POP_PUSH (LPop, LPush, N) in Python. The function should Pop out the last N elements of the list LPop and Push them into the list LPush. For example:

If the contents of the list LPop are [10, 15, 20, 30]
And value of N passed is 2, then the function should create the list LPush as [30, 20] And the list LPop should now contain [10, 15]
NOTE: If the value of N is more than the number of elements present in LPop, then display the message "Pop not possible".

15 A list contains following record of customer:

[Customer_name, Room Type]
Write the following user defined functions to perform given operations on the stack named 'Hotel':
(i) Push Cust() - To Push customers' names of those customers who are staying in 'Delux' Room Type.
(ii) Pop Cust() - To Pop the names of customers from the stack and display them. Also, display "Underflow" when there are no customers in the stack.
For example:
If the lists with customer details are as follows:
[["Siddarth", "Delux"] ["Rahul", "Standard"] ["Jerry", "Delux"]]
The stack should contain
Jerry
Siddharth
The output should be:
Jerry
Siddharth
Underflow

16 Write a function in Python, Push (Vehicle) where, Vehicle is a dictionary containing details of vehicles (Car_Name: Maker). The function should push the name of car manufactured by TATA' (including all the possible cases like Tata, TaTa, etc.) to the stack. 3
For example:
If the dictionary contains the following data: Vehicle=("Santro": "Hyundai", "Nexon": "TATA", "Safari": "Tata"}

The stack should contain

Safari
Nexon