Programming Assignment 4:

Queues Using Double-Linked Linear Lists

Name: Derek (劉哲瑋) Number: D1262032

I first define two classes named IQueue and Node respectively, as shown in the pictures below. Furthermore, I make IQueue to use the private part of Node by using “friend class IQueue;”

```cpp
#include "Node.h"
// Integer Queue Class
class IQueue {
    private:
        Node *head; // Point to head of the queue.
        Node *tail; // Point to tail of the queue.

    public:
        IQueue(); // Default constructor.

        int enqueue(int); // Enqueue operation.

        int dequeue(); // Dequeue operation.

        int getSize(); // Get the size of the queue.

        void printHeadToTail(); // Print the queue from head to tail.
};
```

```cpp
#ifndef NODE_HPP
#define NODE_HPP

class Node {
    friend class IQueue; // Class IStack can access the private data
        elements.

    private:
        int elem; // Data of a node.
        Node *prev; // Link of the previous node.
        Node *next; // Link of the next node node.

    public:
        Node(); // Default constructor. Set elem to 0 and prev and next
            to NULL.

        Node(int); // Constructor with data element. Set elem to the
            parameter value and prev and next to NULL.
};

#endif
```

Then, I define count, i, j, enq_c, deq_c using int data type, and que using IQueue. And I also use the random generator to get a trial count between 1 and 10. Next, I use for loops to perform enqueue operation and dequeue operation and to print the queues, as shown below.

```
16      for(i=0;i<count;i++){
17          cout<<endl<<">>>> Trial "<<i+1<<": enqueue and dequeue
                operations"<<endl;
18          enq_c=rand()%99+1;
19          for(j=0;j<enq_c;j++){
20              que.enqueue(rand() % 99 + 1);
21          }
22          cout<<"Enqueue "<<enq_c<<" elements to the quene."<<endl;
23          que.printHeadToTail();
24          do{
25              deq_c=rand()%99+1;
26          }while(deq_c>=que.getSize());
27          for(j=0;j<deq_c;j++){
28              que.dequeue();
29          }
30          cout<<endl<<endl<<"Dequeue "<<deq_c<<" elements to the
                quene."<<endl;
31          que.printHeadToTail();
32          cout<<"------------------------------------------------------------
                ---------"<<endl;
33      }
```

In “IQueue.cpp”, I define function “int enqueue(int);” and “int dequeue();” to perform enqueue operation and dequeue operation respectively, as shown below.

```
8  int IQueue::enqueue(int e) {//enqueue operation
9      Node* current = head;
10     Node* previous = tail;
11     Node* newNode;
12   int position=0;
13
14     if (current==NULL) {
15         newNode = new Node(e);
16         newNode->prev = newNode;
17         newNode->next = newNode;
18         head = newNode;
19         tail = newNode;
20         return position;
21     }
22     else{
23         newNode = new Node(e);
24         newNode->prev = previous;
25         previous->next = newNode;
26         newNode->next = current;
27         current->prev = newNode;
28         tail = newNode;
29         return position;
30     }
31 }


33 int IQueue::dequeue() {//dequeue operation
34     Node* current = head;
35     Node* previous = tail;
36     if (current==NULL) return -1;
37     if (current->next==current){
38         head = NULL;
39         tail = NULL;
40     }
41     else{
42         previous->next = current->next;
43         current->next->prev = previous;
44         head = current->next;
45     }
46     delete current;
47     return 0;
48 }
```

Next, I define the functions “int getSize();” by using a do-while loop to get the size of the queue and “void printHeadToTail();” by using a while loop to print the queue from head to tail, as shown below.

```
53 int IQueue::getSize() {
54     Node* current = head;
55     int size = 0;
56 
57     if (current==NULL) return size;
58     do {
59         size++;
60         current = current->next;
61     } while (current!=head);
62     return size;
63 }
64 
65 void IQueue::printHeadToTail() {
66     Node* current = head;
67     int position = 0;
68 
69     cout<<"Current queue size: "<<getSize()<<". Content of queue from
           head to tail:"<<endl;
70     if (current!=NULL) {
71         while (current->next!=head) {
72             cout.width(4);
73             cout<<current->elem;
74             if ((position+1)%20==0){ cout<<endl; }
75             current = current->next;
76             position++;
77         }
78         cout.width(4);
79         cout<<current->elem<<endl;
80     }
81 }
```

Finally, I define “Node()” and “Node(int)” in “Node.cpp”, as shown below.

```
5  Node::Node(){ elem=0; prev=NULL; next=NULL; }
6  Node::Node(int value){ elem=value; prev=NULL; next=NULL; }
```