Assignment - 01

1. Write a program to sort a stack using a temporary stack. Build a stack as input from the user.
   input: [34, 3, 31, 98, 92, 23]

```
Element taken out: 23
input: [34, 3, 31, 98, 92]
tmpStack: [23]
------------------------------------------------
Element taken out: 92
input: [34, 3, 31, 98]
tmpStack: [23, 92]
------------------------------------------------
Element taken out: 98
input: [34, 3, 31]
tmpStack: [23, 92, 98]
------------------------------------------------
Element taken out: 31
input: [34, 3, 98, 92]
tmpStack: [23, 31]
------------------------------------------------
Element taken out: 92
input: [34, 3, 98]
tmpStack: [23, 31, 92]
------------------------------------------------
Element taken out: 98
input: [34, 3]
tmpStack: [23, 31, 92, 98]
------------------------------------------------
Element taken out: 3
input: [34, 98, 92, 31, 23]
tmpStack: [3]
------------------------------------------------
Element taken out: 23
input: [34, 98, 92, 31]
tmpStack: [3, 23]
------------------------------------------------
Element taken out: 31
input: [34, 98, 92]
tmpStack: [3, 23, 31]
------------------------------------------------
Element taken out: 92
input: [34, 98]
tmpStack: [3, 23, 31, 92]
------------------------------------------------
Element taken out: 98
input: [34]
tmpStack: [3, 23, 31, 92, 98]
------------------------------------------------
Element taken out: 34
input: [98, 92]
tmpStack: [3, 23, 31, 34]
------------------------------------------------
Element taken out: 92
input: [98]
tmpStack: [3, 23, 31, 34, 92]
------------------------------------------------
Element taken out: 98
input: []
tmpStack: [3, 23, 31, 34, 92, 98]
------------------------------------------------
```

final sorted list: [3, 23, 31, 34, 92, 98]

2. Write a program to print the characters of the string in sorted order using stack.

   Input: str = "hello3569world12478"
   Output: 123456789dehllloorw

   Algorithm:
   - Initialize two stacks, one main stack and other auxiliary stack.
   - Insert the first character of the string in the main stack.
   - Iterate for all the characters in the string one by one.
   - if the current character (nth character) is greater than or equal to the top element of the main stack, then push the element.
   - if the current character is not greater, then push all the greater elements of the main stack into the auxiliary stack, and then push the character into the main stack. After this, push all the greater elements of the auxiliary stack to the main stack.
   - Print all elements of the stack in reverse order when the iteration is completed.

3. Write a program to reverse a stack using recursion. (You are not allowed to use loop constructs like while, for..etc,). Use the function mentioned below.

   - `isEmpty(S)`
   - `push(S)`
   - `pop(S)`

4. Write a program to sort a stack using recursion (Use of any loop constructs like while, for etc. is not allowed.)

   ```
   Output:
   Stack elements before sorting:
   -3   14   18   -5   30

   Stack elements after sorting:
   30   18   14   -3   -5
   ```

5. Write a C program to Implement Queue Using stack

6. Write a C program to create a queue where insertion and deletion of elements can be done from both the ends.