Author Name: Prof. Fnu Manju Muralidharan Priya for LLTemplate and Dat Truong for the implementation

Project Structure:

1. The Data class:
Name: TicketItem

Variables:

- String type reverseCode and personName

Constructor:

It takes 2 parameters with a code and a name to assign for the public variables inside the class.

Destructor:

Display once the ticketitem is deleted which will be called later in the program.

Print():

Display the name and the code respectively.

2. The Node:
Name: Node

Variables:

Type pointer which is value stored in the data class.

A node pointer to nextNode

Constructor:

Assigned value to the node using the passed in parameter

And set nextNode to nullptr

Destructor:

Set the value of the node to null

void print():

Print the value inside the node.

3. The Stack Container:
Name: LLStack
Variables:

A Node pointer call top will keep track of top of the stack.

Integer type stackSize that initialize for 0.

Constant integer type SMAXITEMS set to 5 to test for overflow condition.

Constructor:

Set the top is nullptr

Set stackSize start at 0

Pass the SMAXITEMS for checking overflow condition

Destructor:

Set the start at top Node then delete node till no more node available.

bool isEmpty():

Check if the top is nullptr which mean the stack is empty

bool isFull():

Check if the stackSize is smaller than SMAXITEMS if not, return true

int getSize():
return stackSize for later use in getting queue size

T* peek():

Return top node value

void push(T *item):
create a new node and set the data from parameter to the new node and push into the stack.

void pop():

Go to the top of the stack and take out the node and free its memories;

void print():

Set to the top of the stack and print the list from the top. So it appear as LIFO as in example below:

1->2->3->4 what appear will be 4,3,2,1 with 4 as the top of the stack.

4. The Implementation:
Name: StackQ

Variables:

2 LLStack<T> with one name enQStack and one is deQStack.

Integer type queueSize initialize at 0;

Const int QMAXITEMES to check for overflow condition

Constructor:

Initialize the two LLStack and set queueSize = 0

Destructor:

Delete both stack and print to show queue is deleted.

void enqueue(T* item)

Push the parameter to the node in the enQStack. If there is items in deQStack we will move it first to the enQStack to keep the order of the items then push in the new added item. Otherwise, just push item into the empty enQStack. Increase queues size at the end.

void deQueue():

Move entire enQStack to the deQStack then pop the top of the stack; decrease the queue size

T* peek():

Return the deQStack top if deQStack is occupied else move entire enQStack to deQStack to return the top value.

bool isEmpty():

Check if both enQStack and deQStack is empty.

bool isFull():

Checking if the queueSize is greater or equal to the QMAXITEMS if it is then true else false

int ticketQueueSize()

Return the sum of enQStack and deQStack.

void Print():

move item to deQStack if enQStack is occupied and then print the list. Otherwise, print the deQStack. This will keep the list order as FIFO. After print, move item back to enQStack to keep the list order.

void PrintBoth():

display the queue in both Stacks. Will let the user know which kind of queue is empty.

Problem Definition:

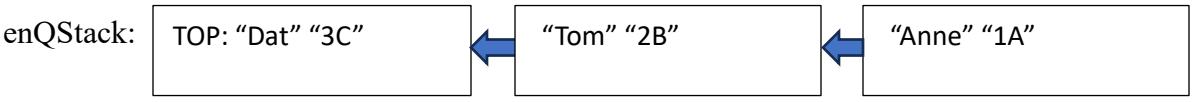
Note: these are also how the queue look when you use option to displays both queues after each call.

At the start of the program:

enQStack: TOP: nullptr

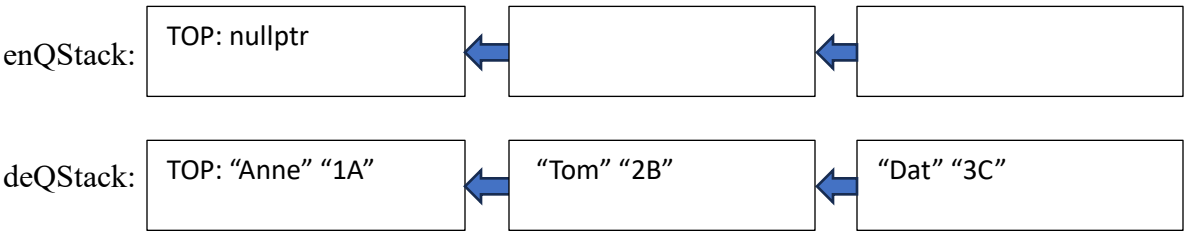
deQStack: TOP: nullptr

Call enqueue 3 times in the program:

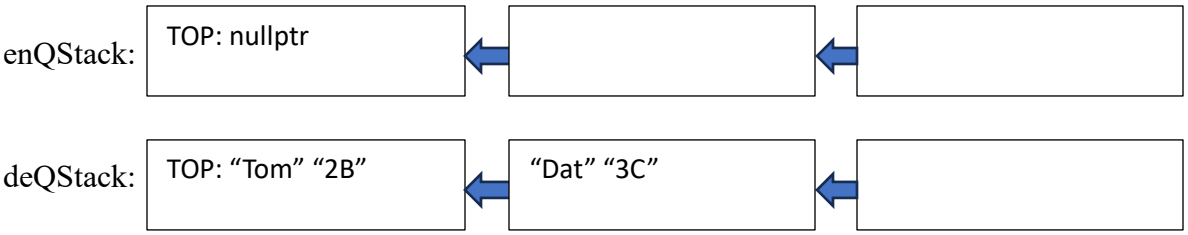


deQStack: TOP: nullptr

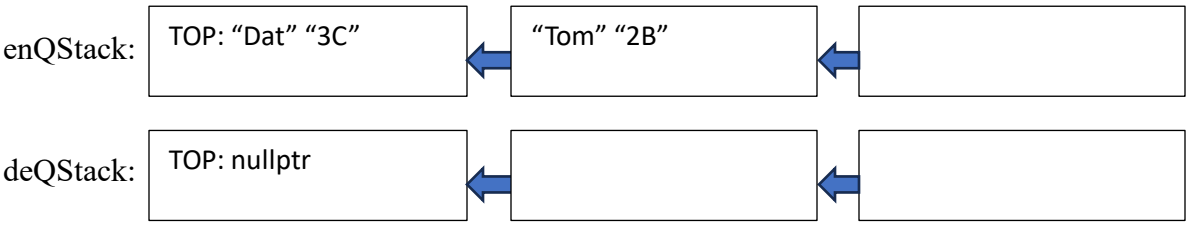
During Dequeue-ing in the program:



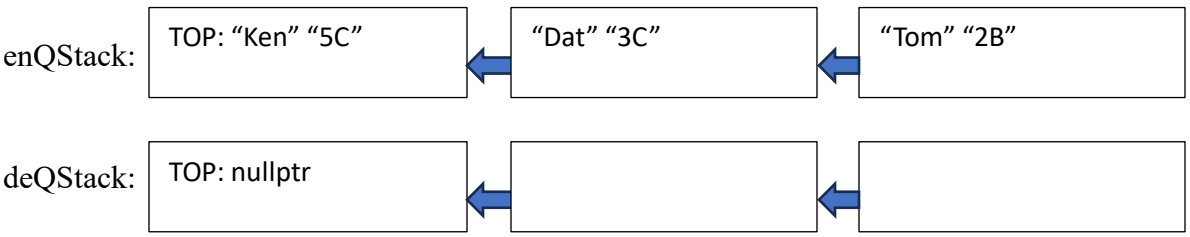
After call dequeue in the program:



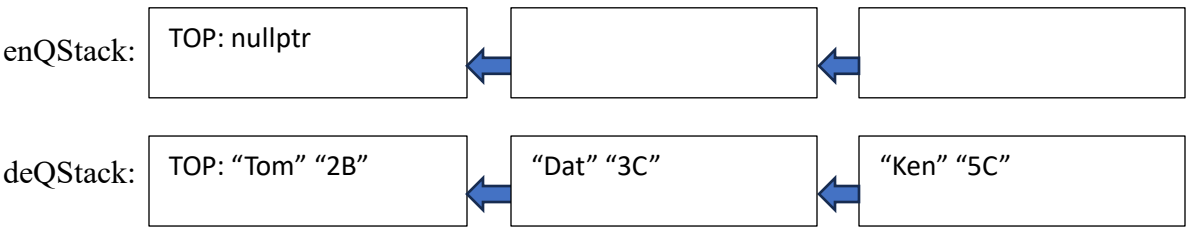
During call enqueue in the program:



After enqueue new value :



After Peek is called:



return top Value which is “Tom” “2B”