SWE 2001 DA-1

NAME: PRANAY AGRAWAL
REGISTRATION NO: 22MIS0172

Q1) Given an array A of n integers. You have to make a queue and stack of the given integers. Queue should contain only prime numbers and stack should contain only composite numbers. Display queue and stack contents.

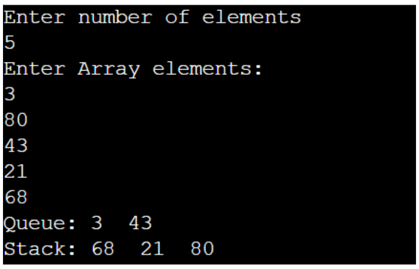
AIM: To create a queue of prime numbers and a stack of composite numbers from an array A of integers, and display the contents of both data structures.

Code:

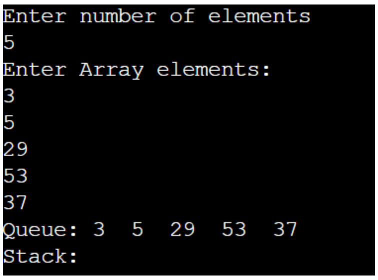
```
#include <stdio.h>
int top=-1,front=-1,rear=-1,fac=0;
void push(int n,int stk[],int data){
  if(top==(n-1)){
    printf("Stack is Full");
  }
  else{
    top+=1;
    stk[top]=data;
  }
}
void add(int n,int que[],int data){
  if(rear==(n-1)){
    printf("Queue is full");
  }
  else{
    if(rear==-1){
      front=0;
    }
    rear+=1;
    que[rear]=data;
  }
```

```
}
void main()
{
   int n,k=0;
   printf("Enter number of elements\n");
   scanf("%d",&n);
   int arr[n];
   printf("Enter Array elements:\n");
   for(int i=0;i<n;i++){
      scanf("%d",&arr[i]);
   }
   for(int i=0;i<n;i++){
      if(arr[i]<2){
            k+=1;


         }
      for(int j=2;j<arr[i];j++){
         if(arr[i]%j==0){
            k+=1;
            break;


         }
      }
   }
   int t=n-k;
   int stk[k];
   int que[t];
   for(int i=0;i<n;i++){
      if(arr[i]<2){
            fac=1;
            }
            else{
        for(int j=2;j<arr[i];j++){
            if(arr[i]%j==0){
                fac=1;
                break;

            }
        }
            }
            if(fac==1){
                push(k,stk,arr[i]);
            }
            else{
                add(t,que,arr[i]);
            }
            fac=0;
    }
    printf("Queue: ");
    for(int i=0;i<=rear;i++){
        printf("%d  ",que[i]);
    }
    printf("\nStack: ");
    for(int i=top;i>=0;i--){
        printf("%d  ",stk[i]);
    }
}
```

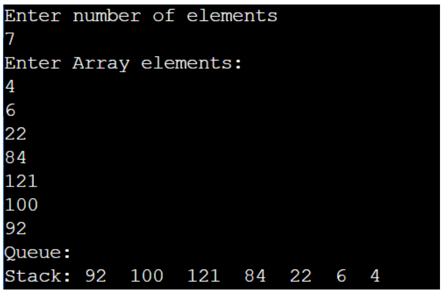
Sample Input/Output:



```
Enter number of elements
5
Enter Array elements:
3
80
43
21
68
Queue: 3  43
Stack: 68  21  80
```



```
Enter number of elements
5
Enter Array elements:
3
5
29
53
37
Queue: 3  5  29  53  37
Stack:
```



```
Enter number of elements
7
Enter Array elements:
4
6
22
84
121
100
92
Queue:
Stack: 92  100  121  84  22  6  4
```

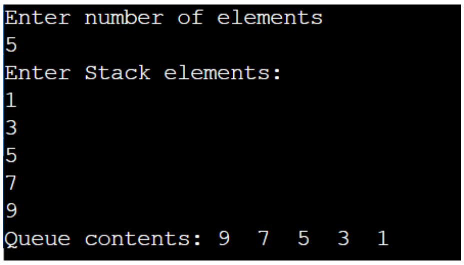
Q2) Using push and pop operations of stack create a queue and display the contents of queue. NOTE: Stack’s property is LIFO and Queue’s property is FIFO.

AIM: To create a queue and display its elements using stack.

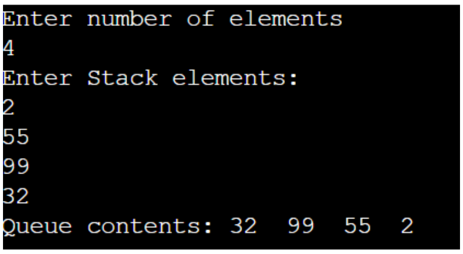
Code:

```
#include <stdio.h>
int top=-1,front=-1,rear=-1,fac=0;
void push(int n,int stk[],int data){
   if(top==(n-1)){
      printf("Stack is Full");
   }
   else{
      top+=1;
      stk[top]=data;
   }
}
int pop(int stk[]){
   if(top==-1){
        printf("Stack is Empty");
    }
    else{
        return stk[top--];
    }
}
void add(int n,int que[],int data){
    if(rear==(n-1)){
        printf("Queue is full");
    }
    else{
        if(rear==-1){
            front=0;
        }
        rear+=1;
        que[rear]=data;
    }
}
void main()
{
    int n,t;
    printf("Enter number of elements\n");
    scanf("%d",&n);
    int stk[n];
    int que[n];
    printf("Enter Stack elements:\n");
    for(int i=0;i<n;i++){
        scanf("%d",&t);
        push(n,stk,t);
    }
    for(int i=0;i<n;i++){
        add(n,que,pop(stk));
    }


    printf("Queue contents: ");
    for(int i=0;i<=rear;i++){
        printf("%d  ",que[i]);
    }



}
```

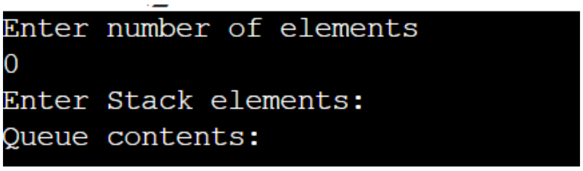
Sample Input/Output:



```
Enter number of elements
5
Enter Stack elements:
1
3
5
7
9
Queue contents: 9  7  5  3  1
```



```
Enter number of elements
4
Enter Stack elements:
2
55
99
32
Queue contents: 32  99  55  2
```



```
Enter number of elements
0
Enter Stack elements:
Queue contents:
```

Q3) Humpy likes to jump from one building to another. But he only jumps to next higher building and stops when no higher building is available. Stamina required for a journey is xor of all the heights on which humpy jumps until he stops.

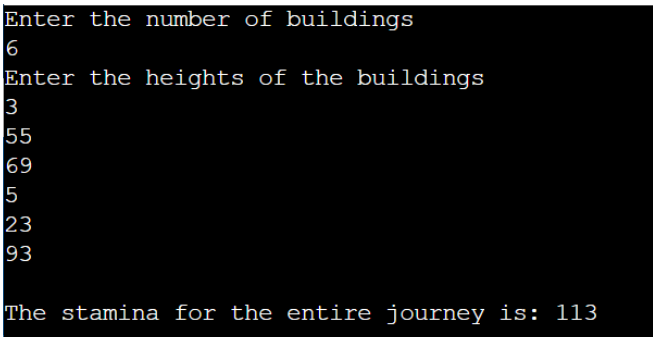
AIM: To calculate the XOR of the heights of the buildings Humpy jumps onto keeping the condition that he only jumps to the next higher building and stops when no higher building is available.

Code:

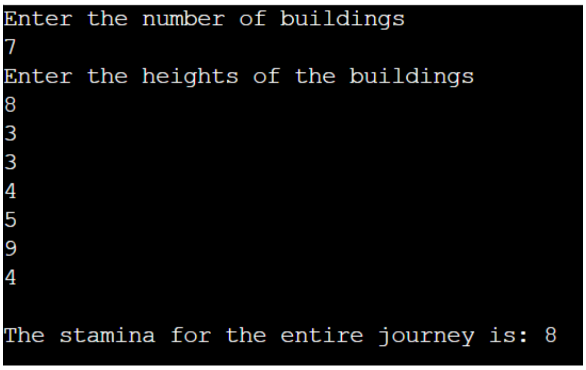
```
#include <stdio.h>


int main()
```

```
{
   int n,st=0;
   printf("Enter the number of buildings\n");
   scanf("%d",&n);
   int arr[n];
   printf("Enter the heights of the buildings\n");
   for(int i=0;i<n;i++){
      scanf("%d",&arr[i]);
   }
   int i=0;
   while(i<n && (i==0||arr[i]>arr[i-1])){
      st^=arr[i];
      i++;
   }
   printf("\nThe stamina for the entire journey is: %d",st);
}
```

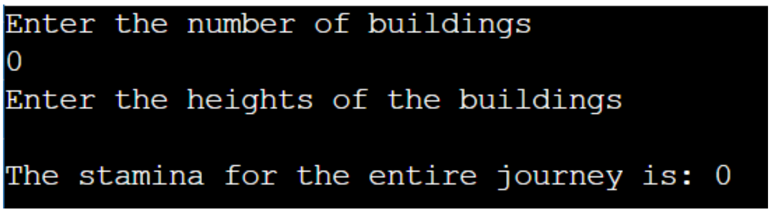
Sample Input/Output:



```
Enter the number of buildings
6
Enter the heights of the buildings
3
55
69
5
23
93

The stamina for the entire journey is: 113
```



```
Enter the number of buildings
7
Enter the heights of the buildings
8
3
3
4
5
9
4

The stamina for the entire journey is: 8
```



```
Enter the number of buildings
0
Enter the heights of the buildings

The stamina for the entire journey is: 0
```