Implement Stack using Arrays

04 April 2025 10:58

```
private int[] arr;
private int top;

public MyStack() {
arr = new int[1000];
top = -1;
}

public void push(int x)
{
if(top==arr.length)
{
return;
}
top++;
arr[top]=x;
// Your Code
}

public int pop() {
if(top==-1)
{
return -1;
}
int a=arr[top];
top--;
return a;
}
```

Implement Queue using Arrays

04 April 2025 11:11

```
class MyQueue {

   int front, rear;
       int arr[] = new int[100005];

   MyQueue()
       {
              front=0;
              rear=0;
       }

       //Function to push an element x in a queue.
       void push(int x)
       {
          if(front>rear||rear>=arr.length-1)
          {
             return;
          }

          arr[rear]=x;
          rear++;
          return;
       }

   //Function to pop an element from queue and return that element.
       int pop()
       {
          if(front==rear)
          {
             front=0;
             rear=0;
             return -1;
          }
          int a=arr[front];
          front++;
          return a;
       }
}
```

Implement Stack using Queue

04 April 2025 11:16

```
class MyStack {
    Queue<Integer>q1;
    Queue<Integer>q2;
    public MyStack() {
        q1=new LinkedList<>();
        q2=new LinkedList<>();
    }

    public void push(int x)
    {
        if(q1.isEmpty())
        {
            q1.add(x);
        }
        else
        {
            while(!q1.isEmpty())
            {
                q2.add(q1.poll());
            }
            q1.add(x);
            while(!q2.isEmpty())
            {
                q1.add(q2.poll());
            }
        }

    }

    public int pop() {
        if(q1.size()==0)
        {
            return -1;
        }
        return q1.poll();
    }

    public int top() {
        if(q1.size()==0)
        {
            return -1;
        }
        return q1.peek();
    }

    public boolean empty() {
        return q1.size()==0;
    }
}
```

Implement Queue using Stack

04 April 2025 11:59

```
class MyQueue {
    Stack<Integer>s1;
    Stack<Integer>s2;
    public MyQueue() {
        s1=new Stack<Integer>();
        s2=new Stack<Integer>();

    }

    public void push(int x) {
        s1.push(x);  //isko O(n) me karo pop and peek ko O(1) me
    }

    public int pop() {
        if(s1.size()==0)
        {
            return -1;
        }
        while(s1.size()>1)
        {
            s2.push(s1.pop());
        }
        int a=s1.pop();
        while(s2.size()>0)
        {
            s1.push(s2.pop());
        }
        return a;
    }

    public int peek() {
        if(s1.size()==0)
        {
            return -1;
        }
        while(s1.size()>0)
        {
            s2.push(s1.pop());
        }
        int a=s2.peek();
        while(s2.size()>0)
        {
            s1.push(s2.pop());
        }
        return a;

    }

    public boolean empty() {
        return s1.size()==0;
    }
}
```

Implement stack using Linkedlist

04 April 2025 12:11

Implement queue using Linkedlist

04 April 2025 12:11

Check for balanced paranthesis

04 April 2025 12:11

```
public boolean isValid(String s)
    {
        Stack<Character> st = new Stack<Character>();
        for (char it : s.toCharArray()) {
            if (it == '(' || it == '[' || it == '{')
                st.push(it);
            else {
                if(st.isEmpty()) return false;
                char ch = st.pop();
                if((it == ')' && ch == '(') ||  (it == ']' && ch == '[') || (it == 
'}' && ch == '{')) continue;
                else return false;
            }
        }
        return st.isEmpty();

    }
```

Implement Min Stack

04 April 2025 12:12

```
class MinStack {
    Stack<Long>s;
    Long ans;
    public MinStack() {
        s=new Stack<Long>();
        ans=Long.MAX_VALUE;
    }

    public void push(int val)
    {
        Long val1 = Long.valueOf(val);  //int to Long
        if(s.size()==0)
        {

            s.push(val1);
            ans=val1;
        }
        else if(val1>=ans)
        {
            s.push(val1);
        }
        else
        {
            Long a=2*val1-ans;
            s.push(a);
            ans=val1;
        }
        //System.out.println(s.peek());
        return;

    }

    public void pop()
    {
        if(s.size()<=0)
        {
            return;
        }
        if(s.peek()<ans)
        {
            //int a=2*ans-s.peek();
            //ans=a;
            ans=2*ans-s.peek();
        }
       s.pop();

    }

    public int top() {
        if(s.size()==0)
        {
            return -1;
        }
        return (s.peek() < ans) ? ans.intValue() : s.peek().intValue();
    }

    public int getMin() {
        return ans.intValue(); //Long to int
    }
}
```

Infix to Postfix Conversion using Stack

04 April 2025 14:25

Prefix to Infix Conversion

04 April 2025 14:25

Prefix to Postfix Conversion

04 April 2025 14:25

Postfix to Prefix Conversion

04 April 2025 14:26

Postfix to Infix

04 April 2025 14:26

Convert Infix To Prefix Notation

04 April 2025 14:26

Next Greater Element

04 April 2025 14:26