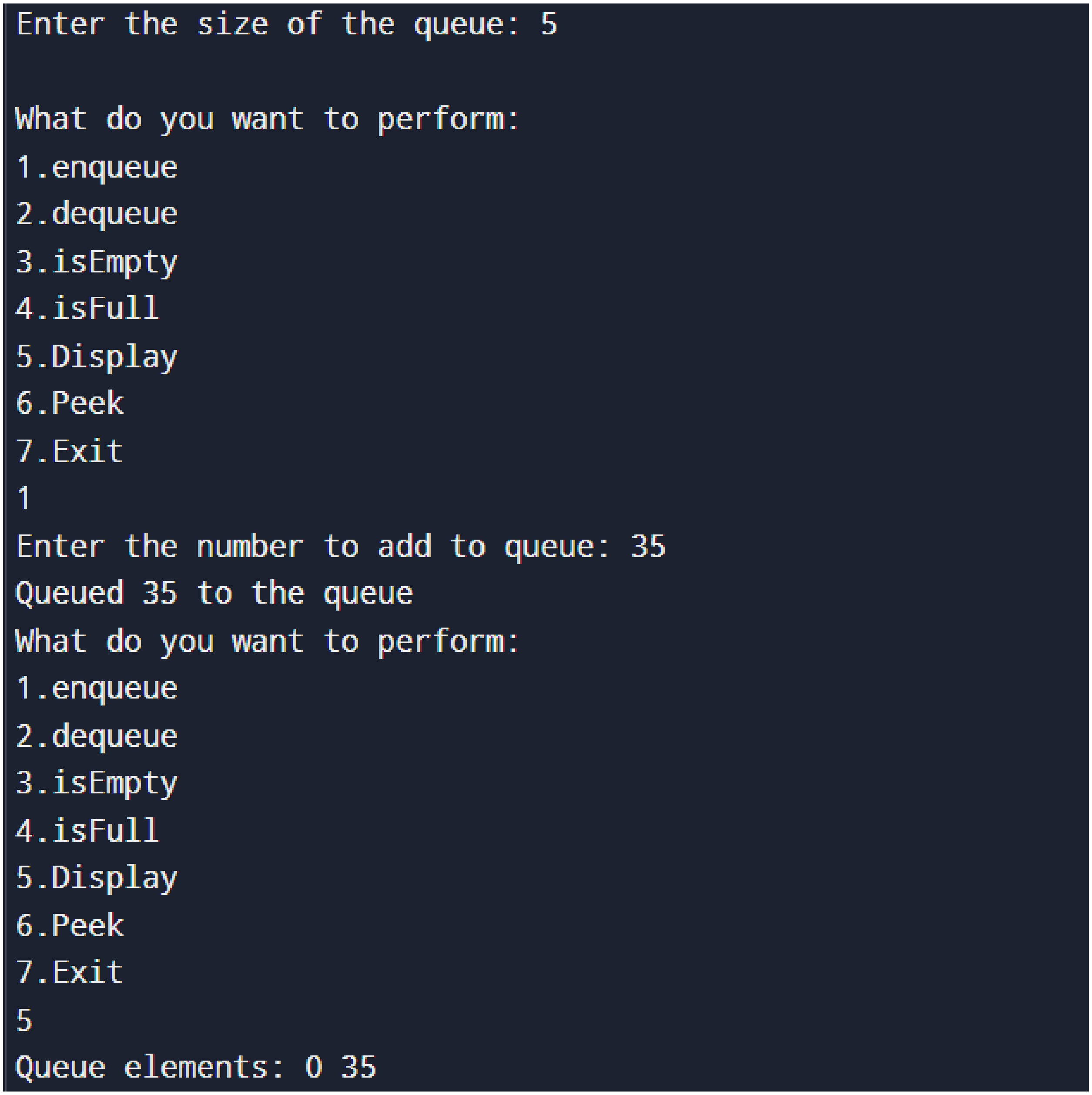
DSA Assignment 4

Q1

```
#include <iostream>
#include <queue>
using namespace std;

int main(){

    int size;
    cout<<"Enter the size of the queue: ";
    cin>>size;

    int qu[size];
    int sizern=0;
    int front=-1;
    int back=-1;

    while(true){
        cout<<("\nWhat do you want to perform: \n");
        cout<<("1.enqueue\n");
        cout<<("2.dequeue\n");
        cout<<("3.isEmpty\n");
        cout<<("4.isFull\n");
        cout<<("5.Display\n");
        cout<<("6.Peek\n");
        cout<<("7.Exit\n");
        int num;
        cin>>num;

        switch (num)
        {
        case 1:
            if(sizern<size){
                int numPush;
                back++;
                cout<<"Enter the number to add to queue: ";
                cin>>numPush;
                qu[back]=numPush;
                cout<<"Queued "<<numPush<<" to the queue";
                sizern++;
            }
        else{
            cout<<"queue is full!";
        }

        break;
    case 2:
        if(sizern!=0){
            front++;
            int numPop=qu[front];
            sizern--;
            cout<<"Removed "<<numPop<<" from the queue";
            if(sizern==0){
                front=-1;
                back=-1;
            }
        }
        else{
            cout<<"The queue is already empty!";
        }
        break;
    case 3:
        if(sizern==0){
            cout<<"The queue is empty";
        }
        else{
            "The queueue is not empty";
        }

        break;
    case 4:
        if(sizern==size){
            cout<<"The queue is full";
        }
        else{
            cout<<"The queue is not full";
        }
        break;
    case 5:
        if (sizern==0) {
            cout << "Queue is empty!";
        } else {
            cout << "Queue elements: ";
            for(int i=front;i<=back;i++){
                cout<<qu[i]<<" ";
                }
            }
            break;
        case 6:
            if (sizern==0) {
                cout << "Queue is empty!";
            } else {
                cout << "Front element: " << qu[front];
            }
            break;
            break;
        case 7:
            cout<<"Exiting! ";
            return 0;

        default:
            cout<<"Invalid code. Try again!";
            break;
        }
    }
}
```



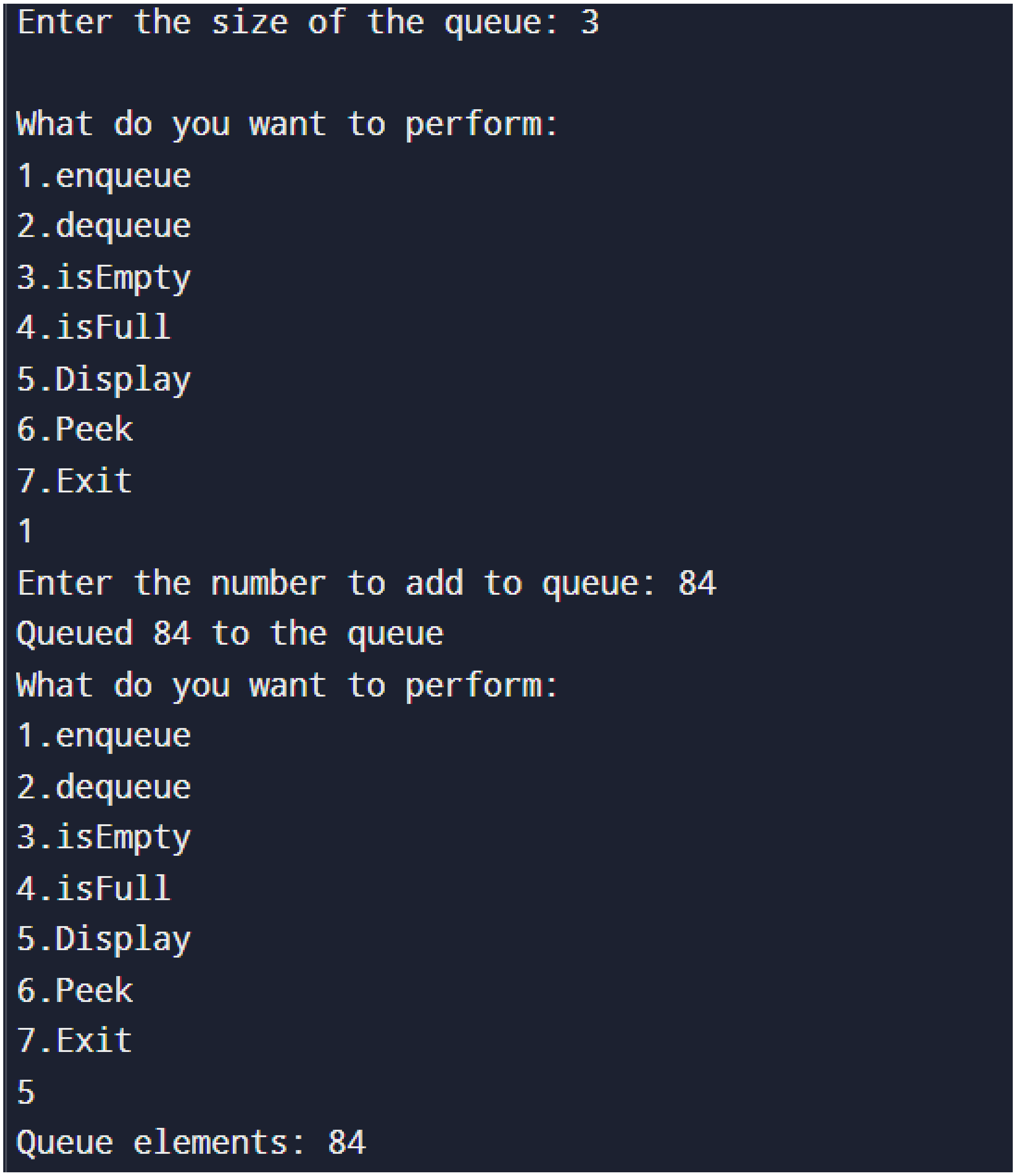
```
Enter the size of the queue: 5

What do you want to perform:
1.enqueue
2.dequeue
3.isEmpty
4.isFull
5.Display
6.Peek
7.Exit
1
Enter the number to add to queue: 35
Queued 35 to the queue
What do you want to perform:
1.enqueue
2.dequeue
3.isEmpty
4.isFull
5.Display
6.Peek
7.Exit
5
Queue elements: 0 35
```

Q2

```
#include <iostream>
#include <queue>
using namespace std;

int main(){

   int size;
```

```
cout<<"Enter the size of the queue: ";
cin>>size;

int qu[size];
int sizern=0;
int front=-1;
int back=-1;

while(true){
    cout<<("\nWhat do you want to perform: \n");
    cout<<("1.enqueue\n");
    cout<<("2.dequeue\n");
    cout<<("3.isEmpty\n");
    cout<<("4.isFull\n");
    cout<<("5.Display\n");
    cout<<("6.Peek\n");
    cout<<("7.Exit\n");
    int num;
    cin>>num;

    switch (num)
    {
    case 1:
        if(sizern<size){
            if(front==-1){
                front=0;
            }
            int numPush;
            back=(back+1) % size;
            cout<<"Enter the number to add to queue: ";
            cin>>numPush;
            qu[back]=numPush;
            cout<<"Queued "<<numPush<<" to the queue";
            sizern++;
        }
        else{
            cout<<"queue is full!";
    }

    break;
case 2:
    if(sizern!=0){
        int numPop=qu[front];
        front=(front+1)%size;
        sizern--;
        cout<<"Removed "<<numPop<<" from the queue";
        if(sizern==0){
            front=-1;
            back=-1;
        }
    }
    else{
        cout<<"The queue is already empty!";
    }
    break;
case 3:
    if(sizern==0){
        cout<<"The queue is empty";
    }
    else{
        cout<<"The queueue is not empty";
    }

    break;
case 4:
    if(sizern==size){
        cout<<"The queue is full";
    }
    else{
        cout<<"The queue is not full";
    }
    break;
case 5:
    if (sizern==0) {
```

```
                cout << "Queue is empty!";
            } else {
                cout << "Queue elements: ";
                for(int i=0;i<sizern;i++){
                    cout<<qu[(front + i) % size] << " ";
                }
            }
            break;
        case 6:
            if (sizern==0) {
                cout << "Queue is empty!";
            } else {
                cout << "Front element: " << qu[front];
            }
            break;
            break;
        case 7:
            cout<<"Exiting! ";
            return 0;

        default:
            cout<<"Invalid code. Try again!";
            break;
        }
    }
}
```



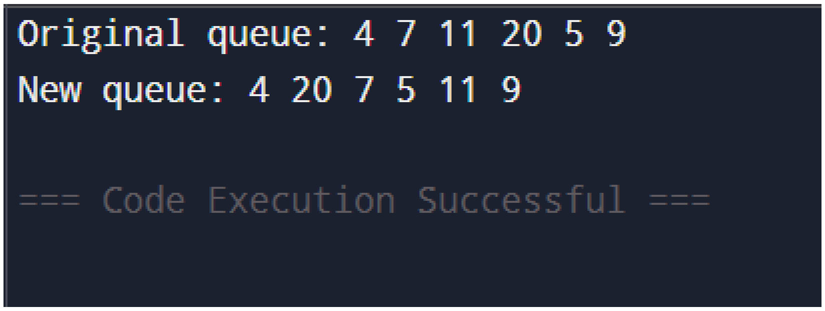
```
Enter the size of the queue: 3

What do you want to perform:
1.enqueue
2.dequeue
3.isEmpty
4.isFull
5.Display
6.Peek
7.Exit
1
Enter the number to add to queue: 84
Queued 84 to the queue
What do you want to perform:
1.enqueue
2.dequeue
3.isEmpty
4.isFull
5.Display
6.Peek
7.Exit
5
Queue elements: 84
```

Q3

```
#include <iostream>
#include <queue>
using namespace std;

int main(){

   queue<int> qu;
```

```
int arr[]={4,7,11,20,5,9};

int n=sizeof(arr)/sizeof(arr[0]);

for(int i=0;i<n;i++){
    qu.push(arr[i]);
}
queue<int> temp;
temp=qu;

cout<<"Original queue: ";
for(int i=0;i<n;i++){
    cout<<temp.front()<<" ";
    temp.pop();
}

temp=qu;
queue<int> newqu;

for(int i=0;i<n/2;i++){
    temp.pop();
}
queue<int> temp1=qu;

for(int i=0;i<n;i++){
    if(i%2==0){
        newqu.push(temp1.front());
        temp1.pop();
    }
    else{
        newqu.push(temp.front());
        temp.pop();
    }
}
queue<int> temp2=newqu;

cout<<"\nNew queue: ";
```

```
  for(int i=0;i<n;i++){
    cout<<temp2.front()<<" ";
    temp2.pop();
  }
  return 0;
}
```

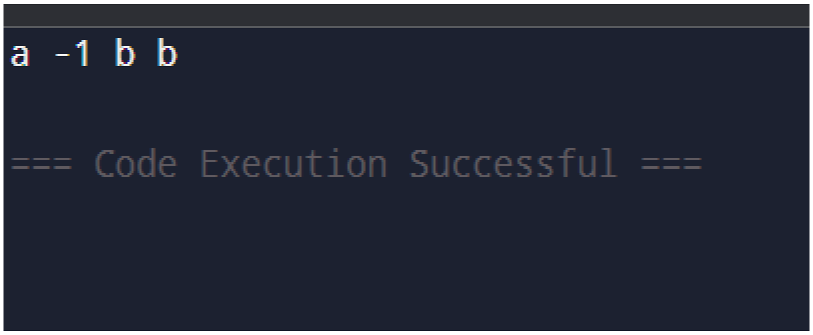


```
Original queue: 4 7 11 20 5 9
New queue: 4 20 7 5 11 9

=== Code Execution Successful ===
```

Q4

```
#include <iostream>
#include <queue>
using namespace std;

int main(){

  int freq[26] = {0};
  queue<char> qu;
  char arr[]={'a','a','b','c'};

  int size=sizeof(arr)/sizeof(arr[0]);

  for(int i=0;i<size;i++){
    char ch=arr[i];
    freq[ch-'a']++;
    qu.push(ch);
    while(freq[qu.front()-'a']>1){
      qu.pop();
    }
    if(qu.empty()){
      cout<<-1<<" ";
```

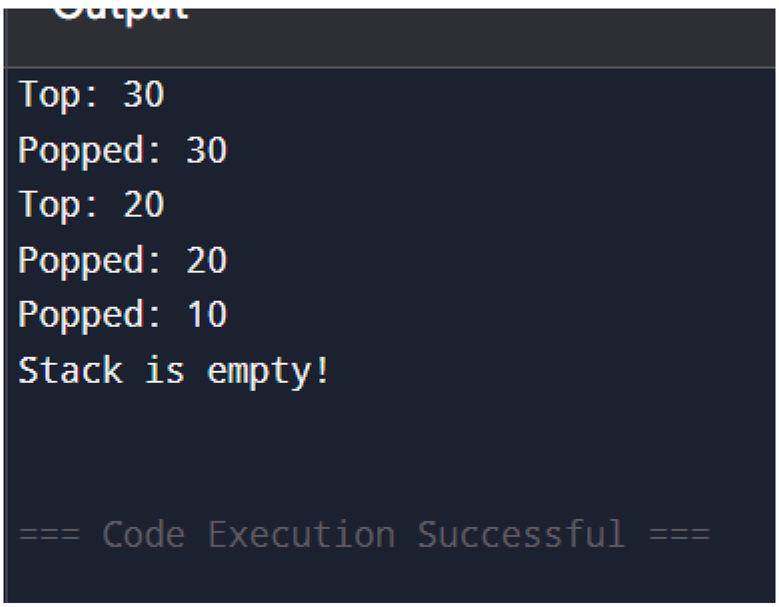
```
        }
        else{
        cout<<qu.front()<<" ";
        }
    }
    return 0;
}
```



```
a -1 b b

=== Code Execution Successful ===
```

Q5 A

```
#include <iostream>
#include <queue>
using namespace std;

queue<int> q1, q2;

void push(int x) {
    q2.push(x);

    // Move all elements from q1 → q2
    while(!q1.empty()) {
        q2.push(q1.front());
        q1.pop();
    }
    swap(q1, q2);
}

void pop(){
    if (q1.empty()){
        cout<<"Stack is empty!\n";
        return;
    }
    cout<<"Popped: "<< q1.front() <<endl;
    q1.pop();
}

int top(){
    if (q1.empty()){
        cout<<"Stack is empty!\n";
        return -1;
    }
    return q1.front();
}

bool empty(){
    return q1.empty();
}

int main() {
    push(10);
    push(20);
    push(30);

    cout <<"Top: "<<top()<<endl;
    pop();
    cout <<"Top: "<<top()<<endl;
    pop();
    pop();
    pop();
}
```

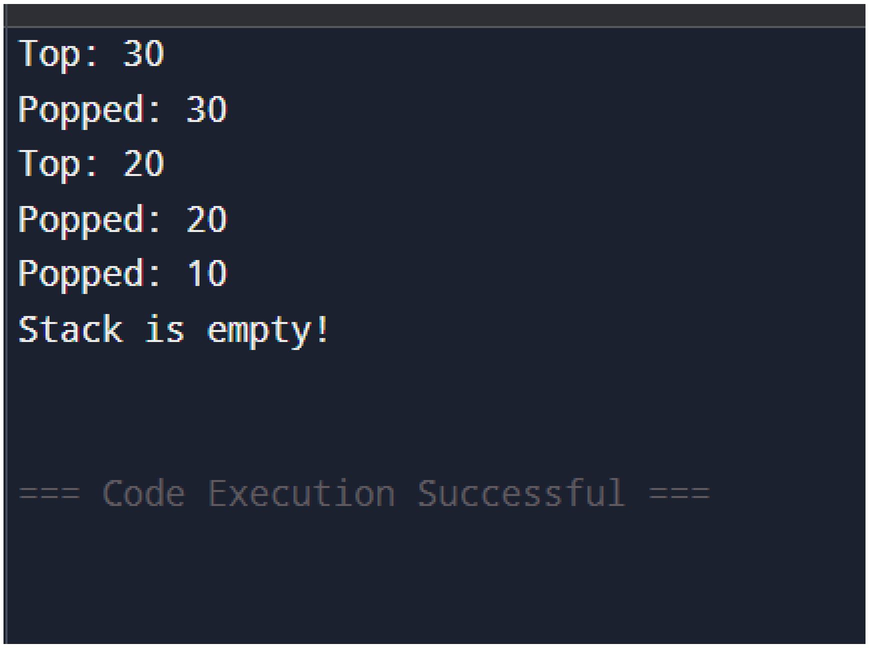


```
Top: 30
Popped: 30
Top: 20
Popped: 20
Popped: 10
Stack is empty!


=== Code Execution Successful ===
```

Q5 B

```
#include <iostream>
#include <queue>
using namespace std;

queue<int> q;

void push(int x) {
   int size = q.size();
   q.push(x);

   // Rotate elements to make new element front
   for (int i=0; i<size; i++) {
      q.push(q.front());
      q.pop();
   }
}

void pop(){
   if (q.empty()){
      cout<<"Stack is empty!\n";
      return;
```

```
    }
    cout<<"Popped: "<<q.front()<<endl;
    q.pop();
}

int top(){
    if(q.empty()) {
        cout<<"Stack is empty!\n";
        return -1;
    }
    return q.front();
}

bool empty(){
    return q.empty();
}

int main(){
    push(10);
    push(20);
    push(30);

    cout<<"Top: "<<top()<<endl;
    pop();
    cout<<"Top: "<<top()<<endl;
    pop();
    pop();
    pop();
}
```



```
Top: 30
Popped: 30
Top: 20
Popped: 20
Popped: 10
Stack is empty!


=== Code Execution Successful ===
```