

2021

**Lab Report# 01**

**Submitted by: MUHAMMAD USMAN.**

**Registration # 200901104.**

**Submitted to: Mr. Nadeem Yousaf.**

**Subject: Data Structure Lab.**

**DATED:24/10/2021**

**Lab task # 01:**

**Write a program to let the user enter a string of his own choice. Check whether the given string is a palindrome or not.**

**Code:**

from collections import deque  
  
reverse\_string = ''  
stack3 = deque() # create stack  
string = input("enter the string: ") # take input  
for i in range(len(string)):  
 stack3.append(string[i]) # append the input one by one in stack  
  
while not (len(stack3) == 0):  
 reverse\_string += stack3.pop() # pop the output one by one and store in a variable until stack is empty  
  
if reverse\_string == string: # compare the original sting with reversed string  
 print("entered string is palindrome") # print the output  
else:  
 print("entered string is not palindrome")

**Output:**

****

****

****

****

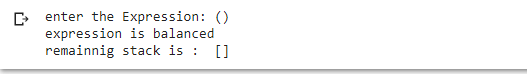
**Lab task # 02:**

**Write a program to check the balanced parenthesis in the expression or not using stack**

**Code:**

from collections import deque # import deque  
  
stack3 = deque() # create stack  
  
  
def Balanced(Expression): # function body  
 for paren in Expression:  
 if paren == '(' or paren == '{' or paren == '[': # conditions for input  
 stack3.append(paren)  
 if paren == ')' or paren == '}' or paren == ']':  
 if len(stack3) == 0: # condition to check empty stack  
 return False  
 top = stack3.pop() # compare the parenthesis with entered parenthesis  
 if (top == '(' and paren != ')') or (top == '{' and paren != '}' or (top == '[' and paren != ']')):  
 return False  
 return not stack3 # return the stack if it is empty  
  
  
if \_\_name\_\_ == '\_\_main\_\_': # main body  
 Expression = input("enter the Expression: ") # take input  
 if Balanced(Expression): # call function  
 print("expression is balanced") # display output  
 else:  
 print(" expression is not balanced")

**Output:**

****

**End!!!**