

과제 #5

HW1, HW2, HW3, (HW4)

과제 제출 마감: 11/10 24:00

조교 노인우, inwoo13@hanyang.ac.kr

조교 한중수, soohan@hanyang.ac.kr

Homework

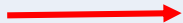
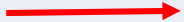
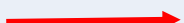
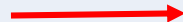
- ◆ 과제 제출 마감: 11/10 24:00
- ◆ Gitlab repository에 “HW5” 폴더를 만든 후 진행
- ◆ 실습 서버 제출, 개인 PC 제출 중 편한 방법으로 제출
- ◆ 각 과제마다 HW5 폴더 내에 과제별 폴더를 만든 후 제출
- ◆ 채점 기준은 실습서버 환경에서 채점

Homework_01 – “정수 집합 연산 (Integer set operations)”

- ◆ HW03과 같이 정수 집합의 합집합, 교집합, 차집합을 계산하는 프로그램을 STL set의 함수로 구현
- ◆ 작성 규칙
 - `set<int> SetIntersection(const set<int>& set0, const set<int>& set1);`
 - `set<int> SetUnion(const set<int>& set0, const set<int>& set1);`
 - `set<int> SetDifference(const set<int>& set0, const set<int>& set1);`
 - 본 함수들은 클래스 내의 static 함수나 전역변수로 구현하여 사용
 - `std::set<int>`를 이용하여 구현. 메인 함수는 변경하지 않는 것을 권장.
 - 함수 연산에 대한 설명은 HW03 참조
- ◆ 파일명: `simple_int_set`
(`simple_int_set.h` `simple_int_set.cc` `simple_int_set_main.cc`)
- ◆ 입력: { num1 num2 ... numk1 } OP { num1 num2 ... numk2 }
OP = +, *, -
- ◆ 출력: 연산 결과 집합을 같은 형식으로 출력

Homework_01 – “정수 집합 연산 (Integer set operations)”

◆ 입력 및 출력 예시 (전과 동일)

```
$ ./simple_int_set  
{ 1 2 3 } + { 3 4 5 }  두 집합의 합집합  
{ 1 2 3 4 5 }  
{ -1 5 3 2 } - { 1 2 3 }  두 집합의 차집합  
{ -1 5 }  
{ -1 5 3 2 } * { 1 2 3 }  두 집합의 교집합  
{ 2 3 }  
0  주어진 형식 외의 입력: 종료
```

Homework_02 – “정렬된 배열 (sorted number array)”

- ◆ 숫자를 입력 받고, 입력 받은 수들의 오름차순, 내림차순, 최대값, 최소값을 구하는 프로그램 작성
- ◆ 설계 시 고려사항
 - 숫자를 입력 받으면서 오름차순 정렬을 하도록 한다.
 - 숫자를 입력하면 배열에 추가한다. (문자열을 숫자로 변환하는 함수 사용)
 - 다음과 같은 명령을 처리한다.
 - ascend: 입력 받은 수를 오름차순으로 출력한다.
 - descend: 입력 받은 수를 내림차순으로 출력한다.
 - max: 입력 받은 수 중 최대값을 출력한다.
 - min: 입력 받은 수 중 최소값을 출력한다.
 - quit: 프로그램을 종료한다.
- ◆ 파일명: sorted_array
(sorted_array.h sorted_array.cc sorted_array_main.cc)
- ◆ 입력: { num1 num2 ... numk1 } OP { num1 num2 ... numk2 }
OP = +, *, -
- ◆ 출력: 연산 결과 집합을 같은 형식으로 출력

Homework_02 – “정렬된 배열 (sorted number array)”

◆ 가이드라인: SortedArray 클래스 예시

```
◆ class SortedArray {  
    public:  
        SortedArray();  
        ~SortedArray();  
        void AddNumber (int num);  
  
        vector<int> GetSortedAscending() const;  
        vector<int> GetSortedDescending() const;  
        int GetMax() const;  
        int GetMin() const;  
  
    private:  
        vector<int> numbers_  
};
```

Homework_02 – “정렬된 배열 (sorted number array)”

◆ 입력 및 출력 예시

```
$ ./sorted_array
9 3 6 2 7      → 수열 입력
ascend         → 명령어
2 3 6 7 9      → 명령 결과
descend
9 7 6 3 2
max
9
min
2
10 3          → 수열에 수 추가
ascend
2 3 3 6 7 9 10
quit          → 프로그램 종료
```

Homework_03 – “도형 그리기 (shape drawing)”

- ◆ 캔버스의 크기를 입력 받고, 캔버스 내에 도형을 그리는 프로그램 작성
- ◆ 다음 클래스를 작성. 필요한 경우 private 멤버 함수를 추가하여도 무방함
- ◆
enum { RECTANGLE, TRIANGLE_UP, TRIANGLE_DOWN };
enum { ERROR_OUT_OF_CANVAS = -1, ERROR_INVALID_INPUT = -2 };

```
struct Shape {  
    int type;  
    int x, y;  
    int width, height;  
    char brush; // The character to draw the shape  
};
```

```
class Canvas {  
public:  
    Canvas(size_t row, size_t col);  
    ~Canvas();  
  
    int AddShape(const Shape &s);           // Return the index of the shape.  
    void DeleteShape(int index);  
    void Draw(ostream& os);  
    void Dump(ostream& os);  
  
private:  
    size_t row_, col_;  
    vector<Shape> shapes_;  
};
```


Homework_03 – “도형 그리기 (shape drawing)”

◆ 다음에 유의하여 작성

- 처음 실행 시 캔버스의 크기를 입력 받음 (가로길이, 세로길이)
- 모든 도형 정보를 vector에 저장하고 매 출력 시 모두 다시 그리는 형식으로 구성
- 캔버스 내의 모든 도형은 '겹쳐 그리는 것' 이 가능해야 함
 - 뒤에 입력된 도형이 앞에 입력된 도형을 덮어서 그림
- main() 함수에서 명령, 그리기 원하는 도형의 중심 좌표, 넓이, 높이, 모양(brush) 등을 입력 받음
 - 명령: add, delete, draw, dump, quit // add 함수의 경우 (x좌표, y좌표) 순서로 입력.
 - 도형 종류: rect, tri_up, tri_down // 사각형의 경우 평행 및 수직인 선만 입력으로 주어짐.
- 공백은 '.'으로, 도형 범위 내의 공간은 brush 문자로 출력.
- AddShape() 함수는 도형이 추가된 배열에서의 위치(index)를 반환.
 - 도형의 공간이 캔버스의 범위를 벗어나면 에러 반환 (ERROR_OUT_OF_CANVAS)
 - 잘못된 입력값 (짝수 크기)이 입력되면 ERROR_INVALID_INPUT 반환
- DeleteShape() 함수는 해당 index의 도형을 지움. 잘못된 index 값은 무시.
- Draw() 함수에서 입력 받은 좌표와 크기 정보를 이용하여 캔버스에 도형을 그림.
 - tri_up, tri_down 의 경우 좌표를 꼭지점으로 이용하여 계단식으로 올라가거나 내려가며 그림
- Dump() 함수는 현재 배열에 있는 도형 정보를 index와 함께 표시.

◆ 파일명: draw_shape

(draw_shape.h draw_shape.cc draw_main.cc)

Homework_03 – “도형 그리기 (shape drawing)”

◆ 입력 및 출력 예시

```
$/draw_shape
```

```
10 9
```

// 캔버스 가로길이, 세로길이 입력

```
0123456789
```

```
0.....
1.....
2.....
3.....
4.....
5.....
6.....
7.....
8.....
```

```
add rect 5 5 3 3 *
```

```
draw
```

```
0123456789
```

```
0.....
1.....
2.....
3.....
4...***...
5...***...
6...***...
7.....
8.....
```

// 좌표 (5, 5)를 중심으로 너비, 폭이 3인 사각형 그림

```
add tri_up 7 7 2 #
```

```
draw
```

```
0123456789
```

```
0.....
1.@@@@@.
2.@@@.
3...@.
4...* * *
5...* * *
6...* * *
7.....#
8.....###
```

// 좌표 (7, 7)을 꼭지점으로 하는 높이 2의 삼각형

```
add tri_up 8 8 3 #
```

```
error out of canvas
```

// 입력되는 도형의 공간이 캔버스의 범위를 넘어 날 때 에러 출력

```
add rect 5 5 6 6 +
```

```
error invalid input
```

// 사각형의 높이, 너비가 짝수가 입력 될 때 에러 출력

```
dump
```

```
0 rect 5 5 3 3 *
```

```
1 tri_down 3 3 3 @
```

```
2 tri_up 7 7 2 #
```

```
$/draw_shape
```

```
13 4
```

// 캔버스 가로길이, 세로길이 입력

```
01234567890123
```

```
0.....
1.....
2.....
3.....
```

// 처음 캔버스를 생성하면 빈 캔버스를 1회 출력

```
quit
```

Homework_03 – “도형 그리기 (shape drawing)”

◆ 입력 및 출력 예시

```
delete 3           // 없는 인덱스를 삭제하는 명령은 무시됨
delete 1
draw
0123456789
0.....
1.....
2.....
3.....
4...***...
5...***...
6...***...
7.....#..
8.....###.
dump
0 rect 5 5 3 3 *
1 tri_up 7 7 2 #
delete 2           // 없는 인덱스를 삭제하는 명령은 무시됨
```

```
draw
0123456789
0.....
1.....
2.....
3.....
4...***...
5...***...
6...***...
7.....#..
8.....###.
9.....#####
delete 1
draw
0123456789
0.....
1.....
2.....
3.....
4...***...
5...***...
6...***...
7.....
8.....
9.....
dump
0 rect 5 5 3 3 *
quit
```

practice_04 – “문자열 받기 getline()”

◆ std::getline() 함수

- Get line from stream into string

◆ Syntax

- istream& getline (istream& is, string& str, char delim);
- istream& getline (istream& is, string& str);

◆ Parameters & Return Value

- Parameter
 - **is** istream object from which characters are extracted
 - **str** string object where the extracted line is stored
- Return Value
 - The same as parameter 'is'

◆ Example

```
1 // extract to string
2 #include <iostream>
3 #include <string>
4
5 int main ()
6 {
7     std::string name;
8
9     std::cout << "Please, enter your full name: ";
10    std::getline (std::cin, name);
11    std::cout << "Hello, " << name << "!\\n";
12
13    return 0;
14 }
```

