① string 포인터

```
#include <string.h>

char *m_strlen(char* str1, char*str2){

        if (strlen(str1) > strlen(str2)

            return str1
        return str2
```

str1 자체가 string이다.

여기서 *str1은 char를 의미한다,

⇒문장길이가 더 긴 string 반환

```
char str1[80];
```

② 띄어쓰기 받기 →

```
scanf('%[^\n]s', str1)
    fgets(str1, 80, stdin)
```

참고로 스트링 자체가 배열 → 포인터 개념이니까 &가 안붙음.

[illegible]

↳ 띄어쓰기를 한다.

정규표현식으로 \n이 나올때까지 문자를 입력받겠다.

③

```
inyl str[80];
int num = 0
while(1){
     printf("Input string(exit, if input 0)");
     scanf("%[^\n]s", str);
     if (strlen(str)==1 && str[strlen(str)-1] == '0')
     return0;
```

↳s는 생략가능 이게 더 잘 읽힘.

엔터 나올때까지 계속 받는다

나는 str[80]이니까 크기찰때까지 실행하고 생각했는데,

그렇지 않? While은 돌때마다 renewal 됨.