Notepad Plus

벨코즈는 메모장을 사용하고있다. 하지만 반복된 단어를 입력할 때 불편함을 느끼고있다.

벨코즈는 Notepad Plus 제작을 의뢰했다. 기능은 아래와 같다.

1. 입력은 알파벳 소문자만 사용한다. 소문자 입력 시 커서가 우측으로 한 칸 이동한다.
2. 벡스페이스를 하면 글자를 지우고 커서를 좌측으로 한 칸 이동한다.
3. 스페이스를 기준으로 단어로 인식한다.
4. 단어를 입력하는 중 탭을 할 경우 기존에 입력된 단어를 찾아준다.
   A. abcd abcee eabde abcdeab 라고입력 한 후 ab하고 탭을 할 경우 ab를 부분문자로 시작하는 단어는 abcde abcee abcdeab 이고 공통문자열까지만 자동완성해준다. 그러므로 abc까지만 자동완성된다.

함수	기능
void input(char c);	1. a~z 입력시 현재커서에서 입력되고 커서가 우측으로 한칸 이동한다. 2. '₩t' 입력시 현재까지 입력된 단어들 중에서 현재 입력중인 글자를 부분문자열로 가지는 문자열로 자동완성. 3. ' ' 입력시 시작점 or 이전 공백위치로부터 현재까지 입력된 문자들을 단어로 인식하게 되고 새로운 단어의 시작이 된다. 4. '₩b' 입력시 현재커서에서 한글자 지우고 커서 좌측으로 이동한다. ' '를 입력한 후에는 '₩b' 이 들어오지 않는다. (이전에 인식된 단어를 침범하는 경우는 없다, 즉, 최근 빈 공백 까지만 지울 수 있다.)
Void find(char* str, RESULT* result);	전체 텍스트에서 Str과 일치하는 단어를 찾는다. 구조체는 아래와 같이 정의되어 있다. Typedef struct{ Int count; Int startIndex[50000]; }RESULT; Count는 찾은 단어들의 개수 이다. startIndex는 전체 텍스트에서 찾은 단어의 시작 위치를 저장할 배열이다.

<table>
<tr><td></td><td>부분문자열은 찾지않는다
예를들면 abcde 는 bcde의 검색에 해당하지 않는다. 완전 일치하는 경우에만 해당한다.
<table><tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr><tr><td>a</td><td>b</td><td>c</td><td></td><td>d</td><td>e</td><td></td><td>a</td><td>b</td><td>c</td></tr></table>
위 텍스트에서 abc를 검색할 경우
Count = 2
startIndex[0] = 0
startIndex[1] = 7

의 데이터를 입력하여 리턴하면된다.</td></tr>
</table>

추가사항

텍스트는 총 50000까지 입력된다.

단어의 최대 길이는 20이다.

입력

출력

Input	Output
1 //tc개수 0 a //일반문자 0 b 0 c 0 + //space 0 a 0 > //tab 0 - //backspace 0 + 0 a 0 b 0 +	2 //find결과 count 4 //find결과 startIndex 7 //find결과 startIndex

1 ab //검색 2 //tc종료	

Main code

```
#include <stdio.h>

typedef struct {
        int count;
        int startIndex[50000];
}RESULT;

void input(char c) {
}


void find(char* str, RESULT* result) {

}
void init() {

}
int main() {
        cursor = 0;
        RESULT outputData;
        int tc;
        int cmd;
        char data;
        char str[100];
        scanf("%d", &tc);
        for (int n = 0; n < tc; n++) {
                while (1) {
                        scanf("%d", &cmd);
                        if (cmd == 0) {
                                scanf(" %c", &data);
                                input(data);
                        }
                        else if (cmd == 1) {
                                scanf(" %s", str);

                                find(str, &outputData);

                                printf("%d₩n", outputData.count);

                                for (int i = 0; i < outputData.count; i++) {
                                        printf("%d₩n", outputData.startIndex[i]);
                                }
                        }
                        else if (cmd == 2) {
                                break;
                        }

                }
```

```
                init();
        }

        return 0;
}
```