

프로그래밍 과제 05

1. `vector<int>` 클래스 처럼 제한이 없는(unlimited) 개수의 정수를 저장하면서, 인덱싱(indexing), 삽입, 삭제 등의 연산을 제공하는 클래스 `OrderedVecInt`를 작성하라. 단, `vector<int>`와는 달리 정수들은 항상 오름차순으로 정렬된 순서로 저장되어야 한다. 이 클래스 내부에서 `vector`를 사용해서는 안된다. 이 클래스는 다음과 같이 사용될 수 있어야 한다.

```
#include <iostream>
#include <vector>
#include <cstdlib>
#include <ctime>
using namespace std;

class OrderedVecInt {
private:
    int capacity = 0, size = 0;    // capacity는 동적 배열의 길이, size는 저장된 정수 개수
    int *arr = nullptr;           // 동적 배열의 시작주소. 처음에는 길이가 0인 배열
                                   // 첫 번째 원소가 삽입될 때 길이를 1로 만들고,
                                   // 그 다음 부터는 필요시 길이를 2배로 늘린다.

    // 완성하라.

};

int main() {
    srand((unsigned int) time(NULL)); // pseudo-random number generator
    int n, k;
    OrderedVecInt vec;
    cin >> n;
    for (int i=0; i<n; i++) {
        int k = rand() % 1000;
        vec.insert(k); // 정수 k를 삽입한다. 정수들은 정렬되어 저장되어야 한다.
    }

    for (int i=0; i<vec.size(); i++)
        cout << vec[i] << " "; // indexing연산자 []를 제공한다.
    cout << endl;

    if (vec.remove_by_val(vec[2])) // 매개변수로 주어진 정수를 찾아 삭제한다.
        cout << "Remove done" << endl; // 그런 값이 존재하면 true,
    else // 아니면 false를 반환한다.
        cout << "Remove failed" << endl;

    if (vec.remove_by_index(4)) // 매개변수로 주어진 인덱스 위치의 정수를 삭제한다.
        cout << "RemoveIndex done" << endl; // 유효한 인덱스이면 true,
    else // 아니면 false를 반환한다.
        cout << "RemoveIndex failed" << endl;

    for (int i=0; i<vec.size(); i++) // size()는 저장된 정수의 개수를 반환한다.
        vec[i]-=10; // 저장된 정수를 수정할 수 있다. 이 경우 정렬이 흐트러질 수 있다.

    for (int i=0; i<vec.size(); i++)
        cout << vec[i] << " ";
    cout << endl;
```

```

    return 0;
}

```

입력 예	출력
10	101 170 182 263 550 672 773 801 965 991 91 160 253 540 763 791 955 981

2. n 개의 도형이 입력으로 주어진다. 도형의 종류에는 “좌표축에 평행한 직사각형”과 “원”이 있다. 이 도형들이 입력된 후 다시 추가로 하나의 원이 주어진다. 입력으로 주어진 도형들 중 추가로 주어진 원과 교차하는 도형들을 모두 찾아서 면적 순으로 정렬하여 출력하는 프로그램을 작성하라. 도형이 원의 내부에 포함되거나 혹은 반대로 원이 도형의 내부에 포함되는 경우도 교차하는 것으로 간주한다. 입력은 `input1.txt` 파일로 주어진다. 파일의 첫 줄에는 도형의 개수 $n \leq 1000$ 이 주어지고, 이어진 n 줄에는 한 줄에 하나의 도형이 다음과 같은 형식으로 주어진다: 각 줄의 처음에는 먼저 도형의 종류를 나타내는 하나의 문자(사각형은 “R”, 원은 “C”)가 주어진다. 그런 다음 사각형의 경우 4꼭지점의 x 좌표와 y 좌표의 최소값과 최대값을 나타내는 4개의 정수가 $x_{min}, x_{max}, y_{min}, y_{max}$ 의 순서로 주어진다. 원의 경우 중심의 x 좌표와 y 좌표, 반지름을 나타내는 3개의 정수가 주어진다. 파일의 마지막 줄에는 추가로 주어지는 원의 중심의 x 좌표, y 좌표, 반지름을 나타내는 3개의 정수가 주어진다. 출력은 화면으로 한다. 면적 순으로 정렬된 도형들은 한 줄에 하나씩 입력과 동일한 형식으로 출력한다. 배열을 사용해서는 안되며, 원과 사각형을 표현하는 클래스 `Circle`과 `Rect`를 정의하여 사용하라. 두 클래스의 모든 데이터 멤버는 `private`으로 하라. 단, 사각형과 원을 표현하는 객체들은 오로지 동적 생성된 객체만을 사용하라. 즉, `Circle` 혹은 `Rect` 타입의 이름을 가진 객체 혹은 `Circle *` 혹은 `Rect *` 타입이 아닌 `Circle` 혹은 `Rect` 타입의 벡터나 배열을 사용해서는 안된다.

입력 예(INPUT1.TXT)	출력
11 R 2 4 2 5 R 3 6 4 6 C 4 7 2 R 9 11 2 4 R 4 7 8 12 C 8 7 1 R 6 9 9 11 C 3 11 1 C 8 14 3 R 5 11 12 14 R 11 13 5 13 8 9 4	C 8 7 1 R 3 6 4 6 R 6 9 9 11 R 4 7 8 12 R 5 11 12 14 C 4 7 2 R 11 13 5 13 C 8 14 3
8 R 2 4 2 5 C 4 10 1 C 8 7 1 R 7 9 10 13 C 13 4 1 R 12 14 11 14 C 9 10 6 R 3 15 14 16 9 10 5	C 4 10 1 C 8 7 1 R 7 9 10 13 R 12 14 11 14 R 3 15 14 16 C 9 10 6

