Roll no:1601-20-733-167
M.Sai Teja

Week -7

Q1

AIM: A program to implement stack using array and its operations

DESCRIPTION:

1. Define a class Stack with initial attributes of array, size and top and assign top value to be -1
2. Now define a push method in Stack class. Check whether top value is equal to (size-1) value, if it is true, then print “Overflow”, else input a value, store it in the array and increment the top value
3. Similarly define a pop method to remove the top value and decrement the top value by 1. If the top value is -1, then print “Underflow”
4. Also define a display method to print the values of the array stored in the stack method.
5. Now outside the class, input the size of the array and define a object of Stack class with the size of array given. Perform the push, pop and display operations with the keys given to them.

PROGRAM:

```
class Stack:
    def __init__(self,size):
        self.s=[0]*(size)
        self.size=size
        self.top=-1
    def push(self):
        if self.top == self.size-1:
            print("Overflow")
        else:
            self.top+=1
            key=int(input("Enter your value: "))
            self.s[self.top]=key
```

```
    def pop(self):
        if self.top == -1:
            print("stack is empty")
        else:
            self.top-=1
    def show(self):
        if self.top == -1:
            print("Stack is empty")
        else:
            temp=self.top
            while temp!=-1:
                print(self.s[temp],end=' ')
                temp-=1

n=int(input("Enter the no of elements in the stack: "))
print("\t*****Stack operation using array*****\t")
s=Stack(n)
while True:
    choice=int(input("\n1.Push\t2.Pop\t3.Show\t4.Exit\nEnter your choice: "))
    if choice == 1:
        s.push()
    elif choice == 2:
        s.pop()
    elif choice == 3:
        s.show()
    elif choice == 4:
        break
    else:
```

print("Wrong choice, try again")

TIME COMPLEXITY:

1. For insertion: O(1)
2. For deletion: O(1)
3. For display: O(n)

OUTPUT:

```
Enter the no of elements in the stack: 5
        *****Stack operation using array*****

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 1

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 2

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
2 1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 3

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 4

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 5

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
5 4 3 2 1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Overflow
```

```
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 2

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
4 3 2 1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 2

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
3 2 1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 2

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
2 1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 2

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 2

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
Stack is empty

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 2
stack is empty

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 4
```

CONCLUSION: The code is error free and it runs as expected.

Q2

AIM: A program to implement stack using linked list and its operations

DESCRIPTION:

1. Define a Node class with initial attributes data and next. Assign data value to given value and next to Null value
2. Now define a Stack class with initial head attribute and assign it to Null value
3. Define a push method in stack class and input the value to be stored. Create a Node object and store the value in it. Check if the head is null, if it is, then assign head to new node object created. Else assign next attribute of new node to head variable, and make the new node as head.
4. Similarly define a pop method, where it will remove a node from the stack. First check if stack is empty, then print “Stack is empty” else assign head variable to next node of the linked list.
5. Define a show method to print the stack.
6. Now outside the class, create a stack object and perform its operations for push, pop and showing the stack.

PROGRAM:

```
class Node:
    def __init__(self,data):
        self.data=data
        self.next=None
class Stack:
    def __init__(self):
        self.head=None
    def push(self):
        value=int(input("Enter your value: "))
        newnode=Node(value)
        if self.head is None:
            self.head=newnode
```

```
        else:
            newnode.next=self.head
            self.head=newnode
    def pop(self):
        if self.head is None:
            print("stack is empty")
        else:
            self.head=self.head.next
    def show(self):
        ptr=self.head
        if self.head is None:
            print("stack is empty")
        else:
            while ptr!=None:
                print(ptr.data,end=' ')
                ptr=ptr.next
print("\t*****Stack operation using linked list*****\t")
s=Stack()
while True:
    choice=int(input("\n1.Push\t2.Pop\t3.Show\t4.Exit\nEnter your choice: "))
    if choice == 1:
        s.push()
    elif choice == 2:
        s.pop()
    elif choice == 3:
        s.show()
    elif choice == 4:
        break
```

```
else:
    print("Wrong choice, try again")
```

TIME COMPLEXITY:

1. For insertion: O(1)
2. For deletion: O(1)
3. For display: O(n)

OUTPUT:

```
        *****Stack operation using linked list*****

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 2
stack is empty

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
stack is empty

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 1

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 2

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 3

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 4

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
4 3 2 1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 2

1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
3 2 1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 1
Enter your value: 5
```

```
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 3
5 3 2 1
1.Push  2.Pop   3.Show  4.Exit
Enter your choice: 4
```

CONCLUSION: The code is error free and it runs as expected.