ECE250 – Project1
Dequeue
Design Document
Mengze Lyu ID:m6lyu
Jan 31st, 2020

1. Overview of Classes

Class:
Node

Description:
Save each node in the double linked list.

Member viarables:
num:save the key of node
next:save the pointer pointing to the next node
pre:save the pointer pointing to the previous node

Member functions:
No member functions

Class:
Queue

Description:
Save the list of nodes, provides operations to the list, including inserting elements, removing elements, printing, deleting the list etc.

Member variables:
head:pointer pointing at the start of the list
tail:pointer pointing at the end of the list
size:save the length of the list

Member functions:
Queue:the constructor
~queue:the destructor
enqueue_front: insert one integer to the front of the list
enqueue_back: insert one integer to the end of the list
dequeue_front: remove one integer from the front of the list
dequeue_back: remove one integer from the back of the list
clear: clear the content of the list
front: compare the first element in the list to a given integer
back: compare the last element in the list to a given integer

empty: report if the list is empty
sizeoflist: return the length of the list
print: output all the elements in the list two times, fist from front to end, then backwards

Node	Queue
num: int next: Node pre: Node	head: Node tail: Node size: int
No member functions	enqueue_front: void enqueue_back: void dequeue_front: void dequeue_back: void clear: void front: void back: void empty: void sizeoflist: void print: void

2. Constructors/Destructors

For class Node, next and pre would be NULL initially.
For class queue, head and tail would be equal, NULL, representing there is no element in the list. When destruct the list, all the elements would be delete and release the space. Head and tail will be set to NULL again.

3. Test Cases

Test1: inserting 10 elements all from front, compare front with correct number/wrong number, print the list, test if the list is empty, return the size of list
Test2: inserting 10 elements all from back, compare front with correct number/wrong number, print the list, test if the list is empty,return the size of list
Test3: inserting 10 elements, 5 from front, 5 from back, compare front with correct number/wrong number, print the list, test if the list is empty, return the size of list
Test4: base on any list from test1-3, remove all of the elements one by one from front, until it throws an error message
Test5: base on any list from test1-3, remove all of the elements from back, until it throws an error message
Test6: base on any list from test1-3, clear the list and check if the list is empty
Test7: initialize the list and return its size