Documentation

1 CODE OUTLINE:

class Deque {
private:
   string *queue, remove;
   int num_elements, size_of_queue, head, tail, previousHead, previousTail,minSize;
public:
   Deque();//Default constructor
   void grow(int, T*); //Doubles the size of the queue when needed, this method is wrapped around a try/catch structure to catch unexpected error(i.e out of memory and etc). More specifically I threw an OUT_OF_RANGE exception.
   void shrink(int, T*); //Halves the size of the queue when needed this method is wrapped around a try/catch structure to catch unexpected error(i.e out of memory and etc). More specifically I threw an OUT_OF_RANGE exception.—Altho it might never happen when calling shrink method…
   ~Deque(); //Destructor, deletes T *queue when program terminates
   void push_front(T);//adds elements to the front of the queue, calls grow when queue is full, and copied content from the old array to the new array in the following manner: copied the element from previousTail(of the old array) as the 1st element of the new array, and tail++ until it wraps around and copied every element from the old array to the new array.(essentially when tail and head meet)
   void push_back(T);//adds elements to the back of the queue, calls shrink when queue is less than ¼ full, and copied content from the old array to the new array in the following manner: copied the element from previousTail(of the old array) as the 1st element of the new array, and tail++ until it wraps around and copied every element from the old array to the new array.(essentially when tail and head meet)
   string pop_front();//removes the element at the front of the queue—1)Fixed bug found in this method from previous submission of part a, made sure it didn't shrink when the head happens to be @ index 0 and ¼ full when the queue is @ min. size 8,(like it would shrink if there's only 2 element left in size 8 queue) this caused problem when it called the shrink method, and when it called pop_front again, it returned an empty string.---so fixed. 2) Checks and handles the error if the queue happens to be empty.
   string pop_back();//removes the element at the back of the queue. Checks and handles the error if the queue happens to be empty.
   int size();//returns the size_of_queue , or the size of my array
   bool empty();//returns true if the queue is empty
string toStr(); //the toStr() method the professor wants
void print_queue();//prints the queue in another way—this method is commented out

2 General Description

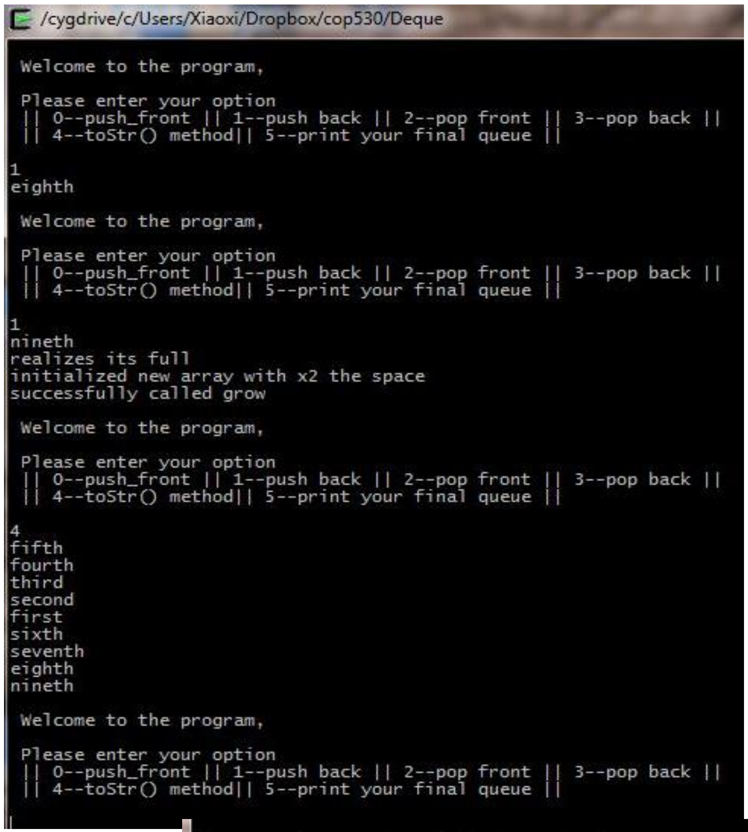
For this part of the project, I implemented a double ended queue that grows when the array is full, and shrinks when it's ¼ full using Templates. I initialized the "head" on the left and of the array, and "tail" on the right end of the array. (They change obviously as user push/pop accordingly) Handles the 2 errors the professor specified, handle errors when popping front and back of an empty array, as well as trying to catch out_of_memory errors.

3 Test Cases:

TEST CASE	TEST DESCRIPTION	EXPECTED RESULT	ACTUAL RESULT	PASS/FAIL
1	push element into an empty queue	Element gets pushed	Element gets pushed	Passed
2	pop an element form an empty queue	Nothing happens/throw exception	Code cout a line saying there's nothing to pop	Passed
3	push element to a full queue(using push_front)	Element gets pushed and queue size doubles	Element gets pushed and queue size doubles	Passed
4	Element gets popped and queue size shrinks by half(both front and back tested)	Element gets popped and queue size shrinks by half	Element gets popped and queue size shrinks by half	Passed
5	Pushing in front the front 6 elements, and pushing front back 3 elements	Elements gets push and doubled bc it reached capacity	Same as expected	Passed
6	Poping from back from the populated array in test case 5	Items get pop out in the order they should be	Same as expected	Passed
7	Pushing 9 elements from front and pop 6 of them from back	Array should populate and double after it gotten to 9, and shrinks after poping so many elements	Same as expected	Passed
8	Populated 17 elements randomly, and poped out from the front continuously until there is only 8 elements in the array	Should expect growing and shrinking of array respectively.	Same as expected	Passed
9	Continue to pop things randomly from front and back from the remaining of above array until its empty, and pop front and back couple times more	To throw some sort of exceptions or error	I wrote my code to catch this in if/else statements, so cout statements was throwed, and suggested the user to put in more stuff before they can pop anymore.	passed
10	Randomly push and pop	Print out accordingly	Same as expected	Passed

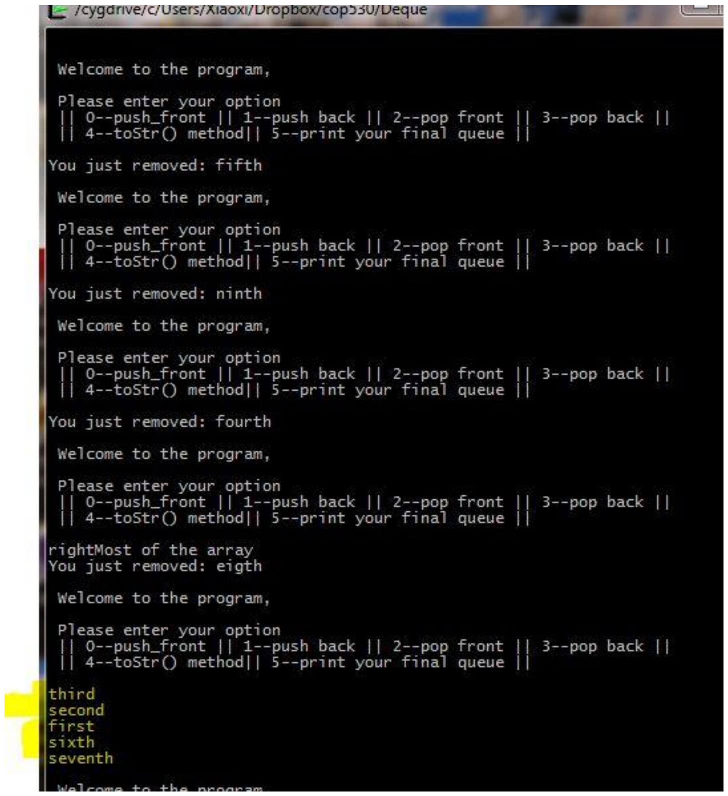
4 Sample output screenshots:

Sample output 1

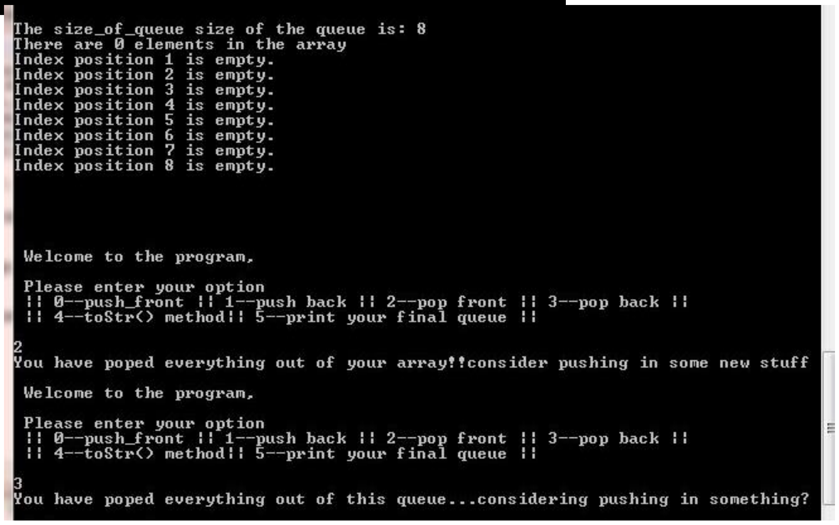


```
/cygdrive/c/Users/Xiaoxi/Dropbox/cop530/Deque

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||

1
eighth

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||

1
nineth
realizes its full
initialized new array with x2 the space
successfully called grow

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||

4
fifth
fourth
third
second
first
sixth
seventh
eighth
nineth

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||
```

Sample output 2



```
/cygdrive/c/Users/Xiaoxi/Dropbox/cop530/Deque

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||
You just removed: fifth

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||
You just removed: ninth

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||
You just removed: fourth

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||
rightMost of the array
You just removed: eigth

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||
third
second
first
sixth
seventh
```



```
The size_of_queue size of the queue is: 8
There are 0 elements in the array
Index position 1 is empty.
Index position 2 is empty.
Index position 3 is empty.
Index position 4 is empty.
Index position 5 is empty.
Index position 6 is empty.
Index position 7 is empty.
Index position 8 is empty.



 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||

2
You have poped everything out of your array!!consider pushing in some new stuff

 Welcome to the program,

 Please enter your option
 || 0--push_front || 1--push back || 2--pop front || 3--pop back ||
 || 4--toStr() method|| 5--print your final queue ||

3
You have poped everything out of this queue...considering pushing in something?
```

Sample output 3