

Homework #1

2018. 03. 26 (월) ~ 4. 9 (월) 23:59

* 마지막 장의 주의사항을 참고할 것.

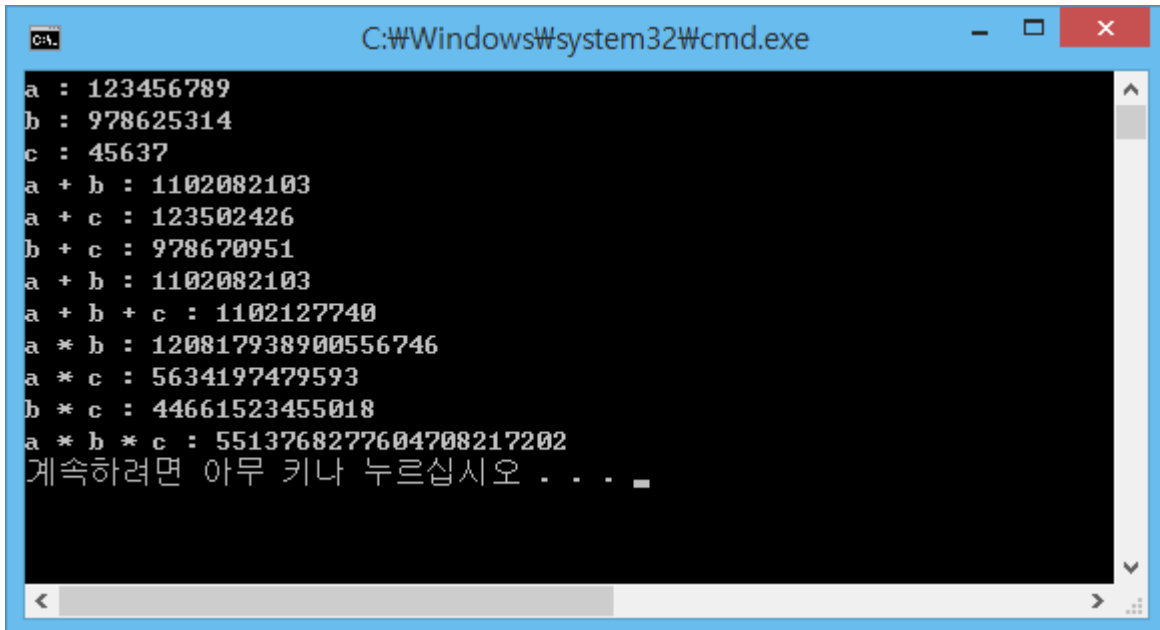
P1 (22점). 실습 시간에 작성한 큰 수 연산을 **Operator Overloading**으로 구현하시오.

제출 파일 : HW1_BigOp.h, HW1_BigOp.cpp

BigNum Class

- **Member Variable** (**Private**로 정의)
 - **int *number** : Array 형태로 저장되어 있는 큰 수를 가리키는 포인터
 - **int size** : number에 저장된 숫자의 자릿수
- **Constructor**
 - **BigNum()**
 - ◆ number를 NULL로, size를 0으로 초기화
 - **BigNum(string num)**
 - ◆ String 형태의 num을 정수형 int로 변환하여 number와 size 값 저장
 - ◆ 입력 형태는 양의 정수만을 가정 (음수, 0, 소수 등은 입력되지 않음을 가정)
 - ◆ 실습2에서와 저장 형태 동일
 - **BigNum(const BigNum& bignum)**
 - ◆ Copy constructor; bignum과 동일한 숫자를 가리키는 새로운 객체 생성
- **Destructor**
 - **~BigNum()**
 - ◆ number 삭제(메모리 할당 해제), size 0으로 초기화
- **Operator Overloading**
 - **Operator = : BigNum& operator= (const BigNum& bignum)**
 - ◆ Assignment op.; 기존 데이터 삭제 후 bignum과 동일한 새로운 객체 생성
 - **Operator + : BigNum operator+ (const BigNum& bignum)**
 - ◆ 두 BigNum variable의 덧셈 결과를 BigNum 형태로 반환
 - **Operator * : BigNum operator* (const BigNum& bignum)**
 - ◆ 두 BigNum variable의 곱셈 결과를 BigNum 형태로 반환
 - **Operator << : ostream& operator<< (ostream& os, const BigNum& bignum)**
 - ◆ number에 저장된 Array 형태의 숫자를 출력

HW1_BigOp_Test.cpp 수행 시 결과는 다음과 같이 출력됩니다.



```
C:\Windows\system32\cmd.exe
a : 123456789
b : 978625314
c : 45637
a + b : 1102082103
a + c : 123502426
b + c : 978670951
a + b : 1102082103
a + b + c : 1102127740
a * b : 120817938900556746
a * c : 5634197479593
b * c : 44661523455018
a * b * c : 5513768277604708217202
계속하려면 아무 키나 누르십시오 . . .
```

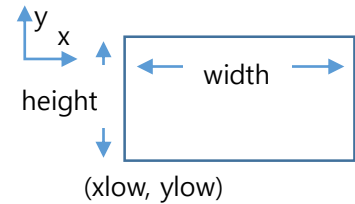
P2 (20점). 실습시간에 작성한 **Rectangle Class**(HW1_Rectangle.h, HW1_Rectangle.cpp에 구현됨)에 다음의 **Operator Overloading**을 추가하십시오.

제출 파일 : HW1_Rectangle.h, HW1_Rectangle.cpp

Rectangle Class

- **Member Variable** (**Private**로 정의)

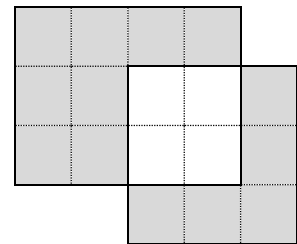
- **xlow, ylow** : 사각형의 가장 왼쪽 하단 꼭짓점의 좌표
- **width, height** : 각각 사각형의 가로/세로 길이



- **Operator Overloading**

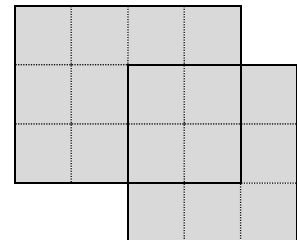
- **Operator -** : 두 사각형의 공통 부분을 뺀 넓이를 **int** 형태로 반환

- ◆ **rectangle1** : (xlow = 0, ylow = 1),
width = 4, height = 3
- ◆ **rectangle2** : (xlow = 2, ylow = 0),
width = 3, height = 3
- ◆ rectangle1 - rectangle2 = 8
- ◆ rectangle2 - rectangle1 = 5



- **Operator +** : 두 사각형에 덮인 넓이를 **int** 형태로 반환

- ◆ **rectangle1** : (xlow = 0, ylow = 1),
width = 4, height = 3
- ◆ **rectangle2** : (xlow = 2, ylow = 0),
width = 3, height = 3
- ◆ rectangle1 + rectangle2 = 17



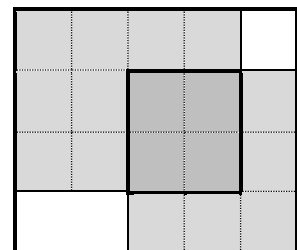
- **Operator &** : 두 사각형을 포함하는 가장 작은 사각형을 **Rectangle** 형태로 반환

- ◆ rectangle1 & rectangle2 : (xlow = 0, ylow = 0),
width = 5, height = 4

- **Operator |** : 두 사각형의 공통 부분을 **Rectangle** 형태로 반환

공통 부분이 없을 경우 xlow, ylow, width, height 모두 0

- ◆ rectangle1 | rectangle2 : (xlow = 2, ylow = 1),
width = 2, height = 2



P3 (35점). 주어진 HW1_Player.h, HW1_Team.h, HW1_League.h에 맞게 다음을 구현하여 Player, Team, League Class를 완성하시오. (HW1_AllPlayers.cpp를 수행하여 정상적으로 동작하는지 테스트할 수 있습니다.)

제출 파일 : HW1_Player.cpp, HW1_Team.cpp, HW1_League.cpp

Player Class (template <class T>)

- **Member Variable** (**Private**로 정의)
 - **string m_name** : 선수의 이름
 - **T m_height, m_weight** : 선수의 키, 몸무게
- **Constructor**
 - **Player()**
 - ◆ m_name을 빈 **string**으로, m_height와 m_weight를 **0**으로 초기화
 - **Player(string name, T height, T weight)**
 - ◆ m_name을 **name**으로, m_height를 **height**로, m_weight를 **weight**로 초기화
- **Operator Overloading**
 - **Operator << : ostream& operator<< (ostream& os, const Player<T>& pl)**
 - ◆ 학생의 이름과 점수를 다음과 같은 꼴로 출력
예) m_name = "KHBAEK", m_height = 200, m_weight=100인 경우
name : KHBAEK / height : 200cm, weight : 100kg
 - **Operator = : Player<T>& operator= (const Player<T>& pl)**
 - ◆ pl의 m_name, m_height, m_weight 복사

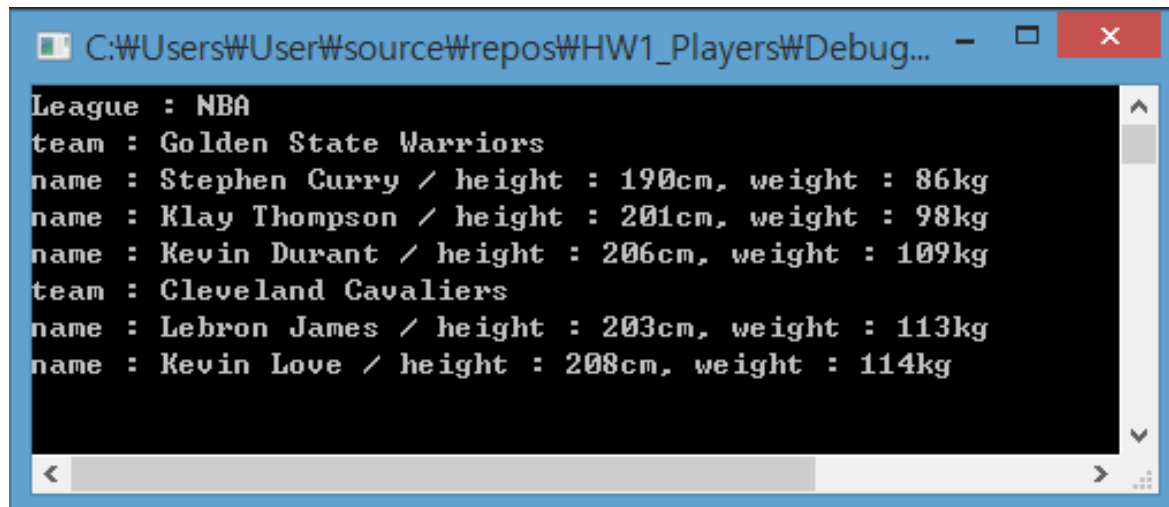
Team Class (template <class T>)

- **Member Variable** (**Private**로 정의)
 - **string** m_name : 팀의 이름
 - **Player<T>*** m_list : 팀 내 선수 목록
 - **int** m_size : 팀 내 선수의 수
- **Constructor**
 - **Team()**
 - ◆ m_name을 빈 **string**으로, m_size를 **0**으로, m_list를 **NULL**로 초기화
 - **Team(string name, int size)**
 - ◆ m_name을 **name**으로, m_size를 **size**로 초기화
 - ◆ m_list에 **size** 크기만큼의 **Player** Array 생성
- **Destructor**
 - **~Team()**
 - ◆ m_list 삭제
- **Operator Overloading**
 - **Operator [] : Player<T>& operator[] (int i)**
 - ◆ m_list의 **i번째** Player 반환 (i는 0부터 m_size-1까지 가정)
 - **Operator << : ostream& operator<< (ostream& os, const Team<T>& te)**
 - ◆ 팀 이름 출력 및 팀 내 선수에 대해 다음과 같은 꼴로 출력
예) m_name = "TEAM1" 인 경우
team : TEAM1
name : KHBAEK / height : 200cm, weight : 100kg ...
 - **Operator = : Team<T>& operator= (const Team<T>& te)**
 - ◆ 기존 데이터 삭제 후 te의 m_name, m_size, m_list 복사

League Class (template <class T>)

- **Member Variable** (**Private**로 정의)
 - **string m_name** : 리그의 이름
 - **Class<T>* m_list** : 리그에 속하는 팀 목록
 - **int m_size** : 리그에 속하는 팀의 수
- **Constructor**
 - **League()**
 - ◆ m_name을 빈 **string**으로, m_size를 **0**으로, m_list를 **NULL**로 초기화
 - **League(string name, int size)**
 - ◆ m_name을 **name**으로, m_size를 **size**로 초기화
 - ◆ m_list에 **size** 크기만큼의 **Team** Array 생성
- **Destructor**
 - **~League()**
 - ◆ m_list 삭제
- **Operator Overloading**
 - **Operator [] : Team<T>& operator[] (int i)**
 - ◆ m_list의 **i번째** Team 반환 (i는 0부터 m_size-1까지 가정)
 - **Operator << : ostream& operator<< (ostream& os, const League<T>& leag)**
 - ◆ 리그 이름 출력 및 리그 내 각 팀에 대해 다음과 같은 꼴로 출력
예) m_name = "LEAGUE2" 인 경우
league : LEAGUE2
team : TEAM1
name : KHBAEK / height : 200cm, weight : 100kg ...
 - **Operator = : League<T>& operator= (const League<T>& leag)**
 - ◆ 기존 데이터 삭제 후 leag의 m_name, m_size, m_list 복사

HW1_AllPlayers.cpp 수행 시 결과는 다음과 같이 출력됩니다.



```
C:\Users\User\source\repos\HW1_Players\Debug...  
League : NBA  
team : Golden State Warriors  
name : Stephen Curry / height : 190cm, weight : 86kg  
name : Klay Thompson / height : 201cm, weight : 98kg  
name : Kevin Durant / height : 206cm, weight : 109kg  
team : Cleveland Cavaliers  
name : LeBron James / height : 203cm, weight : 113kg  
name : Kevin Love / height : 208cm, weight : 114kg
```

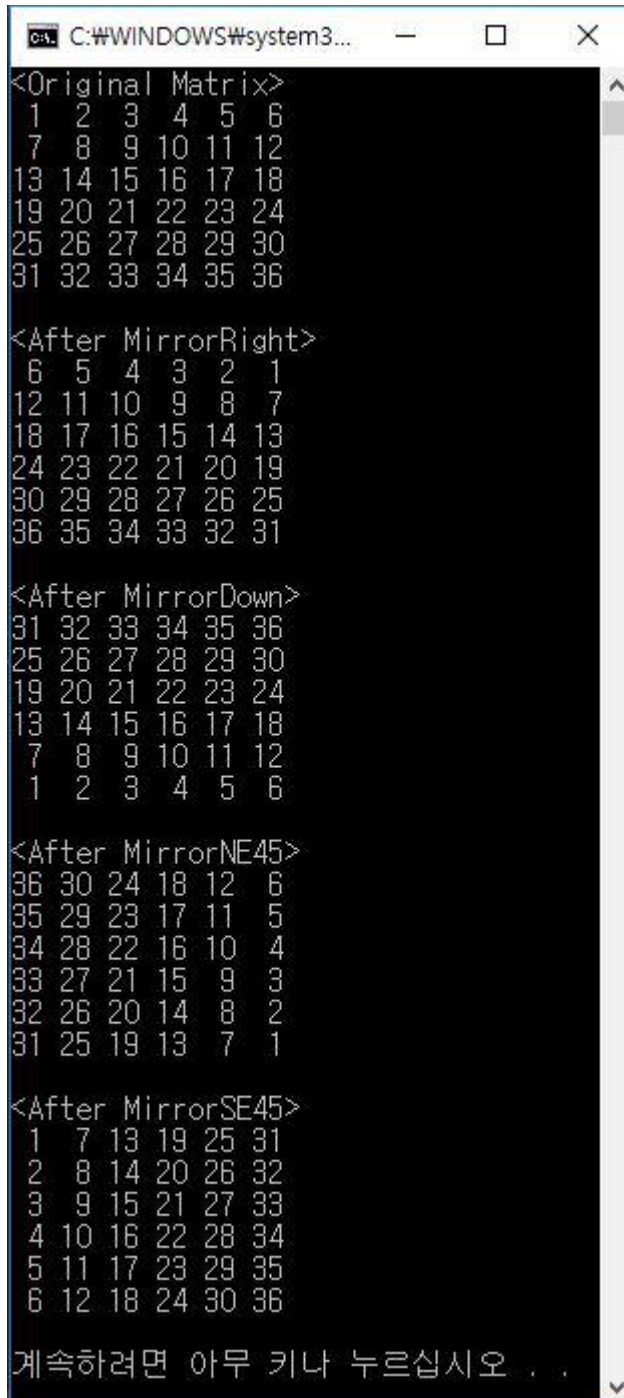
P4 (23점). 주어진 Swap() Function을 이용, 다음 조건을 만족하도록 Array2D Class를 작성하시오.

제출 파일 : HW1_Array2D.h, HW1_Array2D.cpp

Array2D Class

- Member Variable
 - `int** m_array` : 2차원 Array Pointer
 - `int m_size` : Array의 행과 열의 개수 (행과 열의 개수는 동일)
- Constructor
 - `Array2D(int size)`
 - ◆ `m_size`를 `size`로 초기화
 - ◆ `m_array`에 `size X size` 크기의 2차원 `int` Array를 생성하고 각 Element 값을 순서대로 1부터 (`size X size`)로 설정
예) `size = 2`인 경우, `m_array[0][0]`, `[0][1]`, `[1][0]`, `[1][1]`은 각각 1, 2, 3, 4
- Destructor
 - `~Array2D()`
 - ◆ `m_array` 삭제
- Operator Overloading
 - Operator `<<` : `ostream& operator<< (ostream& os, const Array2D& arr)`
 - ◆ `m_array`를 2차원 형태로 출력 예) `size = 2`일 때, 1 2
3 4
- Member Function
 - `void Swap (int* a, int* b)`
 - ◆ `int temp = *a; *a = *b; *b = temp;`
 - `void mirrorRight() / void mirrorDown()`
 - ◆ Swap function을 이용하여 현재 Array를 각각 오른쪽 / 아래로 대칭
 - `void mirrorNE45() / void mirrorSE45()`
 - ◆ Swap function을 이용하여 현재 Array를 각각 NE방향 45도 선 / SE방향 45도 선에 대하여 대칭

HW1_ArrayTest.cpp 수행 시 결과는 다음과 같이 출력됩니다.



```
C:\WINDOWS\system32\cmd.exe
<Original Matrix>
1 2 3 4 5 6
7 8 9 10 11 12
13 14 15 16 17 18
19 20 21 22 23 24
25 26 27 28 29 30
31 32 33 34 35 36

<After MirrorRight>
6 5 4 3 2 1
12 11 10 9 8 7
18 17 16 15 14 13
24 23 22 21 20 19
30 29 28 27 26 25
36 35 34 33 32 31

<After MirrorDown>
31 32 33 34 35 36
25 26 27 28 29 30
19 20 21 22 23 24
13 14 15 16 17 18
7 8 9 10 11 12
1 2 3 4 5 6

<After MirrorNE45>
36 30 24 18 12 6
35 29 23 17 11 5
34 28 22 16 10 4
33 27 21 15 9 3
32 26 20 14 8 2
31 25 19 13 7 1

<After MirrorSE45>
1 7 13 19 25 31
2 8 14 20 26 32
3 9 15 21 27 33
4 10 16 22 28 34
5 11 17 23 29 35
6 12 18 24 30 36

계속하려면 아무 키나 누르십시오 . . .
```

- 참고

- 주어진 파일명을 정확히 지켜 제출할 것! 이를 어길 시 감점
- 주어진 예시와 같은 양식으로 출력되게 할 것! 이를 어길 시 감점
- 채점은 문제에 주어진 예시보다 복잡한 Test Case로 진행
- 표절 금지. 표절 적발 시 해당 과제 0점 처리 및 교수님께 통보
- 질문이 있는 경우, 검색을 충분히 해 본 후에도 해결되지 않는 것에 한해 질문 요망

- 풀이 및 제출 방법

- 각 문제에 대한 코드를 각각의 .h / .cpp 파일에 작성
- 파일 이름은 각 문제에 주어진 대로 정의
- 작성한 코드를 하나의 압축 파일로 압축
 - ◆ 압축 파일 이름은 "HW1_(이름)_(학번).zip"으로, zip 방식 압축 사용
 - ◆ 예) "HW1_김태환_2017-11111.zip"
- 만든 압축 파일을 eTL 강의 페이지에 마련된 "과제1" 제출함으로 제출

- 제출 기한

- 4월 9일 (월) 오후 11시 59분까지
- 오류 발생으로 eTL 과제함에 제출이 되지 않는 경우 기한 내에 이메일로 제출
E-mail : ds@snucad.snu.ac.kr
- 지연 제출은 받지 않음(eTL / 이메일 모두), 지연 제출 또는 미제출 시 0점
- 파일이 첨부되어 제출되었는지 반드시 확인! 첨부되지 않은 채로 제출 시 0점