2010-1학기 프로그래밍입문(1)

11장 입출력과 운영체제

박종혁

Tel: 970-6702

Email: jhpark1@snut.ac.kr

printf()

- 특징
 - 임의의 개수의 인자 출력
 - 간단한 변환 명세나 형식을 사용한 출력 제어

printf()

`printf(control_string, other_argument)`

- 예

```
printf("she sells %d %s for $%f", 99, "sea  
shells", 3.77);
```

control_string: "she sells %d %s for \$%f"

other_arguments: 99, "sea shells", 3.77

- 변환 명세는 %로 시작하여 변환문자로 끝남

printf()

printf() 변환 문자	
변환 문자	대응되는 인자의 출력 형태
c	문자
d,i	10진 정수
u	부호 없는 10진 정수
o	부호 없는 8진 정수
x, X	부호 없는 16진 정수
e	부동 소수점 수; 예: 7.123000e+00
E	부동 소수점 수; 예: 7.123000E+00
f	부동 소수점 수; 예: 7.123000
g	e형식과 f 형식 중 짧은 쪽
G	E형식과 f 형식 중 짧은 쪽
s	문자열
p	대응되는 인자가 void 포인터임; 그 값이 16진수 형태로 출력됨
n	대응되는 인자는 정수형 포인터로서 그 값은 현재까지 출력된 문자의 개수임; 인자는 변환되지 않음
%	%%의 형식으로 %를 출력 스트림에 씀; 대응되는 인자는 없음.

printf()

- 예제

```
printf("she sells %d %s for $%f", 99, "sea  
shells", 3.77);
```

변환 형식	대응되는 인자
%d	99
%s	"sea shells"
%f	3.77

printf()

- %와 변환문자 사이에 올 수 있는 것들
 - 플래그 문자들
 - 빼기 기호
 - 더하기 기호
 - 공백
 - # 기호
 - 0
 - 필드 폭
 - 정밀도
 - h 또는 l
 - L

printf()

선언과 초기화			
<code>char c = 'A', s[] = "Blue moon!";</code>			
변환 형식	대응되는인자	필드 내에서 출력형태	비고
<code>%c</code>	<code>c</code>	"A"	필드 폭 1 (디폴트)
<code>%2c</code>	<code>c</code>	" A"	필드 폭 2, 우측 정렬
<code>%-3c</code>	<code>c</code>	"A "	필드 폭 3, 좌측 정렬
<code>%s</code>	<code>s</code>	"Blue moon! "	필드 폭 10 (디폴트)
<code>%3s</code>	<code>s</code>	"Blue moon! "	공간이 더 필요함
<code>%.6s</code>	<code>s</code>	"Blue m"	정밀도 6
<code>%-11.8s</code>	<code>s</code>	"Blue moo "	정밀도 8, 좌측 정렬

printf()

선언과 초기화			
<code>int</code>	<code>i = 123;</code>		
<code>double</code>	<code>x = 0.123456789;</code>		
변환 형식	대응되는 인자	필드 내에서 출력형태	비고
<code>%d</code>	<code>i</code>	"123"	필드 폭 3 (디폴트)
<code>%05d</code>	<code>i</code>	"00123"	영으로 채움
<code>%7o</code>	<code>i</code>	" 173"	우측 정렬, 8진수
<code>%-9x</code>	<code>i</code>	"7b "	좌측 정렬, 16진수
<code>%-#9x</code>	<code>i</code>	"0x7b "	좌측 정렬, 16진수
<code>%10.5f</code>	<code>x</code>	" 0.12346"	필드 폭 10, 정밀도 5
<code>%-12.5e</code>	<code>x</code>	"1.23457-01 "	좌측 정렬, e-형식

scanf()

`scanf(control_string, other_argument)`

– 예

```
char    a, b, c, s[100];
```

```
int     n;
```

```
double  x;
```

```
scanf("%c%c%c%d%s%lf", &a, &b, &c, &n, s, &x);
```

– *control_string*: "%c%c%c%d%s%lf"

– *other_arguments*: &a, &b, &n, s, &x

scanf()

scanf() 변환 문자		
변환 문자	입력 스트림에서 대응되는 문자	대응 인자의 포인터 형
c	공백을 포함한 모든 문자	char
d, i	10진 정수 (부호는 옵션)	integer
u	10진 정수 (부호는 옵션)	unsigned integer
o	8진수 (부호는 옵션)	unsigned integer
x, X	16진수 (부호는 옵션)	unsigned integer
e, E, f, g, G	실수 (부호는 옵션)	floating type
s	공백 없는 문자열	Char
p	printf() 함수의 %p에 의해 출력되는 것으로 일반적으로 부호 없는 16진 정수임	void *
n, %, [...]	다음 표 참조	

scanf()

scanf() 변환 문자	
변환 문자	설명
n	입력 스트림의 문자와 짝을 이루지 않는다. 대응되는 인자는 정수형 포인터로서, 지금까지 읽어들이는 문자의 개수를 저장한다.
%	입력 스트림에서 하나의 %와 짝을 이룬다. 대응되는 인자는 없다.
[...]	각괄호 [] 안에 있는 문자들을 스캔 집합이라 한다. 이것은 무엇이 짝을 이루는가를 결정한다. (아래 설명을 참조하여라.) 대응되는 인자는 문자 배열의 기본 주소에 대한 포인터이고, 이 배열은 끝에 자동적으로 추가되는 널 문자를 포함하여 대응되는 모든 문자들을 포함할 만큼 큰 크기를 가져야 한다.

scanf()

- 제어 문자열은 다음과 같은 것을 포함할 수 있음
 - 여백
 - %이외의 공백문자가 아닌 일반 문자
 - %로 시작해서 변환 문자로 끝나는 변환 명세
 - h
 - l
 - L

scanf() 예제

```
int    i;  
char   c;  
char   string[15];  
scanf("%d , %*s %% %c %5s %s", &i, &c, string,  
      &string[5]);
```

- * 입력 스트림 : 45 , ignore_this % C read_in_this**
- i : 45
 - c : C
 - string[0-5] : "read_"
 - string[5-14] : "in_this**"
 - scanf()는 4를 리턴

fprintf()/fscanf()

- 각각 printf()와 scanf() 함수의 파일 버전
- 함수 원형

```
int fprintf(FILE *fp, const char *format, ...)
```

```
int fscanf(FILE *fp, const char *format, ...)
```

- fprintf(stdout,...);와 printf(...);는 같은 의미
- fscanf(stdin, ...);은 scanf(...);와 같은 의미

sprintf()/sscanf()

- 각각 printf()와 scanf() 함수의 문자열 버전
- 함수 원형

```
int sprintf(char *, const char *, ...);
```

```
int sscanf(const char *, const char *, ...);
```

fopen()

- fopen(filename, mode) 형태의 함수 호출은 filename 파일을 mode에 지정된 모드로 열고, 파일 포인터를 리턴함

fopen()

- 모드

모드	의미
"r"	읽기 위해 문서 파일 열기
"w"	쓰기 위해 문서 파일 열기
"a"	첨부하기 위해 문서 파일 열기
"rb"	읽기 위해 이진 파일 열기
"wb"	쓰기 위해 이진 파일 열기
"ab"	첨부하기 위해 이진 파일 열기

- 모드 뒤의 +는 파일을 읽기와 쓰기로 모두 연다는 것을 의미함

fopen()/fclose()

- 파일 열기와 닫기의 전형적인 예제 코드

```
#include <stdio.h>
int main(void) {
    int    a, sum = 0;
    FILE   *ifp, *ofp;
    ifp = fopen("my_file", "r");
    ofp = fopen("outfile", "w");
    . . . . .
    fclose(ifp);
    fclose(ofp);
}
```

파일의 임의의 위치 접근

- `ftell(file_ptr)`
 - 파일 위치 지시자의 현재 값을 리턴
- `fseek(file_ptr, offset, place)`
 - 파일 위치 지시자를 `place`부터 `offset` 바이트 떨어진 곳을 나타내는 값으로 설정함
 - `place`의 값은 0(`SEEK_SET`), 1(`SEEK_CUR`), 2(`SEEK_END`) 중 하나가 될 수 있는데, 이것들은 각각파일의 처음, 현재 위치, 파일의 끝을 나타냄

파일의 임의의 위치 접근

- 파일을 역으로 출력하는 프로그램

```
#include <stdio.h>
int main(void){
    char fname[100]; int c; FILE *i fp;
    fprintf(stderr, "\nInput a filename: ");
    scanf("%s", fname);
    i fp = fopen(fname, "rb");
    fseek(i fp, 0, SEEK_END);
    fseek(i fp, -1, SEEK_CUR);
    while (ftell(i fp) > 0) {
        c = getc(i fp);
        putchar(c);
        fseek(i fp, -2, SEEK_CUR) ; }
    return 0;
}
```