실습과제 10

(각 문제에 대해서 주어진 테스트 데이터를 모두 통과하지 못한 경우에는 반드시 그 사실을 명시해야 한다.)

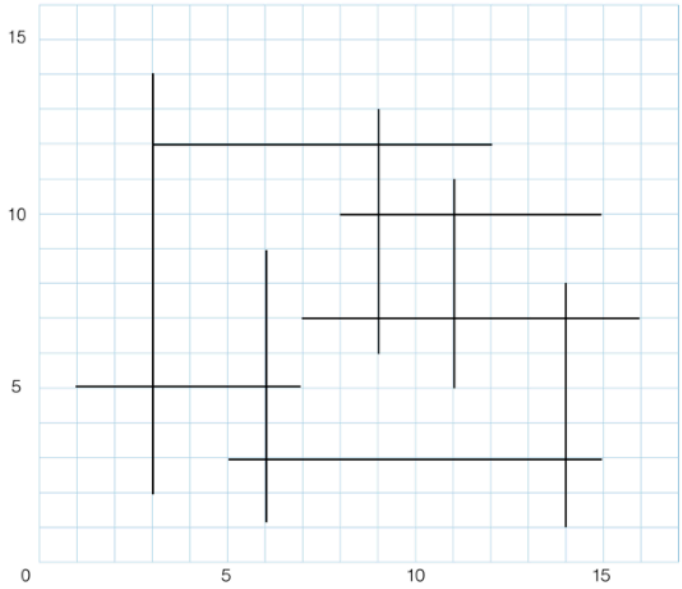
1. 정수 수열에서 어떤 정수가 자신보다 앞에 나온 모든 정수들 보다 크거나 같으면 리더(leader)라고 부른다. 첫 번째 정수는 무조건 리더이다. 입력 파일로부터 정수들을 읽어서 리더들만 배열에 저장한 후 출력하는 프로그램을 작성하라. 이 문제를 해결하기 위해서 새로운 정수가 입력될 때 마다 이 정수가 리더인지 아닌지 판단한 하는 함수를 작성하다. 이 함수를 호출하여 리더이면 배열에 저장하고 아니면 버린다. 먼저 리더들의 개수를 출력하고 이어서 리더들을 순서대로 화면으로 출력하라. 전역변수를 사용해서는 안된다. 입력은 input1.txt 파일로부터 받고, 입력 파일에서 정수의 개수는 주어지지 않고 파일의 끝까지 읽어야 한다.

입력 예 (INPUT1.TXT)	출력
1 2 3 4 5 6 7 8 9 10	10: 1 2 3 4 5 6 7 8 9 10
1 1 1 1 1	5: 1 1 1 1 1
10 1 4 0 4 4 -11 4 1 1 6 -8	1: 10
6 -5 -1 -4 20 -1 -12 -9 20	3: 6 20 20
12 0 28 1 9 37 46 -92 -8 37 0 0 12 124	5: 12 28 37 46 124

2. 키보드로 부터 연속해서 정수들을 입력받는다. 정수가 하나 씩 입력될 때 마다 현재까지 입력된 정수들을 오름차순으로 정렬하여 화면에 출력한다. 단, 새로 입력된 정수가 이미 배열에 저장되어 있다면 추가하는 대신 “duplicate entry”라고 출력한다. 사용자가 -1을 입력하면 프로그램을 종료한다. 이 일을 하기 위해서 다음과 같은 두 함수를 작성하라. 먼저 현재까지 입력된 정수들이 저장되어 있는 배열에 새로 입력된 정수가 이미 존재하는지 검사하여 만약 있으면 그 위치(배열 인덱스)를 반환하고, 없으면 -1을 반환하는 함수 find를 작성하라. 또한 새로 입력된 정수가 duplicate entry가 아닌 경우에 이 정수를 배열에 오름차순으로 정렬되도록 삽입하는 함수 insert를 작성하라. 그리고 main 함수에서는 이 두 함수를 적절히 이용하여 문제를 해결하라. Duplicate entry라고 출력할 때 그 배열에서 그 정수의 위치를 함께 출력하라. 어떤 전역 변수도 사용해서는 안된다.

입력 예	출력
5	5
2	2 5
5	duplicate entry: 1
1	1 2 5
3	1 2 3 5
3	duplicate entry: 2
-1	

3. 입력으로 N개의 수직 혹은 수평 선분이 주어진다. 선분들간의 교차점의 좌표를 모두 계산하여 x좌표에 대한 오름차순으로 정렬하여 출력하는 프로그램을 작성하라. x좌표가 동일한 경우에는 y좌표가 작은 점을 먼저 출력한다. 입력은 input3.txt 파일로부터 받는다. 파일의 첫 줄에는 선분의 개수 N이 주어지고, 이어진 N줄에는 각 줄마다 하나의 선분의 시작점과 끝점의 좌표가 주어진다. 수평 선분의 경우 x좌표가 작은 점이 먼저 주어지고, 수직 선분의 경우 y좌표가 작은 점이 항상 먼저 주어진다. 수직이나 수평이 아닌 선분이 주어지는 경우는 없다. 수평 선분끼리 만나거나 혹은 수직 선분끼리 만나는 경우는 교차점으로 간주하지 않는다. 이 문제를 해결하기 위해서 두 선분이 교차하는지 검사하는 함수 intersect를 만들어 사용하라. 매개 변수로 두 선분을 받아서 교차하면 1, 그렇지 않으면 0을 반환하라. 어떤 전역변수도 사용해서는 안된다.



입력 예 (INPUT3.TXT)	출력
10	[3, 5]
5 3 15 3	[3, 12]
1 5 7 5	[6, 3]
7 7 16 7	[6, 5]
8 10 15 10	[9, 7]
3 12 12 12	[9, 10]
3 2 3 14	[9, 12]
6 1 6 9	[11, 7]
9 6 9 13	[11, 10]
11 5 11 11	[14, 3]
14 1 14 8	[14, 7]

4. 두 개의 입력 파일 input4_1.txt와 input4_2.txt에 각각 정수들이 이미 오름차순으로 정렬된 상태로 저장되어 있다. 먼저 input4_1.txt 파일에 있는 정수들을 읽어 배열 data1에 순서대로 저장하고, input4_2.txt에 있는 정수들을 읽어 배열 data2에 순서대로 저장한다. 두 배열에 저장된 정수들을 배열 data3로 합쳐서 하나의 정렬된 배열을 만들어 출력하라. 두 파일에 저장된 정수들은 각각 최대 1000개이다. 이 문제를 해결하기 위해서 data1과 data2에 있는 정수들을 data3에 정렬된 상태로 저장해주는 함수 merge를 작성하여 이용하라.

```
int main() {
    int data1[1000], data2[1000], data3[2000];
    int n1 = 0, n2 = 0, n3 = 0;
    FILE *fp1 = fopen("input1.txt", "r");
    FILE *fp2 = fopen("input2.txt", "r");
    while (!feof(fp1))  fscanf(fp1, "%d", &data1[n1++]);
    while (!feof(fp2))  fscanf(fp2, "%d", &data2[n2++]);
    fclose(fp1); fclose(fp2);

   /* 여기에서 배열 data3에 정렬하여 저장하기 위해서 함수 merge를 호출한다. */

    for (int i=0; i<n3; i++)    /* n3 = n1 + n2 */
        printf("%d\n", data3[i]);
}
```

INPUT4_1.TXT	INPUT4_2.TXT	출력
2 4 5 8 10 11 13 40	3 9 11 28	2 3 4 5 8 9 10 11 11 13 28 40

5. [Self avoiding walk] 2차원 평면에서 원점 (0,0)에서 출발한다. 사용자가 현재의 위치에서 상하좌우 어떤 한 방향으로 얼마 만큼 이동하라는 명령을 내리면 그렇게 이동한다. 명령은 두 음이 아닌 정수로 표현된다. 우선 방향은 0, 1, 2, 3으로 표시하고 0은 y좌표가 증가하는 방향, 1은 x좌표가 증가하는 방향, 2는 y좌표가 감소하는 방향, 그리고 3은 x좌표가 감소하는 방향이다. 예를 들어 2 7은 y좌표가 7만큼 감소하는 위치로 이동하라는 명령이다. 프로그램은 사용자가 현재까지 이동한 궤적을 기억하고 있어야 한다. 사용자가 내린 명령대로 이동했을 때 만약 지금까지 이동한 궤적과 교차하면 invalid move라고 출력하고 이동 명령을 거부한다. 만약 그렇지 않으면 명령대로 이동하고 이동한 점의 좌표를 출력한다. 사용자가 −1 −1을 입력할 때 까지 이 일을 계속한다. 사용자가 −1 −1을 입력하면 프로그램을 종료한다. 이 문제를 해결하기 위해 사용자가 내린 명령이 invalid move인지 아닌지 검사하는 함수 check를 작성하라. 전역변수를 사용해서는 안된다.

입력 예	출력
1 3	3 0
3 5	invalid move
2 7	3 -7
3 9	-6 -7
0 3	-6 -4
1 11	invalid move
1 8	2 -4
2 3	invalid move
0 5	invalid move
0 2	2 -2
-1 -1	