**Data structure And Algorithm (Amrit panta)**

**Data structure:**

Data structure are used to store data in an organized and efficient manner.

There are many data structure among them some are here after:

1.Arrays

2.Stack

3.Queue

4.Linked List

5.Tree

6.Graph

7.Hashing

Let’s try to learn all of them one by one with appropriate example and coding. Note in this course we are going to use python as a programming language. Hence, you must have the knowledge of python coding to learn this course.

**1.Array**

Basic Operation:

1.1 Traverse

1.2 Insertion

1.3 Deletion

1.4 Update

1.5 Search

Let’s learn each operation in details:

**1.1Traverse**

Traverse means printing all the elements of an array.

Below is the code in python to traverse an array.

# import array class from array module

from array import array

# creating array of integer value

my\_array=array('i', [2, 4, 6, 8, 10])

# for traverse we used loop to show all the elments of an array

print("Element of an array are listed here after::")

for i in range(len(my\_array)):

    print(f"my\_array[{i}]={my\_array[i]}")

output:

Element of an array are listed here after::

my\_array[0]=2

my\_array[1]=4

my\_array[2]=6

my\_array[3]=8

my\_array[4]=10

**1.2 Insertion**

Insertion operation means to insert one or more data elements into an array.

from array import array

# creating an empty array , in which elements will be enter by user

my\_array = array('i', [])

n = int(input("Enter the number of elements you want to enter:: "))

# taking input from the user and inserting into the array

for i in range(n):

    element = int(input(f"Enter {i+1} Element -> "))

    my\_array.append(element)

# showing all the element of an array after inserting

print("Element's in an array after insertion:")

print(my\_array)

output:

Enter the number of elements you want to enter:: 5

Enter 1 Element -> 5

Enter 2 Element -> 10

Enter 3 Element -> 15

Enter 4 Element -> 20

Enter 5 Element -> 25

Element's in an array after insertion:

array('i', [5, 10, 15, 20, 25])

**1.3 Deletion**

Deletion refers to removing an existing element from the array.

import array

my\_array = array.array('i', [1, 2, 3, 4, 5, 6, 7, 8, 9,11, 10])

print(f"Your array list \n {my\_array}  ")

num = int(input("Enter the number you want to delete \n ->"))

if num in my\_array:

    my\_array.remove(num)

    print(f"Array after the remove of {num}")

    print(my\_array)

else:

    print(f'{num} is not found in an array . Try with new number ')

output:

Your array list

array('i', [1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 10])

Enter the number you want to delete

->11

Array after the remove of 11

array('i', [1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

**1.4 Update**

Updating refers to updating an existing element from the array at a given index.

from array import array

my\_array = array('i', [1, 2, 3, 5, 6, 7, 9])

print(f"Your array List\n {my\_array}")

position = int(

    input("\nEnter the position on which you want to update :: \n -> "))

new\_element = int(input("Enter the new element you want to enter \n -> "))

# updating the new eelement at the given position

my\_array[position-1] = new\_element

print("\nArray list after updating")

print(my\_array)

output:

Your array List

array('i', [1, 2, 3, 5, 6, 7, 9])

Enter the position on which you want to update ::

-> 7

Enter the new element you want to enter

-> 8

Array list after updating

array('i', [1, 2, 3, 5, 6, 7, 8])

**1.5 Searching**

Searching is the process of finding the element in the existing array with the help of index.

import array

my\_array = array.array('i', [1, 3, 5, 7, 9, 1, 2, 4, 6, 8, 10])

search = int(input("Enter the number to be search :: "))

l = len(my\_array)

found = False

# searching number in an array

for i in range(l):

    if my\_array[i] == search:

        print(f"{search}  is found in {i+1} position  !!")

        found = True

# if number is not found then display the message

if found == False:

    print(f"{search } is not found in array")

Output:1

Enter the number to be search :: 4

4 is found in 8 position !!

Output:2

Enter the number to be search :: 1

1 is found in 1 position !!

1 is found in 6 position !!

**Python code for all operation of array in single program**

from array import array

import sys

from time import sleep

class Array:

    def \_\_init\_\_(*self*, *my\_array*):

*self*.my\_array = *my\_array*

    def insertelement(*self*, *n*):

        for i in range(*n*):

            num = int(input(f"Enter {i+1} element :: "))

*self*.my\_array.append(num)

        print(f'Congratulation {*n*} element are added into an array')

        sleep(1)

    def searchelement(*self*, *s*):

        found = False

        # searching number in an array

        for i in range(len(*self*.my\_array)):

            if my\_array[i] == *s*:

                print(f"{*s*}  is found in {i+1} position  !!")

                found = True

        # if number is not found then display the message

        if found == False:

            print(f"{*s* } is not found in array")

    def updatearray(*self*, *position*, *newelement*):

*self*.my\_array[*position*-1] = *newelement*

        print("\nArray list after updating")

        print('[', *end*='')

        for i in range(len(*self*.my\_array)):

            print(f'{*self*.my\_array[i]}', *end*=',')

        print(']')

    def deletearray(*self*):

        num = int(input("Enter the number you want to delete \n ->"))

        if num in *self*.my\_array:

*self*.my\_array.remove(num)

            print(f"Array after the remove of {num}")

            print('[', *end*='')

            for i in range(len(*self*.my\_array)):

                print(f'{*self*.my\_array[i]}', *end*=',')

            print(']')

        else:

            print(f'{num} is not found in an array . Try with new number ')

    def showarray(*self*):

        if len(*self*.my\_array) == 0:

            print(

                "Enter some element first in an array \n As there is no any element to Show at the moment")

        else:

            print("Element in an array:")

            print('[', *end*='')

            for i in range(len(*self*.my\_array)):

                print(f'{*self*.my\_array[i]}', *end*=',')

            print(']')

    @staticmethod

    def menu():

        print('''

        1.Enter element in array(insertion)

        2.Delete element from array

        3.Update an element from array

        4.Search an element from an array

        5.Show all element from an array

        0.Exist

        ''')

        option = int(input("Enter the option ::   "))

        return option

if \_\_name\_\_ == '\_\_main\_\_':

    # initializing an empty array

    print("\n\nAll Operation related to an Array")

    my\_array = array('i', [])

    a = Array(my\_array)

    while True:

        choice = a.menu()

        if choice == 1:

            # insertion of an array

            n = int(input("How many element  you want insert in an array::  "))

            a.insertelement(n)

        elif choice == 2:

            # deletion

            a.deletearray()

        elif choice == 3:

            # updating

            position = int(

                input("\nEnter the position on which you want to update :: \n -> "))

            new\_element = int(

                input("Enter the new element you want to enter \n -> "))

            a.updatearray(position, new\_element)

        elif choice == 4:

            # searching

            s = int(input("Enter an element  you want to search in an array::  "))

            a.searchelement(s)

        elif choice == 5:

            # traverse an array

            a.showarray()

        elif choice == 0:

            print("Thank you \n See you later")

            sleep(1)

            sys.exit(0)

        else:

            print("Enter the valid option mention above....")

output:

All Operation related to an Array

1.Enter element in array(insertion)

2.Delete element from array

3.Update an element from array

4.Search an element from an array

5.Show all element from an array

0.Exist

Enter the option :: 5

Enter some element first in an array

As there is no any element to Show at the moment

1.Enter element in array(insertion)

2.Delete element from array

3.Update an element from array

4.Search an element from an array

5.Show all element from an array

0.Exist

Enter the option :: 1

How many element you want insert in an array:: 5

Enter 1 element :: 100

Enter 2 element :: 110

Enter 3 element :: 120

Enter 4 element :: 130

Enter 5 element :: 99

Congratulation 5 element are added into an array

1.Enter element in array(insertion)

2.Delete element from array

3.Update an element from array

4.Search an element from an array

5.Show all element from an array

0.Exist

Enter the option :: 4

Enter an element you want to search in an array:: 105

105 is not found in array

1.Enter element in array(insertion)

2.Delete element from array

3.Update an element from array

4.Search an element from an array

5.Show all element from an array

0.Exist

Enter the option :: 4

Enter an element you want to search in an array:: 130

130 is found in 4 position !!

1.Enter element in array(insertion)

2.Delete element from array

3.Update an element from array

4.Search an element from an array

5.Show all element from an array

0.Exist

Enter the option :: 3

Enter the position on which you want to update ::

-> 5

Enter the new element you want to enter

-> 155

Array list after updating

[100,110,120,130,155,]

1.Enter element in array(insertion)

2.Delete element from array

3.Update an element from array

4.Search an element from an array

5.Show all element from an array

0.Exist

Enter the option :: 2

Enter the number you want to delete

->110

Array after the remove of 110

[100,120,130,155,]

1.Enter element in array(insertion)

2.Delete element from array

3.Update an element from array

4.Search an element from an array

5.Show all element from an array

0.Exist

Enter the option :: 0

Thank you

See you later

**Some important question related to array**

1**.**How do you remove duplicates from an array?

2.How do you reverse an array?

3.How is an array sorted?

4.How Do you find second largest number in unsorted integer array?

**2.Stack**

Basic Operation:

2.1: Push

2.2: Pop

2.3: Traverse

Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO (Last In First Out) or FILO (First In Last Out).

There are many real-life examples of a stack. Consider an example of plates stacked over one another in the canteen. The plate which is at the top is the first one to be removed, i.e. the plate which has been placed at the bottommost position remains in the stack for the longest period of time. So, it can be simply seen to follow LIFO (Last In First Out)/FILO (First In Last Out) order.

import sys

class Stack:

    def \_\_init\_\_(*self*):

*self*.stack = []

    def size(*self*):

        return len(*self*.stack)

    def isEmpty(*self*):

        if *self*.stack == []:

            return True

        else:

            return False

    def push(*self*):

        try:

            value = int(input("Enter the value you want to PUSH -> "))

*self*.stack.append(value)

            print(f'{value} is added to the stack....')

            print()

        except:

            print("Enter the number only !!! You have enter the other character..")

    def pop(*self*):

        if *self*.stack == []:

            print("No Element to POP , Stack is Empty at the moment")

        else:

            l = *self*.stack.pop()

            print(f'{l} value is pop from the stack')

    def stop(*self*):

        print("Thanks for the operation....... have an nice day ahead")

        sys.exit()

    def search(*self*, *f*):

        count = 0

        for i in range(len(*self*.stack)):

            if *self*.stack[i] == *f*:

                print(f'{*f*} found in {i+1} Position')

                count += 1

        if count == 0:

            print(f'{*f*} is not found in our stack')

    def update(*self*):

        #do in future1

        pass

    def show(*self*):

        if *self*.stack == []:

            print("Stack is Empty...\n So Enter Element First")

        else:

            print("Element in the stack are::::")

            # to make the latest enter element at the top we reverse the list

            for i in *self*.stack[::-1]:

                print(i)

        print()

    @staticmethod

    def menu():

        print("""

        Enter 1 for PUSH(C)

        Enter 2 for Traverse(R)

        Enter 3 for Update(U)

        Enter 4 for POP(D)

        Enter 5 for Search

        Enter 6 for size of stack

        Enter 7 for check empty or not

        Enter 0 for Exit

        """)

        try:

            choice = int(input("Enter your choice \n -> "))

            return choice

        except:

            print("Enter Number only ... You havee Enter the Other Character..")

            print()

s = Stack()

while True:

    print()

    print('\*\*\*\*\*\*\*\*\*\* Welcome to Stack Opeartion \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

    print()

    c = Stack.menu()

    if c == 1:

        s.push()

    elif c == 2:

        s.show()

    elif c == 3:

        s.update()

    elif c == 4:

        s.pop()

    elif c == 5:

        try:

            f = int(input("Enter the element you want to search... "))

            s.search(f)

        except ValueError:

            print("Enter integer value only for search")

    elif c == 6:

        print(s.size())

        print()

    elif c == 7:

        print(s.isEmpty())

        print()

    elif c == 0:

        s.stop()

    else:

        print("Invalid Option !!!! Please Enter the correct option")

        print()

Here we made a module and using the object we can use this class, and method in other code.

class Stack:

    def \_\_init\_\_(*self*):

*self*.stack = []

    def size(*self*):

        return len(*self*.stack)

    def isEmpty(*self*):

        if *self*.size() == 0:

            return True

        else:

            return False

    def push(*self*, *value*):

*self*.stack.append(*value*)

    def pop(*self*):

        if *self*.isEmpty():

            return print("Stack is empty")

        else:

            return *self*.stack.pop()

    def traverse(*self*):

        if *self*.stack == []:

            return print("Stack is empty")

        else:

            return *self*.stack[::-1]

with the use of above module, we can use the function and class as in example below:

# this is user define module

from stack2 import Stack

from time import sleep

s = Stack()

while True:

    print("""

    enter 1 for push

    enter 2 for pop

    enter 3 traverse

    enter 4 size

    enter 5 for isEmpty

    enter 0 for exit

    """)

    c = int(input("Enter your choice ->"))

    if c == 1:

        value = int(input("Enter the value you want to enter ->"))

        s.push(value)

        sleep(1)

    elif c == 2:

        element = s.pop()

        print(f'{element} is pop from the stack')

        sleep(1)

    elif c == 3:

*list* = s.traverse()

        for i in *list*:

            print(i)

        sleep(1)

    elif c == 4:

        print(s.size())

        sleep(1)

    elif c == 5:

        print(s.isEmpty())

        sleep(1)

    elif c == 0:

        print("thanks for the operation")

        exit()

    else:

        print("Enter the valid option mention above")

        sleep(1)

**some application of stack in real world are below:**

note: In the below example we use the module mention above

**1.Reversing string or and number**

# importing user define module

from stack2 import Stack

s = Stack()

print("enter the character one by on\n")

print("Enter 0 to Exit\n")

while True:

    c = input()

    if c == 0 or c == '0':

        break

    s.push(c)

print(" After Reversing")

reverse = ''

while s.size() != 0:

    reverse = reverse+str(s.pop())

print()

print(reverse)

**2.Decimal to Binary Number**

from stack2 import Stack

s = Stack()

decimal\_number = int(input("Enter the deciaml number ->"))

while decimal\_number > 0:

    num = decimal\_number % 2

    s.push(num)

    decimal\_number = decimal\_number//2

binary\_number = ''

print(binary\_number)

while not s.isEmpty():

    binary\_number = binary\_number+str(s.pop())

print(binary\_number)

**3. Postfix , prefix and infix expression**

**4.Checking the parenthesis**

# this is user define module

from stack2 import Stack

def parChecker(*symbolString*):

    s = Stack()

    balanced = True

    index = 0

    while index < len(*symbolString*) and balanced:

        symbol = *symbolString*[index]

        if symbol == "(":

            s.push(symbol)

        else:

            if s.isEmpty():

                balanced = False

            else:

                s.pop()

        index = index + 1

    if balanced and s.isEmpty():

        return True

    else:

        return False

print(parChecker('((()))'))

print(parChecker('(()'))

print(parChecker('()'))

**Algorithms:**

Algorithms are the procedure to solve any problem in computer science.

There are many algorithms which are used for solve problems among them some are discuss here after.

1.Sorting

**2.Searching**

Searching Algorithms are designed to check for an element or retrieve an element from any data structure where it is stored. Based on the type of search operation, these algorithms are generally classified into two categories:

1. **Sequential Search**: In this, the list or array is traversed sequentially and every element is checked. For example: Linear Search.
2. **Interval Search**: These algorithms are specifically designed for searching in sorted data-structures. These type of searching algorithms are much more efficient than Linear Search as they repeatedly target the center of the search structure and divide the search space in half.

For Example: Binary Search.

**1.Linear Search**

print("linear searching....")

def linearSearch(*list*, *element*):

    count = 0

    for i in range(len(*list*)):

        if *list*[i] == *element*:

            print(f"{*element*} is found in {i+1} position")

            count += 1

    if count == 0:

        print(f'{*element*} is not found ...')

*list* = []

n = int(input("\n\nHow many element do you want to enter in list ?"))

print(f"Enter {n} element:")

for i in range(n):

    num = int(input(f"Enter {i+1} element :=> "))

*list*.append(num)

print()

print()

element = int(input("Enter the element you want to search \n-> "))

linearSearch(*list*, element)

output:

linear searching....

How many element do you want to enter in list ?5

Enter 5 element:

Enter 1 element :=> 7

Enter 2 element :=> 5

Enter 3 element :=> 23

Enter 4 element :=> 8

Enter 5 element :=> 9

Enter the element you want to search

-> 8

8 is found in 4 position

**2.Binary Search**