자료구조를 구조하자

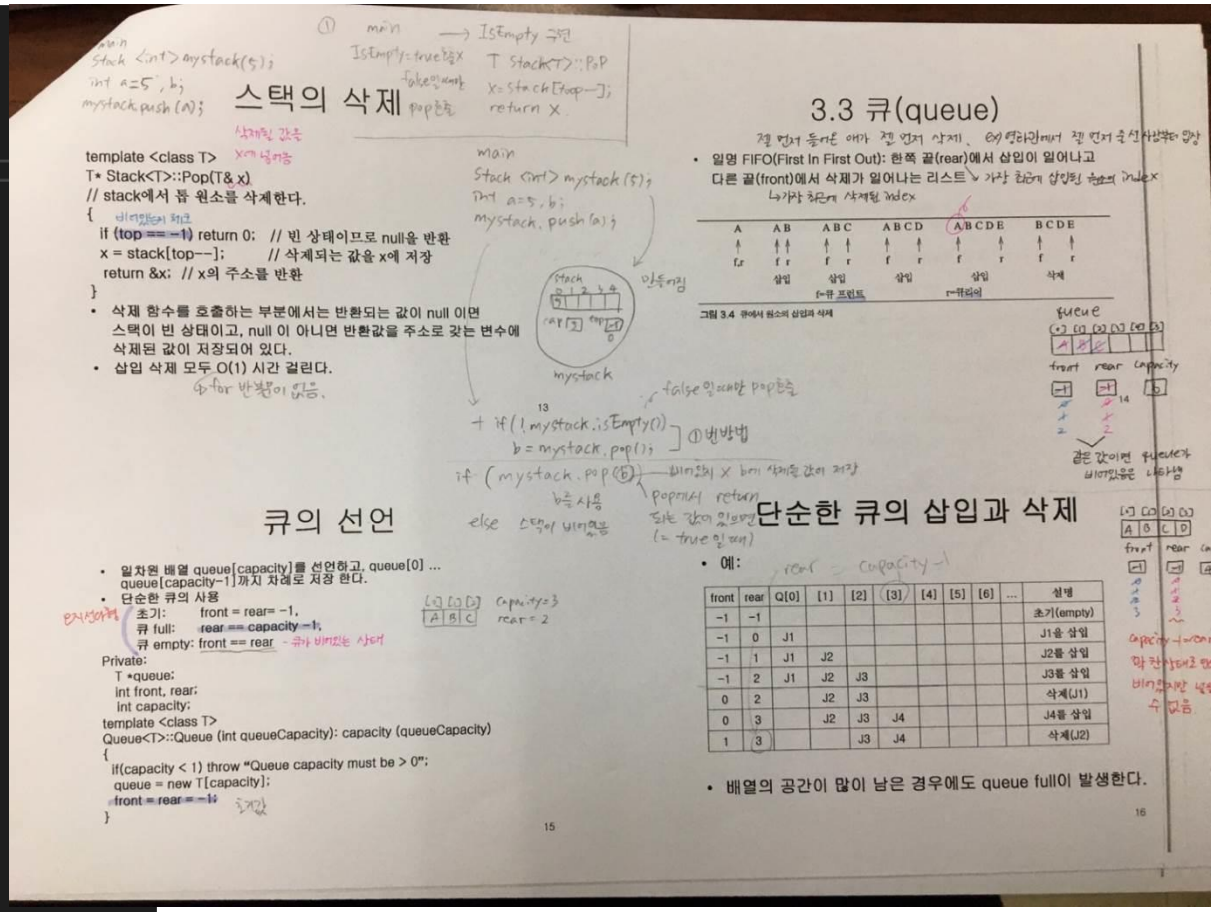
2019.10.13

32153180 이상민

32162436 신창우 32163006 이건욱
32164420 조정민 32164959 허전진

학습 내용

```cpp
#include <iostream>
using namespace std;

class ChainNode
{
    friend class LinkedQueue;
private:
    int data;
    ChainNode* link;
public:
    ChainNode(int element = 0, ChainNode * next = 0)  // 생성자
    {
        data = element; link = next;
    }
};

class LinkedQueue //큐 정의
{
private:
    ChainNode* front, * rear; //처음,끝을 나타냄
public:
    LinkedQueue() { front = rear = 0; };// 생성자
    void Push(const int&);        // 삽입 함수
    int Pop();                    // 삭제 함수
    void Print();                 // 출력 함수
};

//큐의 삽입
void LinkedQueue::Push(const int& e)
{
    if (front == 0) front = rear = new ChainNode(e, 0); //공백 큐
    else rear = rear->link = new ChainNode(e, 0);
    //노드를 삽입하고 rear를 수정함
}
//큐의 삭제
int LinkedQueue::Pop()
{
    if (front == 0) return 0;   // 공백 큐이므로 null을 반환
    ChainNode* delNode = front;
    front = front->link;        //앞의 노드를 제거
    delete delNode;             //노드 반환
    return 1;
}
//큐 내용 보기
void LinkedQueue::Print()
{
    ChainNode* p;
    for (p = front; p; p = p->link) //p가 널일 때 까지
        cout << p->data << " ";  //p의 data 출력
    cout << endl;
}
```



신창우

학습 내용

2-2 C → LinkedStack

```
1  class ChainNode {
2      friend class LinkedStack;   // 연결스택 클래스 프렌드 지정
3  private:
4      int data;           // 노드의 데이터 필드
5      ChainNode *link;    // 노드의 링크 필드
6  public:
7      ChainNode(int element = 0, ChainNode *next = 0) {   // 노드 생성자 함
8          data = element; link = next;
9      }
10 };
11 class LinkedStack {
12 private:
13     ChainNode *top;
14 public:
15     LinkedStack() { top = 0; }; // 연결스택 생성자 함수
16     void Push(const int&);      // 연결스택 삽입 함수
17     int *Pop(int&);             // 연결스택 삭제 함수
18 };
19 void LinkedStack::Push(const int& e) {
20     top = new ChainNode(e, top);    // 새 노드를 스택의 맨 위에 저장
21 }
22 // 스택 삭제
23 int* LinkedStack::Pop(int &x)
24 //스택에서 톱 노드를 삭제하고 값을 x에 저장한 후에 x의 주소를 반환
25 {
26     if (top == 0) return 0; //빈 스택이면 null을 반환
27     ChainNode* delNode = top;
28     x = top->data;          //톱 노드의 data 필드를 x에 저장
29     top = top->link;        //톱 위치를 다음 노드로 이동
30     delete delNode;         //노드 삭제
31     return &x;
32 }
```

▶ 큐(Queue)

FIFO(First In First Out) : 한쪽 끝(rear)에서 삽입이 일어나고 다른 끝(front)에서 삭제가 일어나는 리스트

✓ 원형 큐의 특성과 삽입

• 원형 큐의 동작을 위해 front와 rear를 시계 방향으로 이동시킴

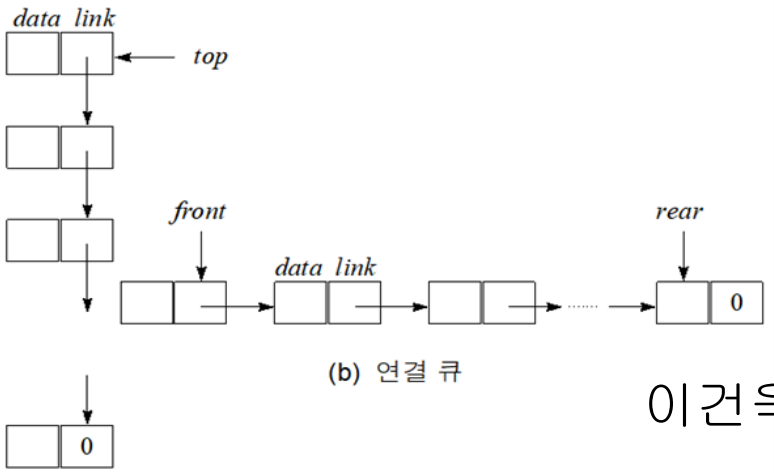
```
if (rear == capacity - 1) rear = 0;
else rear++;
// 위 if 문은 rear = (rear + 1) % capacity와 같음
```

• front == rear 이면 empty

• 큐가 full인 경우에도 front == rear 이므로 capacity - 1개만의 데이터를 삽입한다. 즉 front == (rear + 1) % capacity 이면 full

• 연결 스택과 큐

> 스택과 큐를 연결 리스트로 구현하는 것



(b) 연결 큐

(a) 연결 스택

이건욱

학습 내용

32164420 조정민.cpp

기타 파일

```cpp
#include<iostream>
using namespace std;

class LinkedStack;              // 연결스택 전방선언
class LinkedQueue;              // 연결큐 전방선언
class ChainNode
{
    friend LinkedStack;         // LinkedStack freind 지정
    friend LinkedQueue;         // LinkedQueue freind 지정
private:
    int data;                   // data field
    ChainNode* link;            // link field
public:
    ChainNode(int element = 0, ChainNode* next = 0)   // (
    {
        data = element;
        link = next;
    }
};
class LinkedStack
{
private:
    ChainNode* top;
public:
    LinkedStack()               // LinkedStack 생성자
    {
        top = 0;
    }
    void Push(const int&);      // LinkedStack 삽입함수
    int* Pop(int&);             // LinkedStack 삭제함수
    void Print();               // LinkedStack 출력함수
};
```

32164420 조정민.cpp

기타 파일

ChainNode

```cpp
void LinkedStack::Push(const int& x)
{
    top = new ChainNode(x, top);     // 새 노드를 top에 저장
}
int* LinkedStack::Pop(int& x)
{
    if (top == 0)                    // 스택이 비어있을 경우
        return 0;
    ChainNode* temp = top;
    x = top->data;                   // top 노드의 data field를 x에 저장
    top = top->link;                 // top을 다음 노드로 이동
    delete temp;                     // 노드 삭제
    return &x;                       // x의 주소 반환
}
void LinkedStack::Print()
{
    ChainNode* p = top;
    if (top == 0)                    // 스택이 비어있을 경우
    {
        cout << "비어있는 스택" << endl;
        return;
    }
    else
    {
        cout << "스택(LIFO순) : ";
        for (p; p->link; p = p->link)   // p가 top부터 다음 노드로 이동
        {
            cout << p->data << ' ';     // p의 data field 출력
        }
        cout << p->data << endl;          // 마지막 data field 출력
    }
}
```

조정민

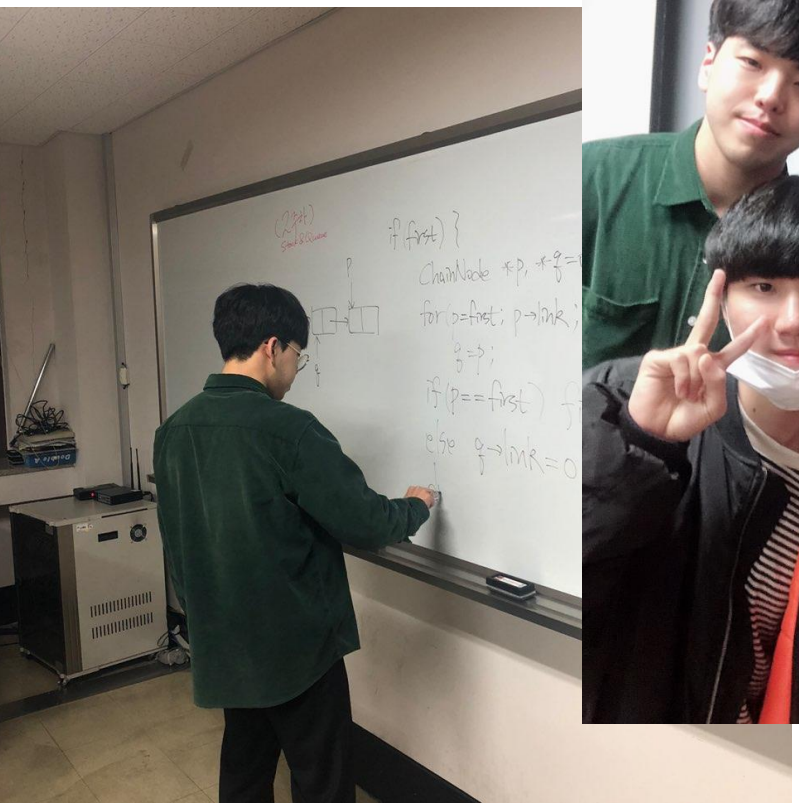
학습 내용

```
class LinkedStack {
private:
        ChainNode* top;
public:
        LinkedStack() { top = 0; }; //생성자
        void Push(const int&); //스택 삽입 함수
        int* Pop(int&); //스택 삭제 함수
        void Print(); //스택 출력 함수
};

void LinkedStack::Push(const int& e)
{
        top = new ChainNode(e, top);
}
```

```
edStack::Pop(int& x)

if (top == 0) return 0; //빈 스택이면 0 반환
ChainNode* delNode = top;
x = top->data; //톱 노드의 data필드 x에 저장
top = top->link; //top 위치 -> 다음 노드
delete delNode; //노드 삭제
return &x; //x 주소 반환


kedStack::Print()

if (top == 0)
        cout << "스택이 비었습니다." << endl;
else { //스택이 있으면 LIFO순 출력
        ChainNode* temp = top;

        cout << "스택(LIFO순) : ";
        while (temp != 0)
        {
                cout << temp->data << " ";
                temp = temp->link;
        }
}
fgetc(stdin);
//fflush(stdin);
getchar();
}
```

허전진

활동 사진



참고자료

https://www.youtube.com/watch?v=DsZHDmth6Pc

https://www.youtube.com/watch?v=BdsyG5yP1cQ