PrintItem():
float fileSize - float variable for file size
string fileName - string variable for file name
void print() - prints out file size and file name

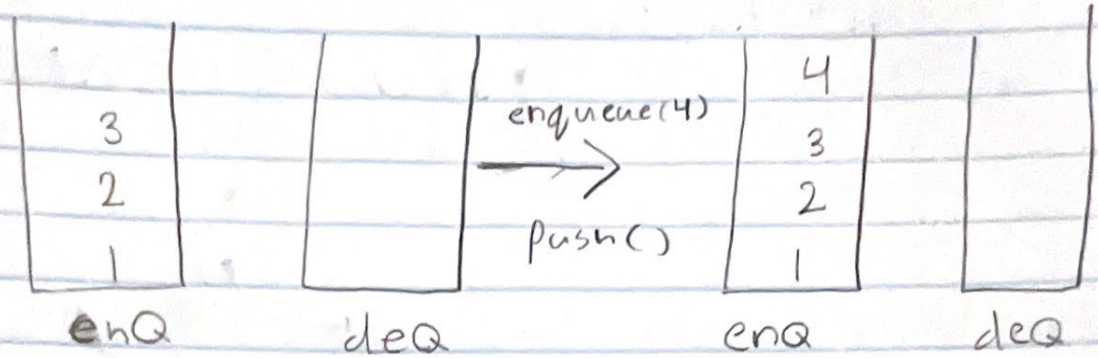
Node():
next - pointer to the next node
value - data stored in node
void print - value points to print

LLStack():
LLStack constructor - sets initial values of top to nullptr, stackSize to 0, and SMAXITEMS to 10.
LLStack constructor - initializes stack size to 1 and top.
top - top of the stack
int stacksize - stack length
int SMAXITEMS - max items in stack
int getStackSize() - returns stack size
int getTop() - returns top of stack
temp - pointer that points to new top of stack before
~LLStack destructor - updates top of stack when a node is deleted
void printList() - makes a temporary stack and uses push, pop, and peek to print the items of the stack before placing them back
isFull() - returns stack size is full
isEmpty() - returns stack size is empty
void push() - pushes item into stack
void pop() - pops item out of stack
void peek() - checks top item of stack without popping it

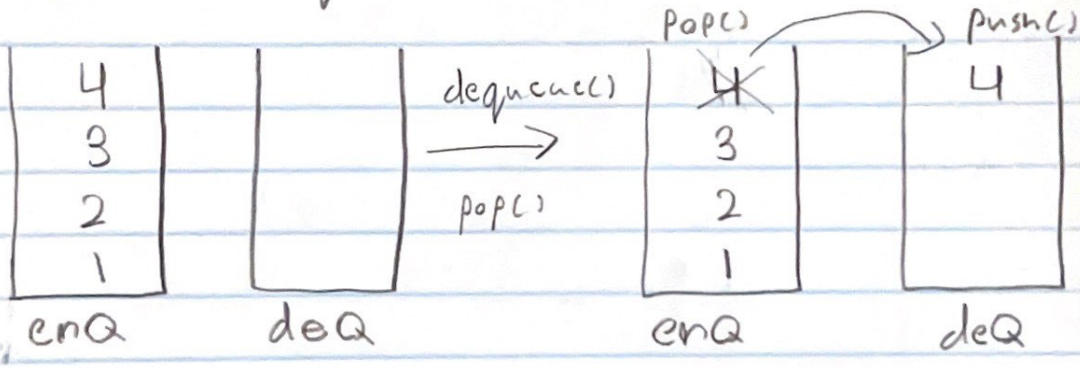
StackQ():
enQStack - pointer to stack
deQStack - pointer to stack
int queueSize - size of queue
const int QMAXITEMS - max number of items in queue
StackQ() - initializes enQStack, deQStack, queueSize, QMAXITEMS
~StackQ - deletes enqueue and dequeue
void enqueue() - checks if queue is full before adding an item
void dequeue() - checks if queue is empty. Also pops things out of enqueue and pushes them into dequeue.
peek- prints out items of queue

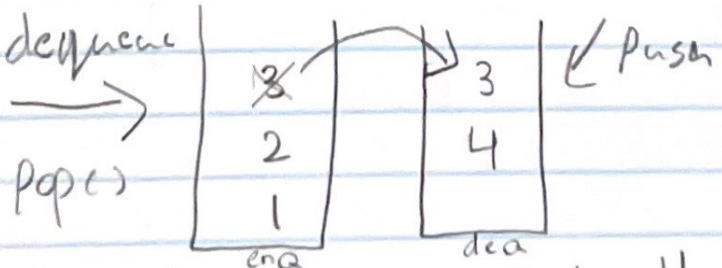
void printQueue() - prints items in queue
int getSize() - returns queue size
void printStacks() - prints elements of stack

main():
myQueue - instance of StackQ that helps pre-populate code with data
newItem - points to new item
peekedItem - points to peeked item



When enqueue is called, the new input will go to the top of the enQ queue. As seen with the example, when the number "4" is inputted, it becomes the top of the queue through the stack ADT's push().





When dequeue is called, the top of the enQ stack gets popped and placed into the deQ stack. As seen with the example, When you dequeue, 4 is popped & placed into deQ. When 3 is popped from enqueue & pushed into deQ, moving 4 down.