Programming Assignment #0:

Making own "my_string.h"

Due: 21st Mar. (Mon), 11:59 PM

1. Introduction

이번 과제에선, 앞으로 있을 다른 과제들을 수행하기 위한 필요할 함수들을 구현한다. 그 대상은, 문자열 조작/검사/변환 함수들을 담은 C 표준 라이브러리의 <string.h>를 비롯한 헤더 파일의 일부 함수이다. 그리고 본 과제의 다른 목적은 학생이 각자 독자적인 Linux 환경을 갖추는 것이다. 또한, 간단한 문자열 및 포인터 조작을 통해 향후 원활한 과제 수행을 돕고자 한다.

2. Problem specification

본 과목에서 과제를 수행할 때, Linux 시스템 콜과 같은 일부 특수한 함수들을 제외하곤 C 표준 라이브러리나 다른 라이브러리 함수들을 사용하는 것이 허락되지 않는다. 따라서 자신이 필요한 함수들을 직접 구현해야 하는데, C 표준 라이브러리의 <string.h>를 비롯한 문자열을 이용하는 함수들은 앞으로 거의 모든 과제에서 필요할 것이기 때문에, Linux 환경에서 이 함수들을 직접 구현해본다. 또한, 향후 과제에서 본인이 필요한 함수는 **my_string.c** 및 **.h**에 구현하여 사용할 수 있다.

구현할 함수와 각 함수의 기능은 다음과 같다. 각 함수들은 C 표준 함수들이 제공하는 기능과 100% 동일하게 구현하면 되고, 다른 기능을 추가할 필요는 없다.

(C 표준 함수의 경우, 정확한 함수 정의를 찾는 방법은 수업시간에 설명한 대로 리눅스 매뉴얼 페이지 man을 사용하도록 한다. 밑줄을 친 함수들은 C 표준 함수가 아님을 뜻하고, 해당 함수들에 대한 정의는 **Section 2.1.** 에서 설명한다.)

strtok_r을 구현하지 않아도 되나, 구현할 시에는 추가 점수를 받을 수 있다.

Conversions string to numeric formats:
int my_*atoi* (const char *nptr);
- 문자열 str을 정수형 int로 변환하여 출력
long my_*atol* (const char *nptr);
- 문자열 str을 정수형 long으로 변환하여 출력

Conversions numeric formats to string:
char *int2str (char *dest, int num);
- 정수형 num을 문자열로 변환한 뒤 문자열 버퍼 dest에 저장
- [int:-543210 → C string:"-543210"]

String manipulation:
char *strcpy (char *dest, const char *src);
- 문자열 src를 문자열 버퍼 dst로 복사
char *strncpy (char *dest, const char *src, size_t n);
- 문자열 src에서 count 바이트만큼 문자열 버퍼 dst로 복사
char *strcat (char *dest, const char *src);
- 문자열 dst의 뒤에, 문자열 src을 붙임
char *strncat (char *dest, const char *src, size_t n);
- 문자열 dst의 뒤에, 문자열 src를 count 바이트만큼 붙임
char *strdup (const char *str);
- 문자열 str을 동적으로 할당 받은 메모리에(malloc 등) 복사한 뒤 해당 주소를 반환

char *int2str (char *dest, int num);

- 숫자 num을 C 문자열 형태로 바꿔 dest에 저장하고, 변환된 문자열을 반환한다. (일반적인 경우, 반환값은 dest가 될 것이다.)
- 즉, 바뀐 문자열이 반환된다는 점을 제외하면, 다음 함수 호출과 동일한 행동이 수행되어야 한다: sprintf(dest, "%d", num);
- 예외) 인자 dest가 NULL일 경우, 동적으로 메모리를 받고 해당 주소에 변환한 문자열을 저장한 다음, 그 주소를 반환한다. 동적 메모리 할당에 실패하면 NULL을 반환한다.
- 힌트) int형을 문자열로 변환할 경우, 최악의 경우 NULL 캐릭터를 포함하여 char[12] 버퍼가 필요하다.

String examination:

`size_t strlen (const char *s);`
- 문자열 s의 길이를 반환

`int strcmp (const char *s1, const char *s2);`
- 문자열 s1, s2를 비교

`int strncmp (const char *s1, const char *s2, size_t n);`
- 문자열 s1, s2를 최대 n 바이트만 비교

`char *strchr (const char *s, int c);`
- 문자열 s에서 문자 c가 처음 나타나는 위치를 찾음

`char *strrchr (const char *s, int c);`
- 문자열 s에서 문자 c가 마지막으로 나타나는 위치를 찾음

`char *strstr (const char *haystack, const char *needle);`
- 문자열 haystack에서 부분문자열 needle이 처음으로 등장하는 위치를 찾음

`char *strtok (char *str, const char *delim);`
- 문자열 str에서 '문자열 delim의 아무 문자'가 등장하는 위치를 찾고, token화함

`char *strtok_r (char *str, const char *delim, char **saveptr);`
- strtok 함수와 동일하지만 다음 token을 처리할 위치를 담는 saveptr 변수를 사용

Character array manipulation:

`void *memcpy (void *dest, const void *str, size_t n);`
- 메모리 주소 str에서 n 바이트만큼 메모리 주소 dest로 복사

`void *memset (void *dest, int ch, size_t count);`
- 메모리 주소 [dest. dest+count)의 값을 ch 로 변경

`char *strdup (const char *str);`

- 동적으로 메모리를 할당 받고, 해당 주소에 문자열 str의 내용을 복사한 뒤, 해당 주소를 반환한다.
- 예외) 메모리를 동적으로 할당 받을 수 없다면, NULL을 반환한다.
- 참고) `$ man 3 strdup`

strtok_r 은 구현할 시 선택사항으로, 성공하면 추가 점수가 있다.

`char *strtok_r (char *str, const char *delim, char **saveptr);`

- 기본적으로 strtok 함수와 동일하지만, 연속적으로 strtok[_r] 함수를 호출할 때 차이점이 있다. strtok 함수는 함수 내부의 버퍼를 두어 다음에 처리할 주소를 저장하지만, strtok_r 함수는 내부의 버퍼 대신 saveptr에 주소를 담아 그 주소를 사용한다는 차

이점이 있다.

- str이 존재하면 첫째 토큰을 처리하고, 다음부터 처리할 주소를 saveptr에 담은 뒤, 토큰의 주소를 반환한다.
- str이 NULL이라면, saveptr에 담긴 주소를 가져와 다음 토큰을 처리하고 다음에 처리할 주소로 saveptr을 업데이트한 뒤, 토큰의 주소를 반환한다.
- 더 이상 토큰을 생성할 수 없으면 NULL을 반환한다.
- 참고) $ man 3 strtok_r
- 참고) saveptr에 담길 주소에 정답은 없으니 스스로 판단한다.

3. Skeleton codes

이번 과제 수행을 위해 다음 3개의 파일이 주어진다:

my_string.h: 본 과제에서 지정한 함수들이 들어있는 헤더 파일
my_string.c: 본 과제에서 지정한 함수들의 구현 파일
test.c: 구현한 함수들의 디버깅용 파일

Skeleton codes 들은 전부 C library 함수를 단순히 wrapping하는 꼴로 이루어져 있으며, 이를 자신만의 코드로 바꾸어야 한다.

4. Verification of your code

제공한 test.c 파일을 이용해 각자 자유롭게 자신들이 만든 함수가 라이브러리 함수와 동일한지 비교할 수 있다. 예시는 아래와 같다.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "my_string.h"

int main()
{
	char str1[20] = "Hello";
	char str2[20] = "world!";

	printf("<strlen> str1: %zu\n", strlen(str1));
	printf("<strlen> str1: %zu\n", my_strlen(str1));
	...

	return 0;
}
```

5. Background

5.1. Man page

본 과목의 과제를 수행하며 man page (manual pages)를 참고할 수 있다. man page는 Unix-like 환경에서 제공하는 참고용 매뉴얼 프로그램이다. 어떤 함수의 정의나 프로그램의 역할이 궁금하면 셸에서 다음 커맨드를 입력함으로써 정보를 찾아볼 수 있다.

```
$ man <COMMAND_OR_FUNCTION_NAME>
```

예를 들면, `strlen` 함수에 대한 정의를 찾아보고 싶을 땐, `"$ man strlen"` 을 입력한다.

`printf`와 같은 함수는 동명의 프로그램과 라이브러리 함수가 같이 존재하는데, 이럴 경우엔 라이브러리 섹션을 지정해서 `man` 프로그램을 호출한다. (참고: http://en.wikipedia.org/wiki/Man_page)

```
$ man 3 printf
```

만약 `man` 프로그램이 제대로 동작하지 않을 경우, Ubuntu의 경우 다음 명령어를 입력함으로써 라이브러리에 대한 매뉴얼을 다운로드할 수 있다. 다음 명령 수행엔 관리자 권한이 필요하다.

```
$ sudo apt-get install manpages-dev manpages-posix-dev
```

5.2. C References

C 라이브러리 함수에 대한 reference는 다음 사이트를 참고한다:

- http://www.cplusplus.com/reference/clibrary/
- http://en.cppreference.com/w/

6. Restrictions

함수 구현 시 `malloc()`, `calloc()`, `free()` 함수를 제외한 다른 라이브러리 함수는 **사용할 수 없다**. 필요하다면, 직접 구현하여 사용한다.

7. Hand in instruction

- 작성한 **코드 상단의 주석에 이름과 학번을 작성한다.**
- 본 과제 수행 시 구현 방법과 디자인을 설명하는 보고서를 PDF 포맷으로 작성하여 "README.pdf" 이란 이름을 붙인다. (가능하면 PDF가 가장 좋지만, 대중적인 문서 포맷은 다른 포맷도 괜찮음)

- 보고서에는 본인이 구현한 Linux 환경에 대한 스크린샷을 첨부한다.
  - Linux 셸에서 다음 3가지 명령을 실행시킨 결과를 같이 첨부한다.

```
$ cat /proc/version
$ cat /etc/issue
$ lsb_release -a
```

- test.c 파일을 제외하고, my_string.c, my_string.h, README.pdf, screenshot 총 4개의 파일을 tar로 압축한다. 이 때, 압축하는 파일이름은 반드시 본인 학번으로 한다.

Ex) 학번이 2014123456 인 경우,

```
$ tar -cvzf 2014123456.tar.gz my_string.c my_string.h README.pdf screenshot
```

- 과제를 제출하기 위해 [dylee@csl.skku.edu] 주소로 메일을 보낸다. 메일 전송 시 압축한 tar 파일 하나만 첨부하도록 하며, 메일 제목은 다음과 같이 명명한다:

[SSE2033] 학번 이름 PA#0

[SSE2033] 2014123456 홍길동 PA#0

8. Logistics

- 본 과제는 혼자 수행한다.
- **질문 또한 dylee@csl.skku.edu에 메일로 보내며, 메일 제목 앞에는 필히 [SSE2033]을 쓰도록 한다.**
- **제출 상태는 과목 홈페이지 http://csl.skku.edu/SSE2033S16/Projects 에 즉각적으로 공지될 것이다.**
- 과제 제출 시간은 메일 도착 시간을 기준으로 하며, 과제를 지연 제출하면 기한 직후부터 매 **8시간마다** 점수를 10%씩 추가로 감점한다.
- 다른 사람의 과제를 copy할 경우, 개입한 사람 전부 해당 과제에 대해 0점 처리되고, 교수님께 보고되며, **성적 산정에 불이익이 있다**. 또한, copy가 두 차례 이상 적발될 경우 **F 학점이 부여될 수 있다**.

Warming up!

Dong-Yun Lee, TA

Computer Systems Laboratory, Sungkyunkwan Univ.