SUPERIOR UNIVERSITY

Data Structure and Algorithm (Lab)

Assignment – 2

Name:

Ali Maqsood.

Roll no:

SU92-BSAIM-F23-050.

Department:

Software Engineering Department.

Program:

Artificial Intelligence.

Section:

BSAI-3A

Question # 1:

Objective: Write a Python program to reverse a given string using a stack. Define custom methods for push and pop, and process each character of the string using these operations. The program should take a string as input and return the reversed string as output, demonstrating how stacks work with string manipulation.

Code:

```
class task1:
    def __init__(self):
        self.stack1=[]
    def push(self):
        self.input1=input("Enter the string: ")
    def pop(self):
        if len(self.stack1)==0:
            print("Stack is empty")
        else:
            return self.stack1.pop()
    def reverse(self):
        self.push()
        for i in self.input1:
            self.stack1.append(i)
        reversed_string=""
        for i in (self.input1):
            reversed_string+=self.pop()
        print("Reversed string is: ",reversed_string)
obj1=task1()
obj1.reverse()
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 2>python task2.py
Enter the string: hello world
Reversed string is:  dlrow olleh
```

Question # 2:

Objective: Create a Python program that verifies if an input string has balanced parentheses. Use a stack to push opening brackets ((, {, [) and pop them when encountering their corresponding closing brackets (), },]). The program should return True if the parentheses in the string are balanced and False otherwise.

Code:

```
class task2:
    def __init__(self):
        self.stack2=[]
    def checking(self,input1):
        for i in input1:
            if i=='(' or i=='{' or i=='[':
                self.stack2.append(i)
            elif i==')' and self.stack2 and self.stack2[-1]=='(':
                self.stack2.pop()
            elif i=='}' and self.stack2 and self.stack2[-1]=='{':
                self.stack2.pop()
            elif i==']' and self.stack2 and self.stack2[-1]=='[':
                self.stack2.pop()
            else:
                return False
        return len(self.stack2)==0
obj2=task2()
input1=input("Enter a string with parentheses: ")
if obj2.checking(input1):
    print("The parentheses are balanced.")
else:
```

```
    print("The parentheses are not balanced.")
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 2>python task2.py
Enter a string with parentheses: {[()]}
The parentheses are balanced.
```

Question # 3:

Objective: Implement a queue to simulate a ticketing system where customers join the queue to get served. Write a Python program that allows customers to enqueue when they arrive and dequeue when served. The program should print the order of service for customers and demonstrate the use of queue operations.

Code:

```
class task3:
    def __init__(self):
        self.queue1=[]
    def in_queue(self):
        input_name=input("Enter the name: ")
        self.queue1.append(input_name)
    def de_queue(self):
        if len(self.queue1)==0:
            print("Queue is empty")
        else:
            print(f"{self.queue1.pop(0)} is served.")
    def display(self):
        if len(self.queue1)==0:
            print("Queue is empty")
        else:
            print("Current Customers in Queue and their turn number: \n")
            for i in range(len(self.queue1)):
                print(f"{i+1}) {self.queue1[i]}")
    def menu(self):
        print("Ticketing System Menu:\n")
        print("1. Add Customer to Queue")
        print("2. Serve Customer")
```

```
        print("3. Display Queue")
        print("4. Exit")
        choice=int(input("Enter your choice: "))
        if choice==1:
            self.in_queue()
            self.menu()
        elif choice==2:
            self.de_queue()
            self.menu()
        elif choice==3:
            self.display()
            self.menu()
        elif choice==4:
            print("Thank you for using our service.")
            exit()
        else:
            print("Invalid choice. Please try again.")
            self.menu()
obj2=task3()
obj2.menu()
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 2>python task2.py
Ticketing System Menu:

1. Add Customer to Queue
2. Serve Customer
3. Display Queue
4. Exit
Enter your choice: 1
Enter the name: ali
Ticketing System Menu:

1. Add Customer to Queue
2. Serve Customer
3. Display Queue
4. Exit
Enter your choice: 3
Current Customers in Queue and their turn number:

1) ali
Ticketing System Menu:

1. Add Customer to Queue
2. Serve Customer
3. Display Queue
4. Exit
Enter your choice: 2
ali is served.
```

Question # 4:

Objective: Write a Python program to reverse the elements of a queue using a stack. Implement both queue and stack operations (enqueue, dequeue, push, pop) and demonstrate the reversal process by displaying the queue before and after the reversal.

Code:

```
class task4_1:
    def __init__(self):
        self.queue4=[]
    def in_queue(self,input1):
        self.queue4.append(input1)
    def de_queue(self):
        if len(self.queue4)==0:
            print("Queue is empty")
        else:
            return self.queue4.pop(0)
class task4_2:
    def __init__(self):
        self.stack4=[]
    def push(self,input1):
        self.stack4.append(input1)
    def pop(self):
        if len(self.stack4)==0:
            print("Stack is empty")
        else:
            return self.stack4.pop()
def reverse_queue(queue):
    stack=task4_2()
    while len(queue.queue4) > 0:
```

```
        stack.push(queue.de_queue())
    while len(stack.stack4) > 0:
        queue.in_queue(stack.pop())
obj1=task4_1()
obj1.in_queue(1)
obj1.in_queue(2)
obj1.in_queue(3)
obj1.in_queue(4)
print(f"Original Queue is: {obj1.queue4}")
reverse_queue(obj1)
print(f"Reversed Queue is: {obj1.queue4}")
```

Output:

```
E:\Uni\3rd Semester\4) Data Structures & Algorithms (Lab)\Assignments\Assignment 2>python task2.py
Original Queue is: ['A', 'B', 'C', 'D']
Reversed Queue is: ['D', 'C', 'B', 'A']
```