1. 연락처 DataBase

제출횟수 :**020** 제출시간 :**-** 상태 : **-** [제출코드](https://cert.ssafy.com/exam/contest/management/problem?contestId=AWzWvU3UABEBAAXt&problemId=AVljnYgKAObxCYXu&problemNo=1)

[•••](https://cert.ssafy.com/exam/contest/management/problem?contestId=AWzWvU3UABEBAAXt&problemId=AVljnYgKAObxCYXu&problemNo=1)

제출횟수 :**020**

| **제출시각** | **제출** | **결과** |
| --- | --- | --- |

* 문제
* 문제풀이

|  |  |
| --- | --- |
| 제한 조건 | 실행시간 : 최대 50개의 테스트 케이스를 합쳐서 5초 이내    메 모 리 : Heap, Global, Stack 등을 모두 합해 최대 256MB까지 사용 가능 (단, 스택은 최대 1MB까지 사용 가능) |
| 제출 횟수 제한 | 20회 |
| 채점 | 답안을 제출하면 제공된 sample\_input.txt 에 대한 결과를 판정해서 실시간으로 알려주며 그 의미는 다음과 같다.    **Accept : sample input에 대하여 정답 또는 부분 점수 획득**    **Fail : sample input에 대하여 오답 또는 런타임 에러, time out 등**    (※ sample\_input.txt 파일 다운로드는 문제 지문 바로 아래에 있습니다)    최종결과는 별도의 채점용 input으로 진행되므로 시험 중 판정 결과와 다를 수 있다. |
| 평가 | 최종 점수는 별도로 제작된 평가용 테스트 케이스로 측정하며, 동점자간에는 실행시간에 따라 점수 차이가 발생할 수 있습니다.    따라서 시험이 끝날 때까지 본인의 코드를 최적화 하시기를 권장합니다. |

**[코드 작성 시 주의점]**

**① C 또는 C++로 답안을 작성하시는 응시자께서는 검정시스템에 제출 시, Language에서 C++를**

**선택하신 후 제출해 주시기를 바랍니다.**

**② Main 부분과 User Code 부분으로 구성되어 있습니다.**

A. Main 부분 : 수정할 수 없는 코드이며, 채점 시 주어지는 코드 그대로 사용됩니다.

B. User Code 부분 : 실제 응시자가 작성해야 하는 코드이며, 제출 시에는 코드 내에

라이브러리 함수뿐 아니라 표준 입출력 함수도 포함되어서는 안됩니다.

C. 서버에서 Main 부분과 User Code 부분은 별도의 파일로 분리되어 컴파일이 됩니다.

(main.cpp /solution.cpp 또는 Solution.java/UserSolution.java)

**③ Local PC에서 프로그래밍할 시 유의 사항**

A. 2개의 파일을 생성하셔야 합니다.

(main.cpp/solution.cpp 또는 Solution.java/UserSolution.java)

B. Main 부분의 코드를 main.cpp 또는 Solution.java에 복사해서 사용하시기 바랍니다.

C. sample\_input.txt를 사용하시기 위해서는 Main 부분의 코드 내에 표준 입력을 파일로

전환하는 코드(주석처리 되어 있음)의 주석을 풀어서 사용하시면 됩니다.

D. User Code 부분의 코드를 작성하신 후 서버에 제출하실 때, 디버깅을 위한 라이브러리

함수뿐 아니라 표준 입출력 함수를 모두 삭제 또는 주석 처리해 주셔야 합니다.

**④ 문제 내에 제약조건을 모두 명시하지 않으므로 주어지는 코드를 분석하셔야 합니다.**

**⑤ 코드는 개발 언어에 따라 상이할 수 있으므로, 작성할 언어를 기준으로 분석하셔야 합니다.**

**[문제 설명]**

연락처 App에 사용되는 DataBase를 만들려고 한다.

연락처 DataBase에 [Table 1]와 같이 3명의 정보가 저장되어 있다고 생각해보자.

연락처 정보의 각 항목을 **field**라 하고, 5개의 field가 모인 한 사람의 정보를 **record**라고 한다.

 연락처 DataBase에서는 [Table 2] 와 같이 5개의 field가 사용된다.

**연락처 DataBase가 제공해야 하는 5개의 API를 작성하라.**

|  |  |  |  |
| --- | --- | --- | --- |
| **InitDB()** | | | |
|  | void InitDB() | | |
|  | - DataBase를 초기화한다. | | |
|  | **Parameters:** | | |
|  | **Returns:** | | |
| **Add()** | | | |
|  | void Add(char\* name, char\* number, char\* birthday, char\* email, char\* memo) | | |
|  | - DataBase에 한 개의 record를 추가한다.  - record는 name, number, birthday, email, memo 5개의 field로 되어있다. | | |
|  | **Parameters:** | | |
|  | | name | : NAME field 값 |
|  | | number | : NUMBER field 값 |
|  | | birthday | : BIRTHDAY field 값 |
|  | | email | : EMAIL field 값 |
|  | | memo | : MEMO field 값 |
|  | **Returns:** | | |
| **Delete()** | | | |
|  | int Delete(FIELD field, char\* str) | | |
|  | - field값이 str인 record를 삭제한다. | | |
|  | **Parameters:** | | |
|  | | field | : field 정보 |
|  | | str | : 삭제하려는 record 의 field 값 |
|  | **Returns:** | | |
|  | | int | : 삭제한 record 의 개수. 삭제한 record 가 없으면 0 |
| **Change()** | | | |
|  | int Change(FIELD field, char\* str, FIELD changefield, char\* changestr) | | |
|  | - field값이 str인 record를 찾고, 해당 record의 changefield값을 changestr 로 변경한다. | | |
|  | **Parameters:** | | |
|  | | field | : field 정보 |
|  | | str | : field 값 |
|  | | changefield | : 변경해야 하는 field 정보 |
|  | | changestr | : 변경해야 하는 field값 |
|  | **Returns:** | | |
|  | | int | : 변경한 record 의 개수. 변경한 record 가 없으면 0 |
| **Search()** | | | |
|  | RESULT Search(FIELD field, char\* str, FIELD returnfield) | | |
|  | - field값이 str인 record를 찾고, record 개수와 record의 returnfield 값을 반환한다. | | |
|  | **Parameters:** | | |
|  | | field | : field 정보 |
|  | | str | : field 값 |
|  | | returnfield | : 반환 해야 하는 field 정보 |
|  | **Returns:** | | |
|  | | count | : field값이 str인 record 의 개수. |
|  | | str | : 해당 record의 returnfield 값.  count 값이 0 이거나, 2 이상일 경우 str은 무시된다. |
|  |  |  |  |

**[Table 3]**

**[제약 사항]**

1. 각 test case 의 최대 record 개수는 50,000개 이하이다.

2. 각 test case 시작 시에 InitDB() 함수가 호출된다.

3. 입력으로 주어지는 각 field 값은 make\_field() 함수에서 만들어지는 field와 동일한 포맷으로 만들어진다.

4. Delete(), Change(), Search() 함수의 경우, field값이 str인 record 개수가 여러 개 일수 있다.

**[예시]**

아래 코드는 예시를 위한 값으로 실제 입력 데이터와 다를 수 있습니다.

|  |  |
| --- | --- |
| 입력 | 결과 |
| Add(“A”, “111”, “0101”, “a.com”, “aaa”); |  |
| Add(“B”, “222”, “0202”, “b.com”, “bbb”); |  |
| Add(“C”, “333”, “0303”, “c.com”, “ccc”); |  |
| Add(“D”, “444”, “0505”, “d.com”, “ddd”); |  |
| Add(“E”, “555”, “0505”, “e.com”, “eee”); |  |
| result = Search(NAME, “A”, EMAIL); | result.count = 1;  result.str = “a.com”; |
| Change(NAME, “A”, EMAIL, “d.com”); | result = 1; |
| result = Search(NAME, “A”, EMAIL); | result.count = 1;  result.str = “d.com”; |
| Delete(NUMBER, “777”); | result = 0; |
| result = Search(BIRTHDAY, “0505”, MEMO); | result.count = 2;  result.str = “”; // result.str 값은 무시된다. |
| Delete(EMAIL, “d.com”); | result = 2; |
| result = Search(NAME, “A”, EMAIL); | result.count = 0;  result.str = “”; // result.str 값은 무시된다. |
| Change(BIRTHDAY, “0505”, MEMO, “zzz”); | result = 1; |
| result = Search(NAME, “E”, MEMO); | result.count = 1;  result.str = “zzz”; |

**[입력 / 출력]**

입출력은 제공되는 Main 부분의 코드에서 처리함으로 User Code 부분의 코드에서는 별도로 입출력 함수를 사용해서는 안 된다.

출력 결과는 아래와 같은 형태로 보여진다. (출력 값은 예시이며, 실제 출력 값과 동일하지 않을 수 있다.)

|  |
| --- |
| #1 1000  #2 1000  #3 1000  #4 1000  #5 1000  TotalScore = 5000 |