

Tutorial – Stack and Queues

Information: Program templates for questions 1-3 are given as separated files. You must use them to implement your functions.

1. **(reverseStack)** Write a c function `reverseStack()` that reverses a stack using a **queue**. Note that the `reverseStack()` function only uses `push()` and `pop()` when adding or removing integers from the stack, and only uses `enqueue()` and `dequeue()` when adding or removing integers from the queue.

The function prototypes are given as follows:

```
void reverseStack(Stack *s);
```

For example: if the stack is <5, 4, 3, 2, 1>, the resulting stack will be <1, 2, 3, 4, 5>

Sample test cases are given below:

```
1: Insert an integer into the stack;
2: Reverse the stack;
0: Quit;
```

```
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 1
The resulting stack is: 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 2
The resulting stack is: 2 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 3
The resulting stack is: 3 2 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 4
The resulting stack is: 4 3 2 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 5
The resulting stack is: 5 4 3 2 1

Please input your choice(1/2/0): 2
The resulting stack after reversing its elements is: 1 2 3 4
5

Please input your choice(1/2/0): 0
```

2. **(reverseFirstKItems)** Write a c function `reverseFirstKItems()` that reverses the order of the first k elements of the queue using a **stack**, leaving the other elements in the same relative order for given an integer k and a queue of integers. Note that the `reverseFirstKItems()` function only uses `push()` and `pop()` when adding or removing integers from the stack and only uses `enqueue()` and `dequeue()` when adding or removing integers from the queue.

The function prototypes are given as follows:

```
void reverseFirstKItems(Queue *q, int k);
```

For example: if the queue is <1, 2, 3, 4, 5, 6> and k=3, the resulting queue will be <3, 2, 1, 4, 5, 6>

Sample test cases are given below:

```
1: Insert an integer into the queue;
2: Reverse the elements of the queue until the given number;
0: Quit;
```

```

Please input your choice(1/2/0): 1
Input an integer that you want to insert into the queue: 1
The resulting queue is: 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the queue: 2
The resulting queue is: 1 2
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the queue: 3
The resulting queue is: 1 2 3
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the queue: 4
The resulting queue is: 1 2 3 4
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the queue: 5
The resulting queue is: 1 2 3 4 5
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the queue: 6
The resulting queue is: 1 2 3 4 5 6

Please input your choice(1/2/0): 2
Enter an integer to reverse the queue until that number: 3
The resulting queue after reversing first 3 elements is: 3 2
1 4 5 6

Please input your choice(1/2/0): 0

```

3. **(sortStack)** Write a c function `sortStack()` that sorts a given stack in ascending order using another temporary stack. Note that the `sortStack()` function only uses `push()` and `pop()` when adding or removing integers from the stack.

The function prototype is given as follows:

```
void sortStack(Stack *s);
```

For example, if the stack is <6, 5, 4, 3, 2, 1>, the resulting stack will be <1, 2, 3, 4, 5, 6>

Sample test cases are given below:

```

1: Insert an integer into the stack;
2: Sort the stack in ascending order ;
0: Quit;

Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 1
The resulting stack is: 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 2
The resulting stack is: 2 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 3
The resulting stack is: 3 2 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 4
The resulting stack is: 4 3 2 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 5
The resulting stack is: 5 4 3 2 1
Please input your choice(1/2/0): 1
Input an integer that you want to insert into the stack: 6
The resulting stack is: 6 5 4 3 2 1

```

```
Please input your choice(1/2/0): 2
The resulting stack after sorting it in ascending order is: 1
2 3 4 5 6
Please input your choice(1/2/0): 0
```