숙제 1 : 나만의 String 만들기

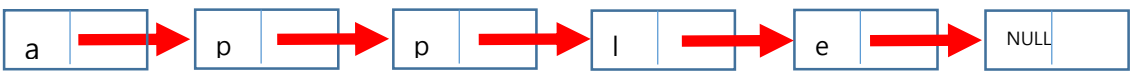
<Due : 2020-04-30 (목) 23:59>

C언어에서 string은 char의 array (포인터)를 사용하여 구현할 수 있었다. 이번 숙제에서는 **string을 linked list로 구현**하여 실제 string.h 에 있는 각종 함수들을 직접 구현해 보는 것을 목표로 한다. my_string 은 **맨 마지막 NULL 문자를 포함하여** 각각의 node가 문자 하나를 저장하고 있는 linked list 로 정의되며 (이 때 문자는 NULL 문자를 포함하여 **알파벳 대문자 + 소문자 + 0-9 의 아라비아 숫자 + 공백 문자로 구성되어 있다** 가정하자), 다음과 같은 형식으로 되어 있다.

```
//node type
typedef struct node {
    char character;
    struct node* next;
}node;

//my_string type
typedef struct my_string {
        struct node* first_symbol;
}my_string;
```

예를 들어 my_string apple 은 다음과 같은 형식으로 저장되어 있다.



여기서 a node는 first_symbol이 되고 NULL node의 next는 NULL pointer가 된다.

2. 구현해야 하는 함수들

(주의 : 교재 슬라이드에 나온 code는 예외 처리가 안된 부분이 많으므로, 숙제에서는 언급한 예외사항을 처리하는 것에 신경 쓸 것)

2-1 my_string to_mystring (char * string)

: C언어에서 string을 받아 my_string 형식으로 바꿔주는 함수.

2-2 void print_mystring (my_string str)

: my_string type 로 저장된 str 을 출력하는 함수.

Ex) print_mystring (to_mystring ("apple")) → apple를 출력.

2-3 int my_strlen(my_string str);

: str의 길이를 반환한다 (길이는 1부터 시작). 이 때 마지막 NULL문자는 포함하지 않는다.

Ex) my_strlen (to_mystring ("apple")) = 5

2-4 my_string my_strcpy(my_string destination, my_string source);

: NULL 문자를 포함한 source의 string을 마지막 NULL 문자를 포함하여 destination에 복사 후 destination을 리턴. destination의 길이가 더 작을 경우 **에러 메세지를 출력한 후**, 임의의 (아무) my_string 을 리턴한다 (즉 아무 my_string이나 리턴해도 무방).

2-5 my_string my_strcat(my_string destination, my_string source);

: destination의 끝에 source을 더한 후, destination을 리턴한다.

2-6 int my_strcmp(my_string str1, my_string str2)

: 두 string str1와 str2를 비교한다. 어느 하나가 NULL문자가 나올 때까지 비교한 후, 두 string이 정확하게 **일치하면 0을 리턴**, 일치하지 않을 경우 str1이 str2보다 사전적 순서가 **앞쪽이면 -1, 뒤쪽이면 1을 리턴** 한다.

참고) 사전적 순서 : **숫자는 알파벳보다 앞**, 알파벳은 기본적으로 **순서를 따르되 대문자는 소문자보다 뒤**. Ex: 3 < a < A < c < D < e…

Ex) my_strcmp(to_mystring ("appple"), to_mystring ("apple"))) = 1

my_strcmp (to_mystring ("apple"), to_mystring ("Apple"))) = -1

my_strcmp (to_mystring ("apple"), to_mystring ("3Apple"))) = 1

my_strcmp (to_mystring ("apple3"), to_mystring ("apple"))) = 1

2-7 int my_strchr(my_string str, int character);

: str에서 character (이 때 character는 int로 받지만 내부에서 아스키 코드에 의해 문자로 변경) 이 나오는 첫번째 위치를 반환한다 (위치는 0부터 출발). NULL문자도 character에 포함될 수 있음. 해당하는 문자가 없으면 str1의 길이를 출력한다.

Ex) my_strchr(to_mystring ("apple"), 112)) = 1 ← 112는 'p' 의 아스키 코드.

my_strchr(to_mystring ("apple"), 0)) = 5

2-8 int my_strcspn(my_string str1, my_string str2);

: str2에 포함된 문자들 중에서 str1에 '처음으로' 나오는 문자를 찾은 뒤, str1 상에서 그 위치를 반환한다. 만약 일치하는 것이 없으면 str1의 길이를 출력한다.

Ex) my_strcspn (to_mystring ("apple"), to_mystring ("epa"))) = 0

my_strcspn (to_mystring ("apple"), to_mystring ("lbcd"))) = 3

my_strcspn (to_mystring ("apple"), to_mystring ("wxyz"))) = 5

2-9 int my_strstr(my_string str1, my_string str2);

: str1에서 str2가 처음으로 나타나는 곳에 위치를 리턴한다. str1이 str2를 포함하지 않으면 -1을 리턴한다.

Ex) my_strstr(to_mystring ("apple"), "app")) = 0

my_strstr(to_mystring ("banana"), "ana")) = 1

my_strstr(to_mystring ("apple"), "ape")) = -1

2-10 my_string my_strrev(my_string str);

: 마지막 NULL 문자를 제외한 str을 뒤집은 후, 맨 뒤에 NULL 문자를 추가한 my_string을 리턴한다.

Ex) print_mystring (my_strrev (to_mystring ("apple"))) → elppa 출력

3. 주의 **(이 중 하나라도 어기면 0점)**

- 주어진 파일은 다음과 같은 형식으로 되어 있음

1. myString.h : myString.c 에서 정의한 함수들을 모아 둔 헤더파일.
2. **myString.c : myString.h 에서 정의한 함수들의 세부 구현 (숙제에서 제출해야 할 파일).**
3. main.c : 테스트용 main file (결과값이 주석에서 언급한 답과 일치하는지 확인할 것.
4. Makefile : 리눅스, 맥 환경에서 작업하는 학생들을 위한 Makefile.

- 숙제 제출은 due 전까지 e-campus의 과제탭의 **과제물 제출 – 파일 첨부를 이용**하여, 오직 **myString.c 파일만을 학번_myString.c 파일로 업로드** 할 것. (예 : 2019000000_myString.c). **그 외 어떠한 파일도 받지 않음**.

- 숙제의 delay는 받지 않음.

- string.h 포함 문제의 myString.h와 myString.c 두 파일에서 skeleton code 에서 주어진 헤더 파일 외의 **어떠한 헤더 파일도 include 하지 말 것**.

- 주어진 myString.h 에 정의된 함수만을 구현하고, **자기가 따로 함수를 정의 및 구현하여 myString.h 및 myString.c 에 추가하지 말 것 (**채점 시 원래 주어진 myString.h 로 채점한다는 것을 잊지 말것).

- 제출하기 전에 예제로 주어진 main.c 파일을 컴파일 및 실행해서 실제로 올바르게 출력되는지 확인할 것.