

(훈련반2) Level126

링크드리스트의 세 번째 시간 입니다.

이번 시간에는 링크드리스트와 더불어 **스택과 큐**에 대해 배우고, 직접 구현합니다.

링크드리스트 그림을 그려가면서, 노드를 추가하고, 삭제하는 연습을 합니다.

Level126 힙에 노드 생성

문제 1번 [\[숙제 목록보기\]](#)

아래 소스코드는 힙에 노드를 4개 생성한 코드입니다.

3을 입력하면 3의 배수로 4개 노드를 생성해 주세요.



이제 while문을 이용해서 만들어진 모든 노드를 출력 해 주세요.

[힌트]

```
cin >> input;

for (int i = 1; i<=4; i++) {

    addNode(i * input);

}
```

입력 예제

3

출력 결과

3 6 9 12

```

#include <iostream>
using namespace std;

const int loopCount = 4;

template <typename T>
struct Node
{
public:
    Node()
        : val(0)
        , pNext(nullptr)
    {
    }
    Node(T data)           // 오버로딩
        : val(data)
        , pNext(nullptr)
    {
    }
public:
    T val;                 // 값을 넣어줄 멤버변수
    Node* pNext;           // 다음 노드의 주소를 저장하는 멤버포인터변수
};

Node<int>* head = nullptr;
Node<int>* tail = nullptr;

void addNode(int data)
{
    if (head == nullptr)
    {
        head = new Node<int>(data);
        tail = head;
    }
    else
    {
        tail->pNext = new Node<int>(data);
        tail = tail->pNext;
    }
}

```

```
int main(void)
{
    int input = 0;
    cin >> input;

    for (int i = 1; i < loopCount + 1; ++i)
    {
        // 노드 생성
        // 각 노드는 순서대로 입력받은 값의 배수를 값으로 가지고 있다.
        addNode(input * i);
    }

    Node<int>* temp = head;
    while (true)
    {
        cout << temp->val << " ";
        temp = temp->pNext;
        if (temp->pNext == nullptr)
        {
            cout << temp->val;
            break;
        }
    }
    return 0;
}
```

Level26 노드 따라가기

문제 2번 [\[속제 목록보기\]](#)

숫자 하나를 입력받으세요 (11 ~ 36 까지 숫자)

입력받은 번호에 해당하는 문자 부터 노드 4개만 연결시켜주세요.

11 : A

12 : B

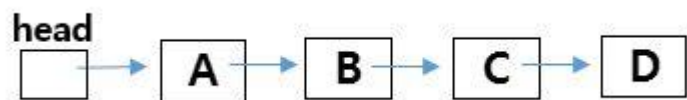
13 : C

14 : D

...

36 : Z

만약 11을 입력받았다면, 아래와 같이 구성하면 됩니다.



그리고 for문을 돌려 모든 노드를 출력 해 주세요.

입력 예제

11

출력 결과

ABCD

```

#include <iostream>
using namespace std;

const int loopCount = 4;
const int offset = 54;

template <typename T>
struct Node
{
public:
    Node()
        : val(0)
        , pNext(nullptr)
    {
    }
    Node(T data)           // 오버로딩
        : val(data)
        , pNext(nullptr)
    {
    }
public:
    T val;                 // 값을 넣어줄 멤버변수
    Node* pNext;           // 다음 노드의 주소를 저장하는 멤버포인터변수
};

Node<char*>* head = nullptr;
Node<char*>* tail = nullptr;

void addNode(int data)
{
    data += offset;
    if (head == nullptr)
    {
        head = new Node<char*>(data);
        tail = head;
    }
    else
    {
        tail->pNext = new Node<char*>(data);
        tail = tail->pNext;
    }
}

```

```

int main(void)
{
    int input = 0;
    cin >> input;          // 입력 받을 수 있는 값의 범위 : 11 ~ 36
    // 11 : A ~ 36 : Z
    // ascii -> A : 65 ~ Z : 90
    for (int i = 1; i < loopCount + 1; ++i)
    {
        // 노드 생성
        addNode(input);
        ++input;
    }

    Node<char>* temp = head;
    while (true)
    {
        cout << temp->val;
        temp = temp->pNext;
        if (temp->pNext == nullptr)
        {
            cout << temp->val;
            break;
        }
    }

    return 0;
}

```

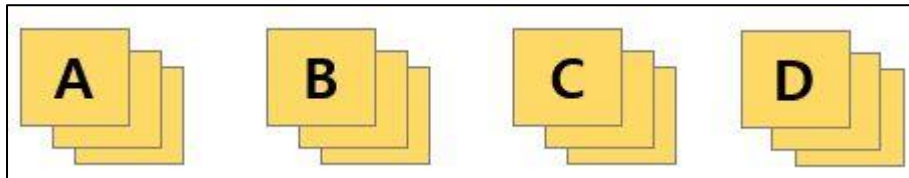
Level26 중복 없는 카드 뽑기

문제 3번 [[속제](#) [목록보기](#)]

네 종류의 카드가 있습니다. (A,B,C,D)

몇 장을 뽑을지 입력받아주세요

n장을 뽑으려고 하는데 중복 없이 뽑는 경우를 출력 해주세요.



만약 2를 입력했다면

AB

AC

AD

BA

BC

...

DC

만약 3을 입력했다면

ABC

ABD

ACB

...

DCB

가지치기 힌트

via 전역배열을 이용해서 중복 판단 가능

입력 예제

2

출력 결과

AB

AC

AD

BA

BC

BD

CA

CB

CD

DA

DB

DC


```

#include <iostream>
using namespace std;

const int cardCnt = 4;
char cards[cardCnt] = { 'A', 'B', 'C', 'D' };
const int initLevel = 0;
int input = 0;
char bucket[10] = {};
bool isVisit[cardCnt] = {};
int cnt = 0;

void recursive(int level)
{
    if (level == input)
    {
        cout << bucket << endl;
        ++cnt;
        return;
    }

    for (int i = 0; i < cardCnt; ++i)
    {
        if (!isVisit[i])
        {
            bucket[level] = cards[i];
            isVisit[i] = true;
            recursive(level + 1);
            bucket[level] = '\0';
            isVisit[i] = false;
        }
    }
}

int main(void)
{
    cin >> input;
    recursive(initLevel);
    cout << "total count : " << cnt;

    return 0;
}

```

Level26 노드 생성과 연결

문제 4번 [\[속제 목록보기\]](#)

숫자 n 을 입력 받고, n 개만큼 노드를 만들고 연결 해주세요.

들어가는 값은 순차적으로 A, B, C, D... / 1, 2, 3, 4... 입니다.

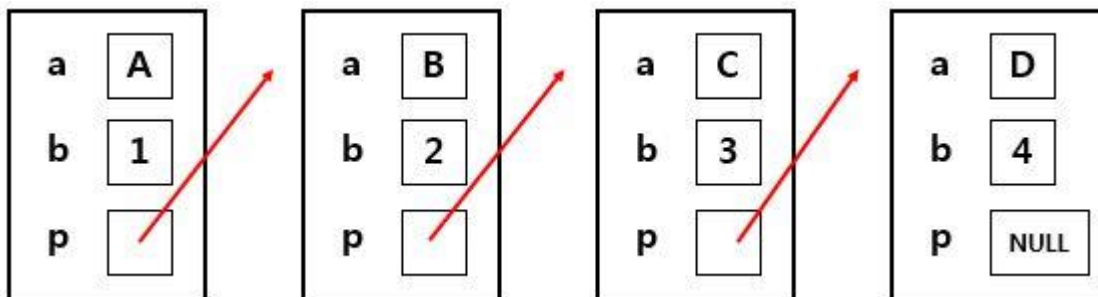
노드를 구성한 후

대문자는 for문으로 출력하고

숫자들은 while로 모두 출력 해 주세요.

ex1)

입력: 4



출력: A B C D

1 2 3 4

ex2)

입력: 3

출력: A B C

1 2 3

입력 예제

4

출력 결과

A	B	C	D
---	---	---	---

1	2	3	4
---	---	---	---

```

#include <iostream>
using namespace std;

int loopCount = 0;
const char firstChVal = 'A';
const int firstNumVal = 1;

template <typename T, typename G>
struct Node
{
public:
    Node()
        : alpha('\0')
        , num(0)
        , pNext(nullptr)
    {
    }
    Node(T data1, G data2)           // 오버로딩
        : alpha(data1)
        , num(data2)
        , pNext(nullptr)
    {
    }
public:
    T alpha;           // 문자를 입력받는 변수
    G num;             // 숫자를 입력받는 변수
    Node* pNext;       // 다음 노드의 주소를 저장하는 멤버포인터변수
};

Node<char, int>* head = nullptr;
Node<char, int>* tail = nullptr;

void addNode(char data1, int data2)
{
    if (head == nullptr)
    {
        head = new Node<char, int>(data1, data2);
        tail = head;
    }
    else
    {
        tail->pNext = new Node<char, int>(data1, data2);
        tail = tail->pNext;
    }
}

```

```

int main(void)
{
    cin >> loopCount;

    int data1 = firstChVal;
    int data2 = firstNumVal;
    for (int i = 1; i < loopCount + 1; ++i)
    {
        // 노드 생성
        addNode(data1, data2);
        ++data1;
        ++data2;
    }

    Node<char, int>* temp = head;
    while (true)
    {
        cout << temp->alpha << " ";
        temp = temp->pNext;
        if (temp->pNext == nullptr)
        {
            cout << temp->alpha;
            break;
        }
    }

    cout << endl;

    temp = head;
    while (true)
    {
        cout << temp->num << " ";
        temp = temp->pNext;
        if (temp->pNext == nullptr)
        {
            cout << temp->num;
            break;
        }
    }

    return 0;
}

```

Level26 인큐디큐 인디인디

문제 5번 [숙제 목록보기]

링크드리스트로 Queue를 만들어주세요.

입력의 첫번째 숫자는 Queue에 넣을 문자 갯수(Enqueue)

두번째 숫자는 Dequeue 할 갯수 입니다.

그 다음 문자들은 Queue에 넣을 문자들 입니다.

예로들어

5 2
A B C G K

를 입력 받았다면

큐에

A B C G K

를 넣은 뒤 2개를 Dequeue 해야하니까

큐상태는

C G K

가 됩니다.

최종 큐 상태를 출력 해주세요.

ex)

3 1
A B C

→ [출력] B C

ex)

6 2
B C D A T K

→ [출력] D A T K

입력 예제

3 1
A B C

출력 결과

B C

```

#include <iostream>
#define _CRTDBG_MAP_ALLOC
#include <stdlib.h>
#include <crtDBG.h>
using namespace std;

int queueSize = 0;
int dequeueSize = 0;

template <typename T>
class Queue
{
public:
    Queue() // 생성자
        : head(nullptr)
        , tail(nullptr)
    {
    }
    Queue(T _data)
        : head(_data)
        , tail(nullptr)
    {
    }

    void push(T _data)
    {
        if (head == nullptr)
        {
            head = new Node(_data);
            tail = head;
        }
        else
        {
            tail->pNext = new Node(_data);
            tail = tail->pNext;
        }
    }

    T pop()
    {
        Node* temp = nullptr; // 임시 포인터 변수
        T ret = NULL;         // 반환 값
        temp = head;
        ret = temp->data;
    }
};

```

```

        head = temp->pNext;
        delete temp;
        temp = nullptr;
        return ret;
    }

    void printAllNode()
    {
        Node* temp = head;
        while (temp != tail)
        {
            cout << temp->data << " ";
            temp = temp->pNext;
        }
        cout << tail->data << endl;
    }

    void delNode()
    {
        Node* temp1 = head;
        Node* temp2 = nullptr;
        while (temp1 != tail)
        {
            temp2 = temp1->pNext;
            temp1->pNext = nullptr;
            delete temp1;
            temp1 = temp2;
        }
        delete tail;
    }

public:
    struct Node
    {
        Node() // 생성자
            : data(0)
            , pNext(nullptr)
        {
        }
        Node(T _data)
            : data(_data)
            , pNext(nullptr)
        {
        }
    }

```



```

        T data;
        Node* pNext;
    };
    Node* head;
    Node* tail;
};

int main(void)
{
    Queue<char>* que = new Queue<char>();
    char* str = nullptr;
    cin >> queueSize >> dequeueSize;
    str = new char[queueSize];
    for(int o = 0; o < queueSize; ++o)
        cin >> str[o];

    // queue에 데이터 넣기
    char data = '\0';
    for (int i = 0; i < queueSize; ++i)
    {
        data = str[i];
        que->push(data);
    }

    cout << "Queue에 들어 있는 값 : ";
    que->printAllNode();

    // deque
    cout << "Queue에서 뺀 값 : ";
    for (int k = 0; k < dequeueSize; ++k)
        cout << que->pop() << " ";
    cout << endl;

    cout << "Queue에 들어 있는 값 : ";
    que->printAllNode();

    delete[] str;
    que->delNode();
    delete que;

    _CrtDumpMemoryLeaks();
    return 0;
}

```

Level26 숫자 parsing하기

문제 6번 [[숙제](#) [목록보기](#)]

숫자와 문자가 섞인 문장을 입력 받습니다.

이때 문장 속에서 숫자는 하나만 존재합니다. (최대 15글자)

ex1) ATP**1326**TTA

ex2) PPPT**756**

이런 숫자와 문자가 섞인 한문장에서 숫자만 파싱하여 5를 더한 숫자를 출력하세요.

아래 예제를 참고하세요.

ex) 1999POW
→ 1999 + 5 = 2004 출력
ex) ATP1326TTA
→ 1326 + 5 = 1331 출력
ex) PPPT756
→ 756 + 5 = 761 출력

[Tip]

parsing(파싱) : 문장에서 내가 원하는 값을 뽑아내는 처리를 뜻합니다.

입력 예제

1999POW

출력 결과

2004

```

#include <iostream>
using namespace std;

/*
    [ASCII]
    'A' ~ 'Z' = 65 ~ 90
    '0' ~ '9' = 48 ~ 57
*/

void Parsing(char* _data, char* _par)
{
    int idx1 = 0;    // 문자열에서 숫자가 시작되는 인덱스
    while (true)
    {
        char value = _data[idx1];
        if (value >= 65 && value <= 90)
            ++idx1;
        else
            break;
    }

    int idx2 = idx1; // 문자열에서 다시 문자가 시작되는 인덱스 or 문자열의 끝 부분
    while (true)
    {
        char value = _data[idx2];
        if (value >= 48 && value <= 57)
            ++idx2;
        else
            break;
    }

    int idx = 0;
    for (int i = idx1; i < idx2; ++i)
    {
        idx = i - idx1;
        _par[idx] = _data[i];
    }
}

```

```
int main(void)
{
    char input[20] = {};
    char par[20] = {};
    cin >> input;
    Parsing(input, par);

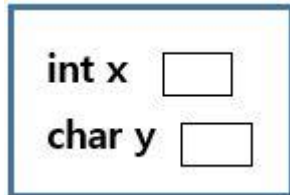
    int length = strlen(par);
    int weight = 1;
    int result = 0;
    for (int k = length - 1; k >= 0; --k)
    {
        int val = par[k] - 48;
        result += val * weight;
        weight *= 10;
    }
    cout << result + 5;

    return 0;
}
```

Level26 구조체 QUEUE

문제 7번 [숙제 목록보기]

한 노드가 다음과 같은 구조체로 된 큐를 배열로 구현해 주세요.

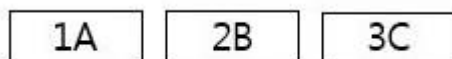


```
Node queue[10];
```

이제 입력 개수에 따라 입력이 주어 집니다.



이렇게 입력을 받았다면 큐에 아래와 같이 채워집니다.



모두 채운 후, 모든 값을 pop 해주세요.

이제 pop을 할때마다 나오는 값을 출력 해주세요.

입력 예제

```
4  
1 A  
2 B  
3 C  
4 D
```

출력 결과

1 A

2 B

3 C

4 D

```

#include <iostream>
#define _CRTDBG_MAP_ALLOC
#include <stdlib.h>
#include <crtDBG.h>
using namespace std;

class MyQueue
{
public:
    MyQueue()
        : que{}
        , headIdx(0)
        , endIdx(0)
    {
    }

    struct Node
    {
    public:
        Node()
            : x(0)
            , y('\0')
        {
        }
        int x;
        char y;
    };

    void push(int num, char ch)
    {
        que[endIdx].x = num;
        que[endIdx].y = ch;
        ++endIdx;
    }

    void pop()
    {
        if (headIdx < endIdx)
        {
            cout << que[headIdx].x << " ";
            cout << que[headIdx].y << endl;
            ++headIdx;
        }
        else
            cout << "큐에 들어있는 값이 없습니다.";
    }
}

```

```

private:
    Node que[20];
    int headIdx;
    int endIdx;
};

int input = 0;
int* arrNum = nullptr;
char* arrCh = nullptr;

int main(void)
{
    cin >> input;
    MyQueue* que = nullptr;
    que = new MyQueue();
    arrNum = new int[input];
    arrCh = new char[input];

    // 값 입력받기
    for (int i = 0; i < input; ++i)
        cin >> arrNum[i] >> arrCh[i];

    for (int k = 0; k < input; ++k)
        que->push(arrNum[k], arrCh[k]);

    for (int o = 0; o < input; ++o)
        que->pop();

    delete que;
    delete[] arrNum;
    delete[] arrCh;

    _CrtDumpMemoryLeaks();
    return 0;
}

```


Level26 n번 Enqueue & Dequeue

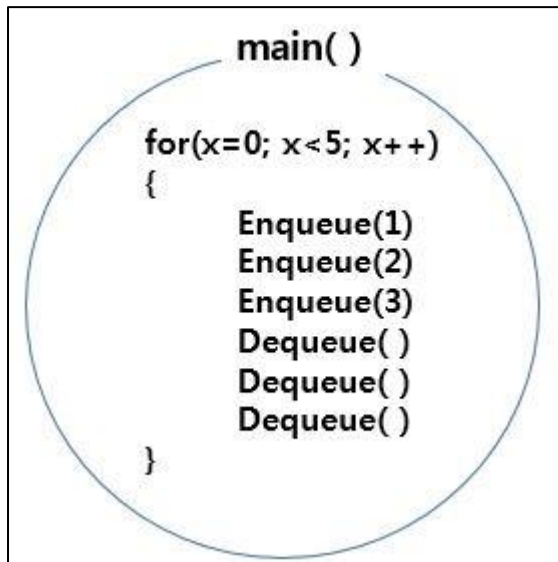
문제 8번 [[숙제](#) [목록보기](#)]

10칸짜리 큐를 만들어주세요.

숫자 하나를 입력받고, Enqueue 3회 / Dequeue 3회 반복하는 소스코드를 작성 해 주세요.

Dequeue할때마다 숫자를 출력하면 됩니다.

만약 n이 5라면 아래와 같은 main 함수가 동작되도록 구현 하시면 됩니다.



입력 예제

5

출력 결과

123123123123123

```

#include <iostream>
// #define _CRTDBG_MAP_ALLOC
// #include <stdlib.h>
// #include <crtdbg.h>
using namespace std;

const int queSize = 10;
int input = 0;

class MyQueue
{
public:
    MyQueue()
        : que{}
        , headIdx(0)
        , endIdx(0)
    {
    }

    struct Node
    {
    public:
        Node()
            : num(0)
        {
        }
        int num;
    };

    void Enqueue(int _num)
    {
        que[endIdx].num = _num;
        ++endIdx;
    }

    void Dequeue()
    {
        if (headIdx < endIdx)    // Queue에 값이 들어있는지 확인
        {
            // 값을 뺀다.
            int val = que[headIdx++].num;
            // 배열을 한칸씩 앞으로 땡긴다.
            int len = endIdx - headIdx;
            int i = 0;
            for (; i < len; ++i)

```

```

        {
            que[i] = que[i + 1];
        }
        que[i].num = 0;
        --headIdx;
        --endIdx;
        // 빼준 값을 출력한다.
        cout << val;
    }
    else
        cout << "Queue에 들어있는 값이 없습니다.";
}

private:
    Node que[queSize];
    int headIdx;    // 큐에서 값이 있는 제일 첫번째 노드
    int endIdx;     // 새로운 값이 들어갈 노드의 위치 / 마지막 값이 있는 노드의
    뒤
};

int main(void)
{
    cin >> input;
    MyQueue* que;
    que = new MyQueue();

    for (int x = 0; x < input; ++x)
    {
        que->Enqueue(1);
        que->Enqueue(2);
        que->Enqueue(3);
        que->Dequeue();
        que->Dequeue();
        que->Dequeue();
    }

    delete que;

    // _CrtDumpMemoryLeaks();
    return 0;
}

```

Level26 입력받은 곳 한줄 출력

(난이도 : ★☆☆)

문제 9번 [[숙제](#) [목록보기](#)]

다음 2차배열을 하드코딩 해주세요.

	A	B	C	D
0	3	5	1	4
1	2	2	1	1
2	0	1	2	3
3	3	1	3	1

문자로 된 숫자1개 또는 문자 1개를 입력 받으세요.

해당되는 한줄을 출력 하면 됩니다.

ex)

1을 입력 받으면 2211 출력

0을 입력 받으면 3514 출력

A를 입력 받으면 3203 출력

입력 예제

1

출력 결과

2211

```

#include <iostream>
using namespace std;

const int arrSize = 4;
const int offsetCh = 65;
const int offsetNum = 48;

int arr2D[arrSize][arrSize] =
{
    3,5,1,4,
    2,2,1,1,
    0,1,2,3,
    3,1,3,1
};

void printVer(int _input)
{
    _input -= offsetCh;
    for (int y = 0; y < arrSize; ++y)
        cout << arr2D[y][_input];
}

void printHor(int _input)
{
    _input -= offsetNum;
    for (int x = 0; x < arrSize; ++x)
        cout << arr2D[_input][x];
}

int main(void)
{
    // ascii -> 'A' = 65, 'D' = 68
    // ascii -> '0' = 48, '3' = 51
    char input = '\0';
    cin >> input;

    cout << "input 값 : " << input << endl;
    cout << "출력 값 : ";
    if (input >= 'A' && input <= 'D')           // 세로줄 출력
        printVer(input);
    else                                           // 가로줄 출력
        printHor(input);
    return 0;
}

```

Level26 꺾쇠 맞추기

문제 10번 [[숙제](#) [목록보기](#)]

꺾쇠 < 와 >, 숫자로 이루어진 한 문장을 입력 받으세요.

꺾쇠가 열리면 ('<') 이후에 꺾쇠가 닫혀야 합니다.('>')

정상적으로 꺾쇠가 열리고 닫혔는지 판단하는 프로그램을 작성하세요.

정상이면 "정상", 비정상이면 "비정상" 출력. (최대 20글자)

ex) <35<6<912>>10> => 정상

ex) 56<7>>65 => 비정상

ex) >>15<< => 비정상

ex) 612<>7<>91 => 정상

입력 예제

<35<6<912>>10>

출력 결과

정상

```

#include <iostream>
using namespace std;

// 최대 20글자
// <, > 꺾쇠 확인하기 (짝이 맞아야한다.)

const int length = 20;

int main(void)
{
    char input[length + 1];
    cin >> input;

    int lBracketCnt = 0;    // <
    int rBracketCnt = 0;    // >
    bool isSafe = false;
    int inputSize = (int)strlen(input);
    for (int i = 0; i < inputSize; ++i)
    {
        if (input[i] == '<')
            ++lBracketCnt;
        else if (input[i] == '>')
            ++rBracketCnt;

        if (lBracketCnt < rBracketCnt)
            break;
    }

    if (lBracketCnt == rBracketCnt)
        cout << "정상";
    else
        cout << "비정상";

    return 0;
}

```

Level26 메모장 만들기 (난이도 : ★★★)

문제 11번 [[숙제 목록보기](#)]

Text 한문장을 입력 받으세요.

그리고 커서 위치를 입력 받으세요.

예를 들어 ABCDEF와 커서위치 2를 입력 받았다면



CMD는 세가지 입니다.

L: Left Cursor

R: Right Cursor

D: Delete Cursor

이제 CMD 4개를 입력받고 CMD에 따라 처리를 해주세요.

만약 RRLD를 입력 받았다면

오른쪽 2칸, 왼쪽 1칸 커서를 이동시키고, Delete를 한번 수행하면 되므로



와 같은 상태가 됩니다.

커서 위치는 3번 index 글자 앞에 있습니다.

명령어를 수행하고 커서가 있는 위치를 출력 해주세요.



입력 예제

ABCDEF

2

RRLD

출력 결과

3

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

const int inputSize = 20;
const int cmdSize = 4;          // CMD 종류 : L,R,D

int main(void)
{
    char input[inputSize] = {};
    int cursorPos = 0;
    cin >> input >> cursorPos;
    char cmdArr[cmdSize + 1] = {};
    cin >> cmdArr;
    for (int i = 0; i < cmdSize; ++i)
    {
        char cmd = cmdArr[i];
        if (cmd == 'L')
            --cursorPos;
        else if (cmd == 'R')
            ++cursorPos;
    }

    cout << cursorPos;

    return 0;
}
```

Level26 척척박사님





문제 12번 [숙제 목록보기]

큐를 이용해서 풀어보세요.

척척박사님은 B, I, A, H 슈퍼영웅들 중 출동할 사람을 순서대로 뽑아야 합니다.

척척박사님은 항상 영웅B를 시작으로 다섯번째 사람을 선택합니다.

반복적으로 다섯번째 사람을 선택했을 때, 출동하는 영웅들의 순서를 출력 하세요.

B	I	A	H
			

B	I	A	H
①	②	③	④
⑤			
→ 첫번째 출동자는 배동맨			

B	I	A	H
	①	②	③
	④	⑤	
→ 두번째 출동자는 캡틴A			

B	I	A	H
			①
	②		③
	④		⑤
→ 세번째 출동자는 헐크H			

출동순서 결과: B A H I

[힌트]

4번 (Dequeue
Enqueue 후에 1회 Dequeue하고 출력 하면

항상 다섯번째 사람이 출력 됩니다.

출력 결과

B A H I

```

#include <iostream>
#define _CRTDBG_MAP_ALLOC
#include <stdlib.h>
#include <crtDBG.h>

using namespace std;
const int people = 4;
char names[people] = { 'B', 'I', 'A', 'H' };
const int selectCnt = 5;
bool isSelected[people] = {};

class MyQueue
{
public:
    MyQueue()
        : head(nullptr)
        , tail(nullptr)
    {
    }
    struct Node
    {
    public:
        Node()
            : name('/0')
            , pNext(nullptr)
        {
        }
        Node(char _data)
            : name(_data)
            , pNext(nullptr)
        {
        }
        char name;
        Node* pNext;
    };
    void Enqueue(char _data)
    {
        if (head == nullptr)
        {
            head = new Node(_data);
            tail = head;
        }
        else
        {
            tail->pNext = new Node(_data);
            tail = tail->pNext;
        }
    }

```

```

    }
    char Dequeue()
    {
        Node* temp = head;
        char val = temp->name;
        head = temp->pNext;
        delete temp;
        return val;
    }
    /*void delNode()
    {
        if (head == nullptr)
        {
            cout << "비었다.";
        }
        else
        {
            cout << "안 비었다.";
        }
    }*/
public:
    Node* head;
    Node* tail;
};

int main(void)
{
    MyQueue* que = new MyQueue();
    int idx = 0;
    for (int i = 0; i < people; ++i)
    {
        int cnt = 0; // 뽑힌 영웅 수
        while (cnt != selectCnt)
        {
            char hero = names[idx];
            if (isSelected[idx] == false)
            {
                que->Enqueue(hero);
                ++cnt;
            }
            idx = (idx + 1) % people;
        }
        que->Dequeue();
        que->Dequeue();
        que->Dequeue();
        que->Dequeue();
        char select = que->Dequeue();
        for (int i = 0; i < people; ++i)
    }
}

```

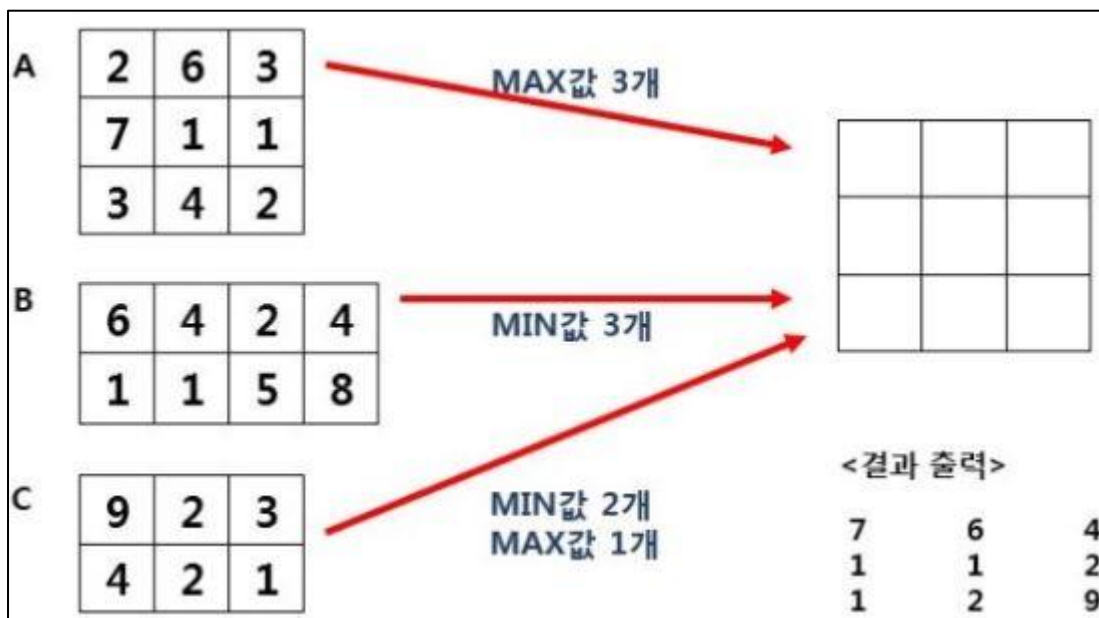
```
        {
            if (names[i] == select)
            {
                isSelected[i] = true;
                break;
            }
        }
        cout << select;
    }
    //que->delNode();
    delete que;
    _CrtDumpMemoryLeaks();
    return 0;
}
```

Level26 정예멤버 선정 (난이도 : ★★☆☆)

문제 13번 [[숙제](#) [목록보기](#)]

A, B, C 배열에 있는 정예멤버를 선정하여 후보배열에 넣으려 합니다. (A, B, C 배열은 하드코딩)

- A배열에서 MAX 3명 선출. (가장 큰수 3개)
- B배열에서 MIN 3명 선출. (가장 작은수 3개)
- C배열에서 MIN 2명, MAX 1명 선출. (가장 작은수 2개, 가장 큰 수 1개)



출력 결과

7 6 4

1 1 2

1 2 9

```

#include <iostream>
using namespace std;

const int A_y = 3;
const int A_x = 3;
const int B_y = 2;
const int B_x = 4;
const int C_y = 2;
const int C_x = 3;
const int R_y = 3;
const int R_x = 3;

int A[A_y][A_x] =
{
    2,6,3,
    7,1,1,
    3,4,2,
};

int B[B_y][B_x] =
{
    6,4,2,4,
    1,1,5,8,
};

int C[C_y][C_x] =
{
    9,2,3,
    4,2,1,
};

int R[R_y][R_x] = {};

// 오름차순 정렬
void SelectSort(int* arr, int size)
{
    for (int i = 0; i < size - 1; ++i)
    {
        for (int k = i + 1; k < size; ++k)
        {
            if (arr[i] > arr[k])
            {
                int temp = arr[i];
                arr[i] = arr[k];
                arr[k] = temp;
            }
        }
    }
}

```

```

    }
}

int main(void)
{
    const int sizeA = sizeof(A) / sizeof(int);
    const int sizeB = sizeof(B) / sizeof(int);
    const int sizeC = sizeof(C) / sizeof(int);
    const int sizeR = sizeof(R) / sizeof(int);

    int arrA[sizeA] = {};
    int arrB[sizeB] = {};
    int arrC[sizeC] = {};

    for (int i = 0; i < sizeA; ++i)
        arrA[i] = A[i / A_x][i % A_x];

    for (int k = 0; k < sizeB; ++k)
        arrB[k] = B[k / B_x][k % B_x];

    for (int o = 0; o < sizeC; ++o)
        arrC[o] = C[o / C_x][o % C_x];

    SelectSort(arrA, sizeA);
    SelectSort(arrB, sizeB);
    SelectSort(arrC, sizeC);

    int lastIdx_a = sizeA - 1;
    for (int i = 0; i < R_x; ++i)
        R[0][i] = arrA[lastIdx_a - i];

    int firstIdx_b = 0;
    for (int k = 0; k < R_x; ++k)
        R[1][k] = arrB[firstIdx_b + k];

    int firstIdx_c = 0;
    int lastIdx_c = sizeC - 1;
    for (int o = 0; o < R_x; ++o)
    {
        if (o == 2)
            R[2][o] = arrC[lastIdx_c];
        else
            R[2][o] = arrC[firstIdx_c + o];
    }
}

```



```
for (int i = 0; i < sizeR; ++i)
{
    if (i % 3 == 0)
        cout << endl;
    cout << R[i / R_x][i % R_x] << " ";
}

return 0;
}
```