**PRACTICAL 8**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SUBJECT CODE:** | CE259 | **SEMESTER:** | 4 | **ACADEMIC YEAR:** | | 2021-22 |
| **SUBJECT:** | PROGRAMMING IN PYTHON | | | | | |
| **NAME:** | JIYA SHAH | | **ROLL NO:** | | 20CS083 | |

|  |  |
| --- | --- |
| **Aim** | **Write a Program in Python to implement a Stack Data Structure using Class and Objects, with push, pop, and traversal method** |
| **Code** | class Stack(object):  def \_\_init\_\_(self, size):  self.index = []  self.size = size  def push(self, data):  # Pushes the element at the top  if (self.isFull() != True):  self.index.append(data)  print("Pushed")  else:  print("Stack overflowed")  def pop(self):  # Popping the top element  if (self.isEmpty() != True):  print("Popped")  return self.index.pop()  else:  print("Stack is already empty")  def isEmpty(self):  # Checks whether the stack is empty  return len(self.index) == []  def isFull(self):  # Checks whether the stack if full  return len(self.index) == self.size  def stackSize(self):  # Returns the current stack size  return len(self.index)  def \_\_str\_\_(self):  myString = ' '.join(str(i) for i in self.index)  return myString  def traversal(self):  print("\nTraversal: ",self,"\n")  if \_\_name\_\_ == '\_\_main\_\_':  st = Stack(10)  st.push(1)  st.push(2)  st.push(3)  st.push(5)  st.push(4)  st.push(10)  st.push(6)  st.push(9)  st.traversal()  st.pop()  st.pop()  st.pop()  st.traversal() |
| **Output** |  |