C프로그래밍 실습

**#12 File Input/Output**

* input and output

데이터를 하나에서 다른 것으로 전송한다.

* Example of I/O in cmd

+ input

scanf(): 키보드 입력 형식

gets(): 한 줄 (Enter키 까지) 키보드 입력

getchar(): 한 글자 키보드 입력

+ output

printf(): 스크린 출력 형식

puts(): 한 줄 스크린 출력

putchar(): 한 글자 스크린 출력

* single-Character I/O

+ getchar()

키보드 입력에서 한 문자를 반환한다.

반환 데이터 유형은 int (char)이다.

char 데이터형은 하나의 문자에만 사용된다 (정수에서 0~255)

예외가 발생시 프로그램에 알리는 방법이 있어야한다: int사용 (더 넓은 범위)

+ putchar()

한 글자만 스크린에 출력

* The gets() Function

문자열을 얻는다.

c.f. char temp[10];

+ get(temp)는 scanf(“%s”,temp)와 같지 않다.

scanf(“%s”,temp); 는 하나의 글자만 얻는다.

gets(temp);는 한 줄을 얻는다. (여러 개의 글자로)

+ return values

get()은 사용자가 유효한 문자열을 입력한 경우 temp에 대한 포인터를 반환한다.

scanf()는 사용자가 입력한 항목의 수를 반환한다.

* The puts() Function

+ cmd에서 유저에게 문자열을 준다.

c.f. char temp[10]=”Hello”;

puts(temp)는 printf(“%s\n”,temp);와 같다.

puts()는 문자열을 표시 할 때 자동으로 줄 바꿈을 추가한다.

* Pointer to a string

+ index안에 있는 문자열의 글자

+ ex) char temp[6] = “Hello”;

temp[1]은 배열temp의 첫번째 값이다.



“temp”는 시작메모리 주소이다

&temp[1]은 temp[1]의 메모리 주소이다.

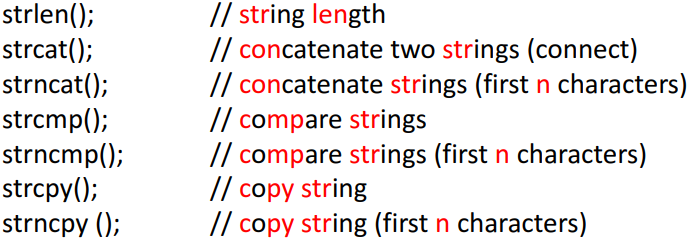
&temp[1]은 temp의 첫번째 요소인 부분 문자열이다.

“temp+1”은 &temp[1]과 같다.

* String Functions

C 표준 라이브러리는 printf() 및 scanf()와 같은 몇가지 기본 함수 뿐 만 아니라 여러 문자열 조작 함수를 제공한다.

#include <string.h> 헤더파일을 추가하면 다음 참조를 사용할 수 있다.



* Buffers

사용자에 의한 입력은 일부 임시 공간에 저장되고 그 이후에 처리된다.

(flushed to program) -> 이것을 buffer라고 부른다.

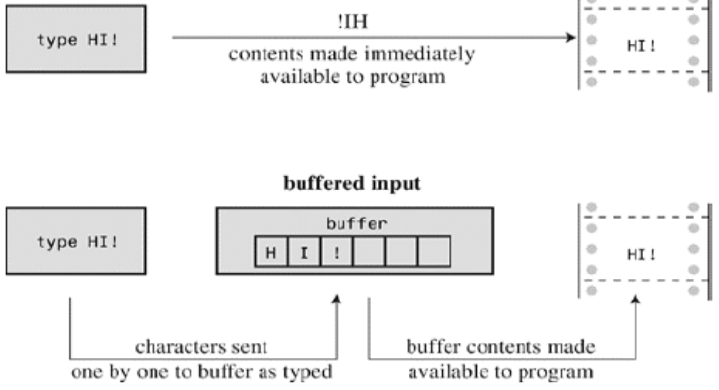
두종류의 input / output buffer가 있다.

+ fully buffered I/O

buffer가 가득 차면 flushed된다.

+ Line buffered I/O

줄 바꿈 문자 (‘\n’)이 나타날 때 마다 buffer가 비워진다.



* What is a File in computers?

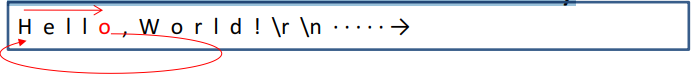
+ file은 일반적으로 하드 드라이브에 있는 명명 된 저장소이다.

programs, documents, data…

+ program이 요구하는 것? create, write, read..

* How is a file stored in computer?

+ 개별적으로 읽을 수 있는 연속적인 bytes의 순서



+ 카세트 테이프처럼 작동한다.

play, stop, rewind

한번 byte가 읽어지면, 반드시 되감기 해야 하며, byte가 사용 가능해 질 때 까지 기다려야한다.

* The file pointer

+ Declaration of a file input and output

FILE 구조체는 한 파일의 포인터이다

= declaration file은 하드 드라이브의 각 파일 저장소를 조작하기위한 핸들

= 포인터로 선언해야하며 FILE과 파일 포인터 이름 사이에 \*를 넣는다.

* FILE \*file\_name;
* the fopen() and fclose() function

+ initialization of a file input and output

fopen()은 파일은 read, write하기 위해 파일을 연다.

반드시 프로그램의 끝에 파일을 닫아주어야 한다, (use fclose())

* fopen(char \*file name, char \*mode);

file name과 mode모두 문자열이다.

반환 값 : 파일의 파일 포인터, 파일 열기에 실패하면 “NULL”

* fclose(FILE \*fp);

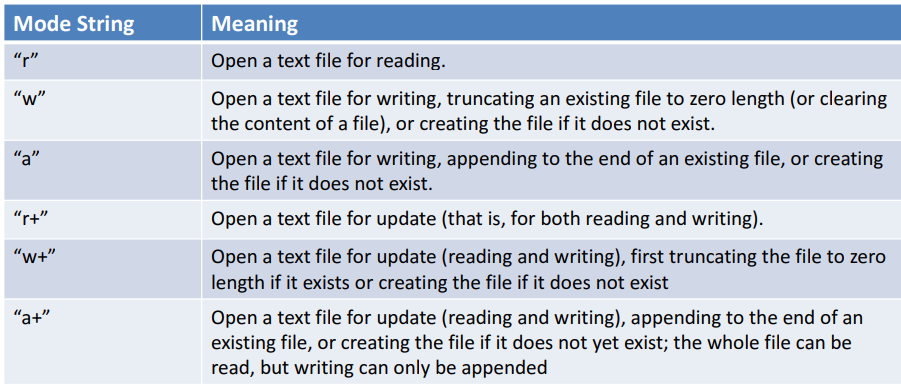
+ Two mode of operations

Text only

Binary

+ 다음을 텍스트 모드로 파일에 쓰면 문자열로 “1234”이다.





* the getc() and putc() functions

+ getchar () 및 putchar()의 파일 입출력 버전이다.

파일에서 문자를 읽거나 쓰는 경우, 파일의 현재 위치는 카세트 테이프처럼

한 문자씩 진행된다.

+ input

cmd창에서 사용자에게 문자를 받는다 ch = getchar()

file 포인터 fp에의해 식별된 파일로부터 문자를 받는다 ch = getc(fp);

+ ouput

cmd창에서 사용자에게 문자를 보여준다. putchar()

file 포인터 fp에 의해 식별된 파일으로 문자를 보여준다. putc(ch, fp);

* End-of File

+ 테이프와 비슷하게 파일에서 데이터를 “읽는” 프로그램은 파일의 끝에 도달하면 중지 해야한다.

+ 프로그램이 끝까지 도달했는지 어떻게 알 수 있을까?

파일 입출력 함수에 달려있다.

getc()함수는 문자를 읽으려고 시도하고 파일의 끝에 도달했음을 발견하면

특수값 “EOF”를 반환한다.

* getc()의 반환데이터 유형은 int
* The fprintf() and fscanf() Functions

+ printf() scanf()의 파일 입출력 버전

cmd에서

printf(): 메시지 출력 / scanf(): input을 받는 함수

fopen()을 사용한 file의 경우

fprintf(): file에 메시지 write / fscanf(): file로부터 메시지를 read

* FILE \*fp;
* fprintf(fp, “ ”, ..);
* fscanf(fp,” ”, …);

마지막에 특수문자 ‘\0’이 없다.

대신 새로운 줄을 추가해서 ‘\n’이 있다.

* the fgets() and fputs() functions

gets()와 puts()의 파일 입출력 버전

fputs() : file에 새로운 줄을 쓴다.

fgets() : file에서의 한 줄을 읽는다.

* FILE \*fp;
* fputs(“ ”,fp);
* fgets(str,len,fp);

str: pointer to string / len: 읽을 문자들의 최대 숫자.

* the rewind() function

파일 입력 및 출력의 현재 위치를 파일의 시작 부분으로 되감는다.

* FILE \*fp;
* rewind(fp);