Given a stack of integers, sort it in descending order using another temporary stack.

In [45]:

```
def sortstack(num):
    stack=[]
    tmpstack=[]
    for i in range(len(num)):
        stack.append(num[i])
    while(len(stack)!=0):
        tmp=top(stack)
        pop(stack)

        while(len(tmpstack)!=0 and top(tmpstack)<tmp):
            push(stack,top(tmpstack))
            pop(tmpstack)
        push(tmpstack,tmp)
    return tmpstack
def push (stack,item):
    stack.append(item)

def top(stack):
    p= len(stack)
    return stack[p-1]
def pop(stack):
    if(len(stack)==0):
        print("Stack Underflow")
        exit(1)
    return stack.pop()
def prints(stack):
    print("descending Order",end=" ")
    for i in range(len(stack)-1, -1, -1):
        print(stack[i], end = ' ')
    print()

num=[34,3,31,98,92,23]

sortest = sortstack(num)
print("Input",num)
prints(sortest)
```

```
Input [34, 3, 31, 98, 92, 23]
descending Order 3 23 31 34 92 98
```

You are given a string ‘s’ consisting of lowercase English letters. A duplicate removal consists of choosing two adjacent and equal letters and removing them.

In [24]:

```
def ShortenString(str1):
    st = []
    i=0
    while i < len(str1):

        if len(st)== 0 or str1[i] != st[-1]:
            st.append(str1[i])
            i += 1

        else:
            st.pop()
            i += 1

    if len(st)== 0:
        return("Empty String")

    else:
        short_string = ""
        for i in st:
            short_string += str(i)
        return(short_string)


str1 ="abbaca"
print("Input:-",str1)
print("Output:-",ShortenString(str1))

```

```
Input:- abbaca
Output:- ca
```

Given an array, print the Next Greater Element (NGE) for every element. The Next greater Element for an element x is the first greater element on the right side of x in the array. Elements for which no greater element exists, consider the next greater element as -1.

In [44]:

```
def NextGreaterElement(arr):
    n=len(arr)
    vector=[None]*n
    stack=[]
    j=0

    for i in range(n-1,-1,-1):
        if(len(stack)==0):
            vector[j]=-1
            stack.append(arr[i])
            j+=1
        elif(len(stack)>0 and top(stack)>arr[i]):
            vector[j]=top(stack)
            j+=1
            stack.append(arr[i])


        elif(len(stack)> 0 and top(stack)<=arr[i]):
            while(len(stack)>0 and top(stack)<=arr[i]):
                stack.pop()

            if(len(stack)==0):
                vector[j]=-1
                j+=1

            else:
                vector[j]=top(stack)
                j+=1
                stack.append(arr[i])
    return vector
def top(stack):
    p= len(stack)
    return stack[p-1]

arr=[4,5,2,25]
print("Input:-",arr)
vector=NextGreaterElement(arr)
print("Output:-",end=" ")
for i in range(len(vector)-1,-1,-1):
    print(vector[i],end=" ")
```

```
Input:- [4, 5, 2, 25]
Output:- 5 25 25 -1
```

You are keeping score for a baseball game with strange rules. The game consists of several rounds, where the scores of past rounds may affect future rounds' scores.

In [2]:

```
def baseball(ops):
    stack=[]
    sum2=0
    for i in range(len(ops)):
        if(ops[i]=="5"):
            stack.append(int(ops[i]))
        elif(ops[i]=="2"):
            stack.append(int(ops[i]))
        elif(ops[i]=="C"):
            stack.pop()
        elif(ops[i]=="D"):
            temp=top(stack)
            pop=stack.pop()
            double=2*temp
            stack.append(double)
            stack.append(pop)
        elif(ops[i]=="+"):
            pop1=stack.pop()
            pop2=stack.pop()
            sum1=pop1+pop2
            stack.append(sum1)
            stack.append(pop1)
            stack.append(pop2)
    for i in range(len(stack)):
        sum2=sum2+stack[i];
    return sum2


def top(stack):
    p= len(stack)
    return stack[p-1]

ops = ["5","2","C","D","+"]
print(baseball(ops))
```

```
30
```

In [ ]:

```

```