

SW 프로그래밍(2)

Algorithm & Solution

20175528 이민지

```
operation == "MIRROR_X":  
    mirror_mod.use_x = True  
    mirror_mod.use_y = False  
    mirror_mod.use_z = False  
operation == "MIRROR_Y":  
    mirror_mod.use_x = False  
    mirror_mod.use_y = True  
    mirror_mod.use_z = False  
operation == "MIRROR_Z":  
    mirror_mod.use_x = False  
    mirror_mod.use_y = False  
    mirror_mod.use_z = True
```

```
#selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.name))  
mirror_ob.select = 0  
= bpy.context.selected_objects  
data.objects[one.name].select  
print("please select exactly one")
```

목차

1 폴더 목록 내의 파일 열기

2 단어가 있는 파일 찾기

3 랭킹 매기기

4 출력 하기

5 총 정리

<폴더 목록 내의 파일 열기>

io.h의 헤더 파일을 이용
그 중의
_findfirst()와
_findnext(),
_findclose()
함수를 이용한다.

_findfirst()를 이용하여 우선
첫번째 파일 하나를 찾는다.



_findnext()를 이용하여 차례
로 파일을 찾는다.




_findclose()를 이용하여 메모
리를 해제해준다.

〈단어가 있는 파일 찾기〉

원하는 단어를 입력받는다.(변수 str에 저장한다.)



feof를 이용하여 파일 끝에 도착할 때 까지 파일 내의 단어(Ex. Word)를 받는다.(fscanf이용)



strstr함수를 이용하여 str과 파일에서 뽑아낸 단어가 일치하는지 비교한다.

〈단어가 있는 파일 찾기〉

strstr 함수를 이용하여 str과 파일에서 뽑아낸 단어가 일치하는지 비교한다.

=> if(strstr(word, str))!=NULL이라는 조건 이용.

=> 대문자와 소문자를 동일하게 처리한다.

if조건 내에서 COUNT를 한다.

=> Char형 배열과 int형 배열을 포함한 구조체를 만들어 파일 순서대로 구조체에 파일 이름과 count를 저장한다.

=> COUNT > 0 이면 일치하는 단어가 있는 경우

=> COUNT = 0 이면 일치하는 단어가 없는 경우

예시)

```
typedef struct hello{  
    int count;  
    int size;  
    char name[50];  
}Hello;
```

<랭킹 매기기>

일치하는
단어가
있다면?

해당 단어가 파일에 있는 횟수가
높을 수록 앞쪽에 랭킹.

일치하는
단어가
없다면?

해당 단어와 파일의 단어 유사도가
높을수록 앞쪽에 랭킹.

⇒ 유사도 구하기 : strlen을 이용하여 구한 입력된 단어의 길이(n)
를 이용하여 strncmp에 n-1, n-2.....0의 순서로 그 값을 매긴
다. 이때, 일치하는 size가 클수록 유사도가 높아지게 된다.
⇒ 구조체에 size를 입력한다.

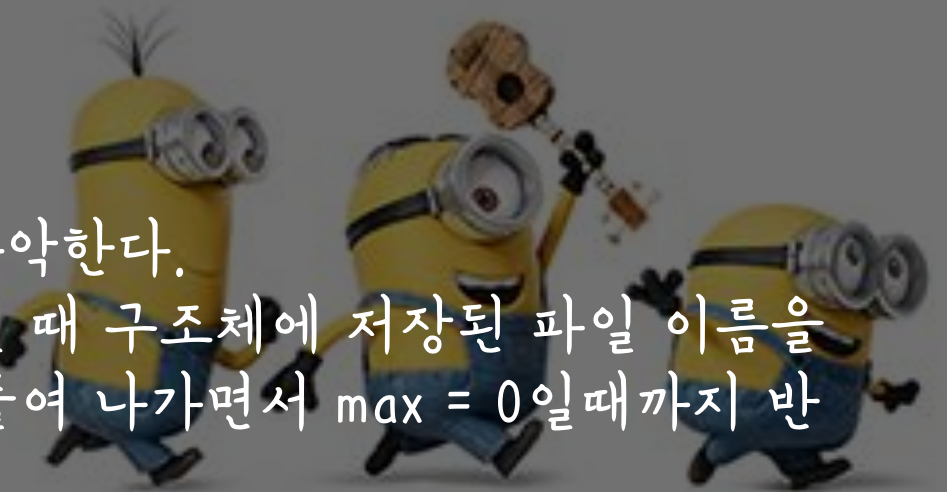
<출력 하기>

해당 단어가 파일에 있는 경우.

- ⇒ 구조체의 count배열에 저장된 값 중 최대값을 파악한다.
- ⇒ 최대값을 max라 하면 if(구조체.배열 == max)일 때 구조체에 저장된 파일 이름을 통해 해당 파일을 출력하도록 한다. Max값을 줄여 나가면서 max>0일때까지 반복한다.

해당 단어가 파일에 없는 경우.

- ⇒ 구조체의 size배열에 저장된 값 중 최대값을 파악한다.
- ⇒ 최대값을 max라 하면 if(구조체.배열 == max)일 때 구조체에 저장된 파일 이름을 통해 해당 파일을 출력하도록 한다. Max값을 줄여 나가면서 max = 0일때까지 반복한다.



<총 정리>

io.h 헤더파일 내의 함수를 이용하여 파일을 오픈한다.



원하는 단어를 입력 받는다.

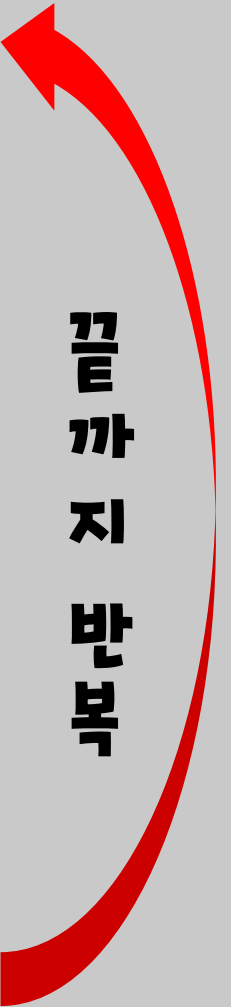
strstr 함수를 이용하여 입력 받은 단어와 파일내의 단어를 비교하고 일치할 때 마다 count를 올린다.

파일의 순서대로 구조체에 파일 이름과 count를 저장한다.



io.h 헤더파일 내의 함수를 이용하여 파일을 닫는다.

끝
까
지
반
복



<총 정리>



모든 파일의 COUNT 파악이 끝나면 COUNT에 따라

Count > 0



count 값이 클수록 기존 파일의
내용이 앞쪽에 출력되도록 한다.

Count = 0



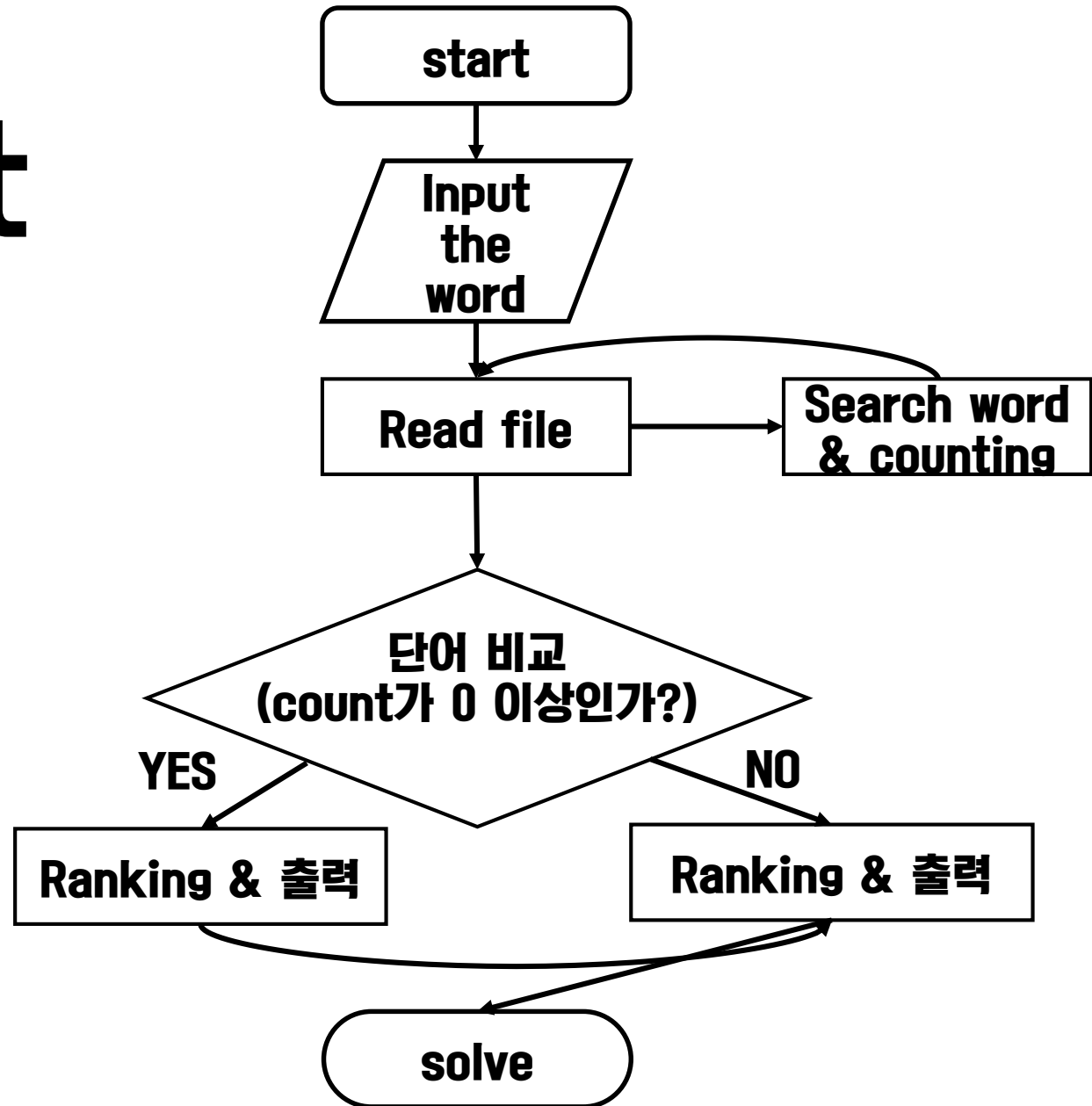
strncmp 함수를 통해 유사도가
높을수록 앞쪽에 출력하도록 한다.

Count > 0 인 경우가 끝나면



〈총 정리〉

- flow chart





감사합니다!!