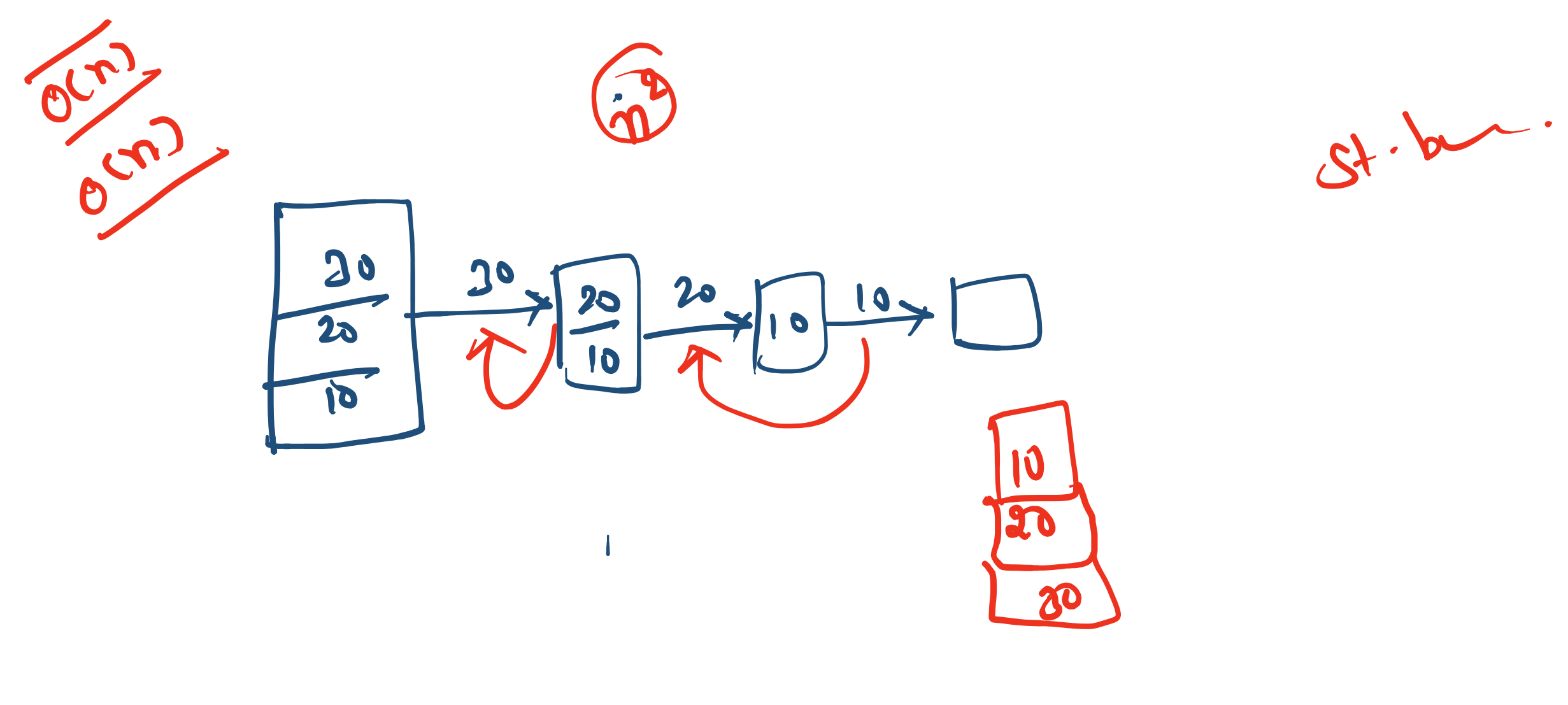
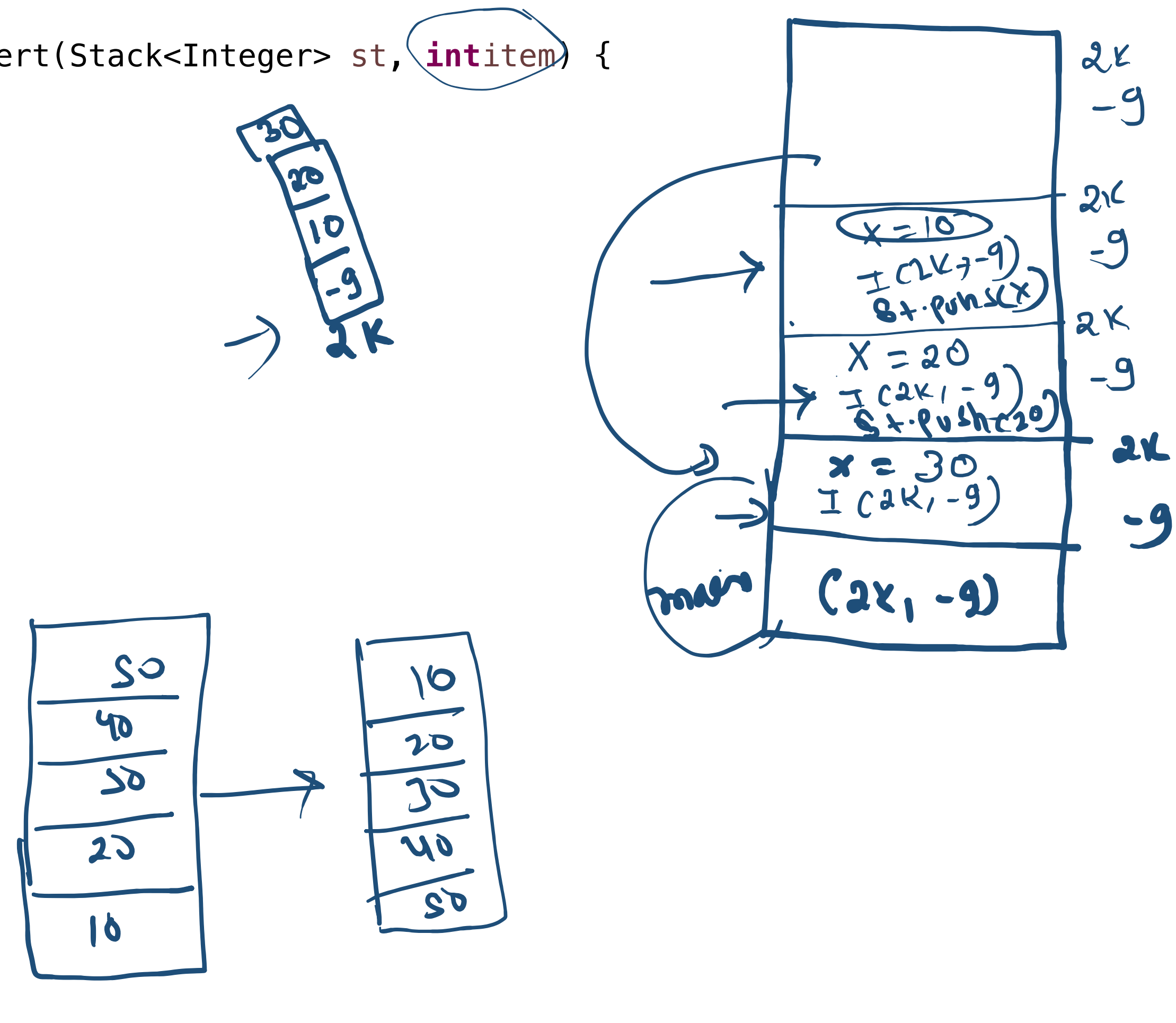
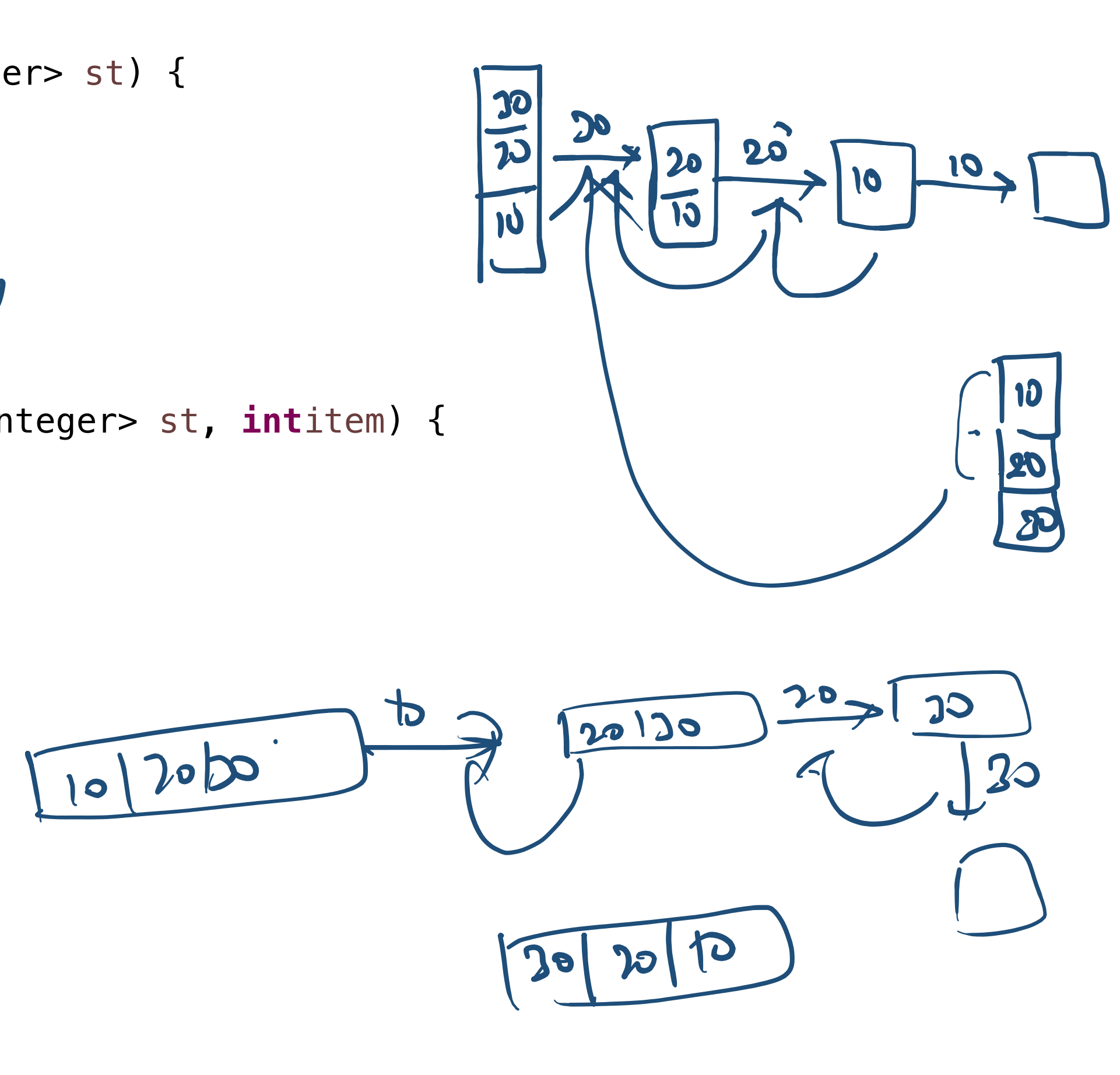


```
public static void Insert(Stack<Integer> st, int item) {  
    if(st.isEmpty()) {  
        st.push(item);  
        return;  
    }  
    int x = st.pop();  
    Insert(st, item);  
    st.push(x);  
}
```



```
public static void Revser(Stack<Integer> st) {  
    if(st.isEmpty()) {  
        return;  
    }  
    int x = st.pop();  
    Revser(st);  
    Insert(st, x);  
}  
  
public static void Insert(Stack<Integer> st, int item) {  
    if(st.isEmpty()) {  
        st.push(item);  
        return;  
    }  
    int x = st.pop();  
    Insert(st, item);  
    st.push(x);  
}
```



Handwritten notes and diagrams for string reversal. Includes a diagram of a stack [1, 2, 3, 4, 5] being reversed to [5, 4, 3, 2, 1]. Shows the process of popping elements from the stack and pushing them back. Includes a diagram of a linked list [1, 2, 3, 4, 5] being reversed to [5, 4, 3, 2, 1].

```
public static String Construct_Smallest(String s) {  
    int[] arr = new int[s.length() + 1];  
    Stack<Integer> st = new Stack<>();  
    int c = 1;  
    for(int i = 0; i < s.length(); i++) {  
        if(i == s.length() || s.charAt(i) == 'I') {  
            arr[i] = c;  
            c++;  
            while(!st.isEmpty()) {  
                arr[st.pop()] = c;  
                c++;  
            }  
            st.push(i);  
        }  
    }  
    String ans = "";  
    for(int i = 0; i < arr.length; i++) {  
        ans = ans + arr[i];  
    }  
    return ans;  
}
```

