**Name : jadav savan**

**Roll No : 31**

**PRN : 2017095900001955**

**Sub : Python Programming ☺**

**Sem : 7th**

**Branch : Computer Engineering**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

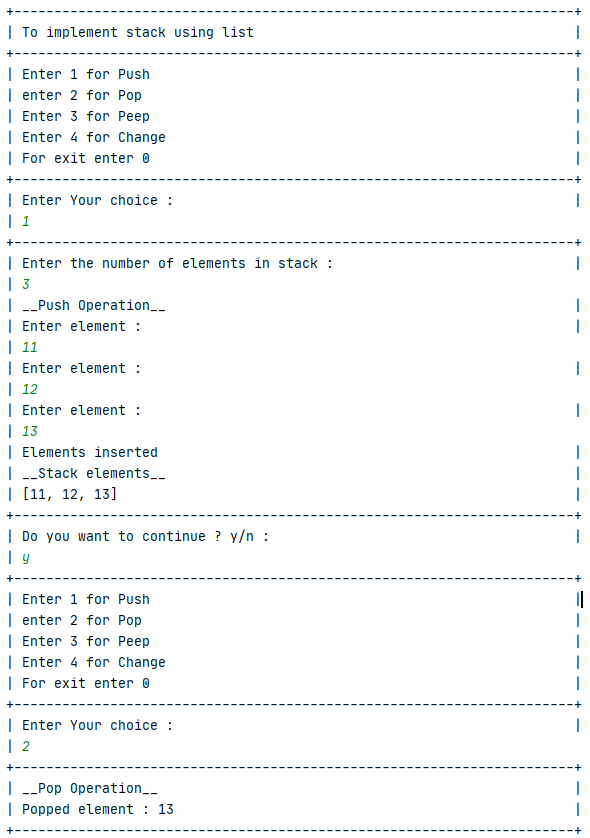
**Practical 6**

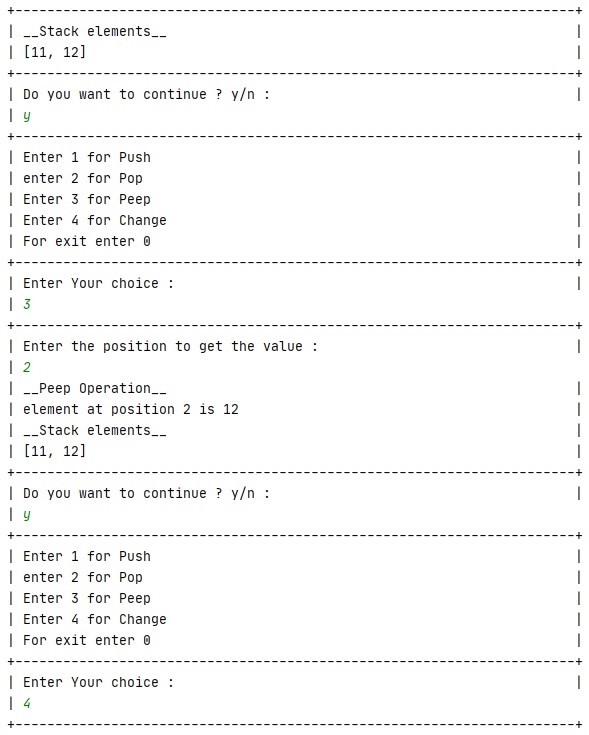
**(a)To implement stack using list**

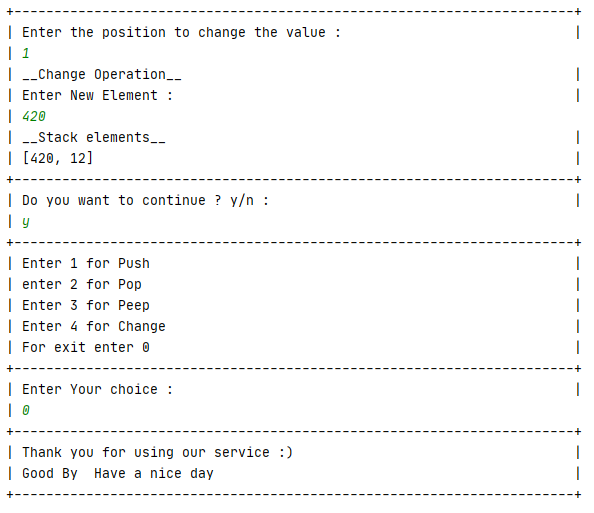
**Solution :**

from format import \*  
  
obj = Line\_maker()  
obj.start\_Line()  
  
obj.string\_Line(string=**"To implement stack using list"**)  
  
stack = []  
  
def push(n):  
 obj.string\_Line(string=**"\_\_Push Operation\_\_"**)  
 for i in range(n):  
 obj.string\_Line(string=**"Enter element : "**)  
 print(**"| "**, end=**""**)  
 element = int(input())  
 stack.append(element)  
 obj.string\_Line(string=**"Elements inserted"**)  
  
def pop():  
 if(len(stack) != 0):  
 obj.string\_Line(string=**"\_\_Pop Operation\_\_"**)  
 obj.string\_Line(string=**"Popped element : "**+str(stack.pop()))  
 obj.start\_Line()  
 else:  
 obj.string\_Line(string=**"stack is empty"**)  
  
def peep(i):  
 obj.string\_Line(string=**"\_\_Peep Operation\_\_"**)  
 obj.string\_Line(string=**"element at position "**+str(i)+**" is "**+str(stack[i - 1]))  
  
def change(i):  
 obj.string\_Line(string=**"\_\_Change Operation\_\_"**)  
 obj.string\_Line(string=**"Enter New Element : "**)  
 print(**"| "**, end=**""**)  
 stack[i - 1] = int(input())  
  
def display():  
 obj.string\_Line(string=**"\_\_Stack elements\_\_"**)  
 obj.string\_Line(string=str(stack))  
 obj.start\_Line()  
  
flag = **'y'**while flag == **'y'**:  
 obj.start\_Line()  
 obj.string\_Line(string=**"Enter 1 for Push"**)  
 obj.string\_Line(string=**"enter 2 for Pop"**)  
 obj.string\_Line(string=**"Enter 3 for Peep"**)  
 obj.string\_Line(string=**"Enter 4 for Change"**)  
 obj.string\_Line(string=**"For exit enter 0"**)  
 obj.start\_Line()  
  
 try:  
 obj.string\_Line(string=**"Enter Your choice : "**)  
 print(**"| "**, end=**""**)  
 choice = int(input())  
 obj.start\_Line()  
 except ValueError:  
 obj.string\_Line(string=**"Please enter integer values"**)  
 obj.start\_Line()  
  
 if choice == 1:  
 obj.string\_Line(string=**"Enter the number of elements in stack : "**)  
 print(**"| "**, end=**""**)  
 n = int(input())  
 push(n)  
  
 elif choice == 2:  
 pop()  
  
 elif choice == 3:  
 obj.string\_Line(string=**"Enter the position to get the value : "**)  
 print(**"| "**, end=**""**)  
 i = int(input())  
 peep(i)  
 elif choice == 4:  
 obj.string\_Line(string=**"Enter the position to change the value : "**)  
 print(**"| "**, end=**""**)  
 i = int(input())  
 change(i)  
 elif choice == 0:  
 obj.string\_Line(string=**"Thank you for using our service :)"**)  
 obj.string\_Line(string=**"Good By Have a nice day "**)  
 flag = **''** else:  
 obj.string\_Line(string=**"Enter right choice please"**)  
  
 if(choice != 0):  
 display()  
 obj.string\_Line(string=**"Do you want to continue ? y/n : "**)  
 print(**"| "**, end=**""**)  
 flag = input()  
obj.start\_Line()

**output:**





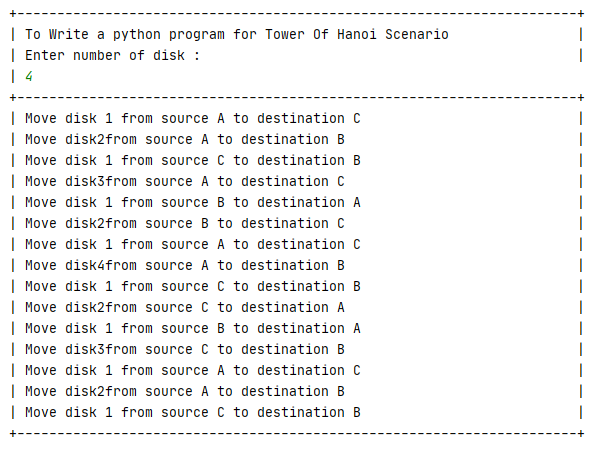


**a python program for Tower Of Hanoi Scenario**

**Solution :**

from format import \*  
  
obj = Line\_maker()  
obj.start\_Line()  
  
obj.string\_Line(string=**"To Write a python program for Tower Of Hanoi Scenario"**)  
  
def TowerOfHanoi(n, s, d, i):  
 if n == 1:  
 obj.string\_Line(string=**"Move disk 1 from source "**+str(s)+**" to destination "**+str(d))  
 return  
 TowerOfHanoi(n-1, s, i, d)  
 obj.string\_Line(string=**"Move disk"**+str(n)+**"from source "**+str(s)+**" to destination "**+str(d))  
 TowerOfHanoi(n-1, i, d, s)  
  
  
obj.string\_Line(string=**"Enter number of disk : "**)  
print(**"| "**, end=**""**)  
n =int(input())  
obj.start\_Line()  
TowerOfHanoi(n, **'A'**, **'B'**, **'C'**)  
obj.start\_Line()

**Output :**

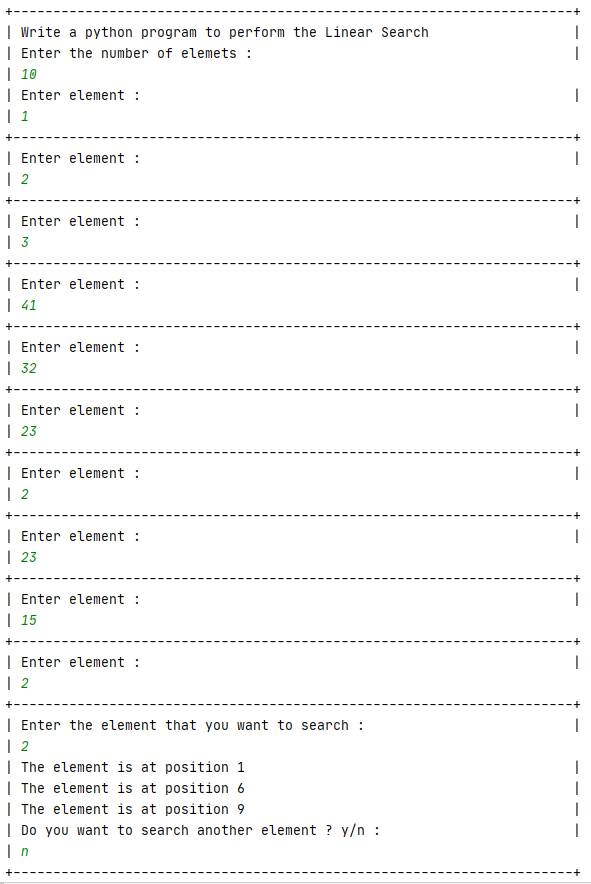
****

**(c) Write a python program to perform the Linear Search**

**Solution** :

from format import \*  
  
obj = Line\_maker()  
obj.start\_Line()  
  
obj.string\_Line(string=**"Write a python program to perform the Linear Search"**)  
list1 = []  
def Linear\_Search(n, e):   
 for i in range(n):  
 if list1[i] == e:  
 obj.string\_Line(string=**"The element is at position "**+str(i))  
 return  
obj.string\_Line(string=**"Enter the number of elemets : "**)  
print(**"| "**, end=**""**)  
n = int(input())  
for i in range(n):  
 obj.string\_Line(string=**"Enter element : "**)  
 print(**"| "**, end=**""**)  
 list1.append(int(input()))  
 obj.start\_Line()  
flag = **'y'**while flag == **'y'**:  
 obj.string\_Line(string=**"Enter the element that you want to search : "**)  
 print(**"| "**, end=**""**)  
 e = int(input())  
 Linear\_Search(n, e)  
 obj.string\_Line(string=**"Do you want to search another element ? y/n : "**)  
 print(**"| "**, end=**""**)  
 flag = input()  
obj.start\_Line()

**Output** :

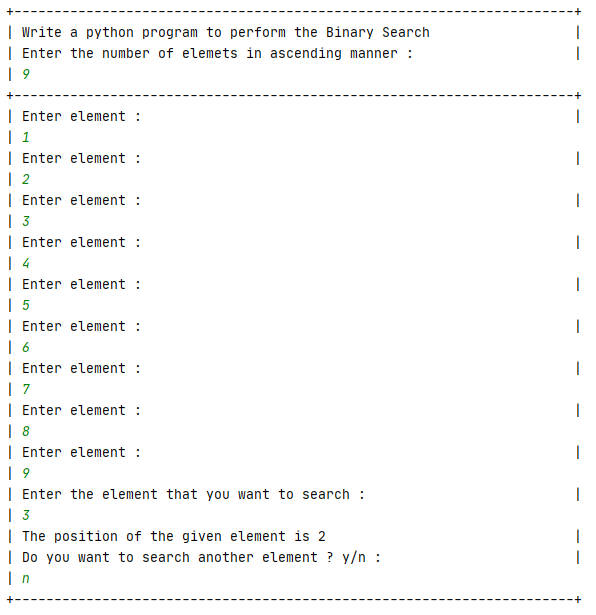


**(d) Write a python program to perform the Binary Search**

**Solution :**

from format import \*  
  
obj = Line\_maker()  
obj.start\_Line()  
  
obj.string\_Line(string=**"Write a python program to perform the Binary Search"**)  
  
list1 = []  
def Binary\_Search(low, high, e):  
 if high >= low:   
 middle = (high + low) // 2  
 if list1[middle] == e:  
 return middle   
 elif list1[middle] > e:  
 return Binary\_Search(low, middle-1, e)   
 else:  
 return Binary\_Search(middle+1, high, e)  
 else:  
 return -1  
  
obj.string\_Line(string=**"Enter the number of elemets in ascending manner : "**)  
print(**"| "**, end=**""**)  
n = int(input())  
obj.start\_Line()  
for i in range(n):  
 obj.string\_Line(string=**"Enter element : "**)  
 print(**"| "**, end=**""**)  
 list1.append(int(input()))  
flag = **'y'**while flag == **'y'**:  
 obj.string\_Line(string=**"Enter the element that you want to search : "**)  
 print(**"| "**, end=**""**)  
 e = int(input())  
 low = 0  
 high = n-1  
 result = Binary\_Search(low, high, e)  
 if result != -1:  
 obj.string\_Line(string=**"The position of the given element is "**+str(result))  
 else:  
 obj.string\_Line(string=**"Element is not present "**)  
 obj.string\_Line(string=**"Do you want to search another element ? y/n : "**)  
 print(**"| "**, end=**""**)  
 flag = input()  
obj.start\_Line()

**Output :**

****