[1] Write a menu-driven python program to implement stack operation.

```
def check_stack_isEmpty(stk):
    if stk==[]:
        return True
    else:
        return False

def main_menu():
    while True:
      print("Stack Implementation")
      print("1 - Push")
      print("2 - Pop")
      print("3 - Peek")
      print("4 - Display")
      print("5 - Exit")
      ch = int(input("Enter the your choice:"))
      if ch==1:
          el = int(input("Enter the value to push an element:"))
          push(s,el)
      elif ch==2:
          e=pop_stack(s)
          if e=="UnderFlow":
              print("Stack is underflow!")
          else:
              print("Element popped:",e)
      elif ch==3:
          e=pop_stack(s)
          if e=="UnderFlow":
              print("Stack is underflow!")
          else:
              print("The element on top is:",e)
      elif ch==4:
          display(s)
      elif ch==5:
          break
      else:
          print("Sorry, You have entered invalid option")
def push(stk,e):
    stk.append(e)
    top = len(stk)-1
def display(stk):
    if check_stack_isEmpty(stk):
        print("Stack is Empty")
    else:
        top = len(stk)-1
        print(stk[top],"-Top")
        for i in range(top-1,-1,-1):
            print(stk[i])
def pop_stack(stk):
    if check_stack_isEmpty(stk):
        return "UnderFlow"
    else:
        e = stk.pop()
        if len(stk)==0:
            top = None
        else:
            top = len(stk)-1
        return e
def peek(stk):
    if check_stack_isEmpty(stk):
        return "UnderFlow"
    else:
        top = len(stk)-1
        return stk[top]

s=[]
top = None
main_menu()
```

[2] Write a program to implement a stack for the employee details (empno, name).

```
stk=[]
top=-1
def line():
  print('~'*100)

def isEmpty():
  global stk
  if stk==[]:
    print("Stack is empty!!!")
  else:
    None

def push():
  global stk
  global top
  empno=int(input("Enter the employee number to push:"))
  ename=input("Enter the employee name to push:")
  stk.append([empno,ename])
  top=len(stk)-1

def display():
  global stk
  global top
  if top==-1:
    isEmpty()
  else:
    top=len(stk)-1
    print(stk[top],"<-top")
    for i in range(top-1,-1,-1):
       print(stk[i])

def pop_ele():
  global stk
  global top
  if top==-1:
    isEmpty()
  else:
    stk.pop()
    top=top-1

def main():
  while True:
    line()
    print("1. Push")
    print("2. Pop")
    print("3. Display")
    print("4. Exit")
    ch=int(input("Enter your choice:"))
    if ch==1:nm
      push()
      print("Element Pushed")
    elif ch==2:
      pop_ele()
    elif ch==3:
      display()
    else:
      print("Invalid Choice")

main()
```

[3] Write a python program to check whether a string is a palindrome or not using stack.

```
stack = []
top = -1

# push function
def push(ele):
        global top
        top += 1
        stack[top] = ele

# pop function
def pop():
        global top
        ele = stack[top]
        top -= 1
        return ele
```

```
# Function that returns 1 if string is a palindrome
def isPalindrome(string):
        global stack
        length = len(string)

        # Allocating the memory for the stack
        stack = ['0'] * (length + 1)

        # Finding the mid
        mid = length // 2
        i = 0
        while i < mid:
                push(string[i])
                i += 1

        # Checking if the length of the string is odd, if odd then neglect the middle character
        if length % 2 != 0:
                i += 1

        # While not the end of the string
        while i < length:
                ele = pop()

                # If the characters differ then the given string is not a palindrome
                if ele != string[i]:
                        return False
                i += 1
        return True
string = input("Enter string to check:")

if isPalindrome(string):
      print("Yes, the string is a palindrome")
else:
      print("No, the string is not a palindrome")
```