Kumoh National Institute of Technology

C Programming

Ch.2-3 printf 함수와 scanf 함수

컴퓨터공학과

C프로그래밍

Contents

printf 함수와 특수 문자

- printf 함수는 기본적으로 문자열을 출력하는 함수!
  - \n(다음 줄) 등 특수 문자 사용 가능

```
int main(void)
{
    printf("I like programming \n");
    printf("I love puppy! \n");
    printf("I am so happy \n");
}
```

```
I like programming
I love puppy!
I am so happy
```

- 잘못된 printf 함수 호출문

printf("앞집 강아지가 말했다. "멍~! 멍~!" 정말 귀엽다.");

수정

큰 따옴표는 문자열의 시작과 끝으로 해석이 되니, 큰 따옴표 자체의 출력을 원하는 경우에는 큰 따옴표 앞에 \ 문자를 붙여주기로 하자!

printf("앞집 강아지가 말했다. \"멍~! 멍~!\" 정말 귀엽다.");

특수문자의 탄생 배경

특수 문자의 종류

특수문자	의미하는 바
\a	경고음
\b	백스페이스(backspace)
\f	폼 피드(form feed)
\n	개 행(new line)
\r	캐리지 리턴(carriage return)
\t	수평 탭
\v	수직 탭
\'	작은 따옴표 출력
\"	큰 따옴표 출력
\?	물음표 출력
\\	역슬래쉬 출력

\f와 \v는 모니터 출력이 아닌 프린터 출력을 위해 정의된 특수문자이기 때문에 모니터의 출력에 사용하면, 이상한 문자 출력!

printf 함수의 서식 지정

- printf 함수의 서식 지정 : 출력의 형태를 지정
  - 예) 출력할 문자열 내에 정수 삽입 (%d)
  - printf의 f는 “formatted”, 즉 “서식이 지정된”을 의미
  - 서식 지정의 예

```
int main(void)
{
    int age = 12;
    printf("10진수로 %d살이고 16진수로 %X살 입니다. \n", age, age);
}
```

```
10진수로 12살이고 16진수로 C살 입니다.
```

printf 함수의 서식 문자들

서식문자	출력 대상(자료형)	출력 형태
%d	char, short, int	부호 있는 10진수 정수
%ld	long	부호 있는 10진수 정수
%lld	long long	부호 있는 10진수 정수
%u	unsigned int	부호 없는 10진수 정수
%o	unsigned int	부호 없는 8진수 정수
%x, %X	unsigned int	부호 없는 16진수 정수
%f	float, double	10진수 방식의 부동소수점 실수
%Lf	long double	10진수 방식의 부동소수점 실수
%e, %E	float, double	e 또는 E 방식의 부동소수점 실수
%g, %G	float, double	값에 따라 %f와 %e 사이에서 선택
%c	char, short, int	값에 대응하는 문자
%s	char *	문자열
%p	void *	포인터의 주소 값

char, short 값은 int 로 변환되어 전달

float 값은 double 로 변환되어 전달

정수 : 8진수와 16진수 출력

- 8진수 %o, 16진수 %x
  - #을 삽입하면 출력값 앞에 8진수의 경우 0, 16진수의 경우 0x가 삽입됨

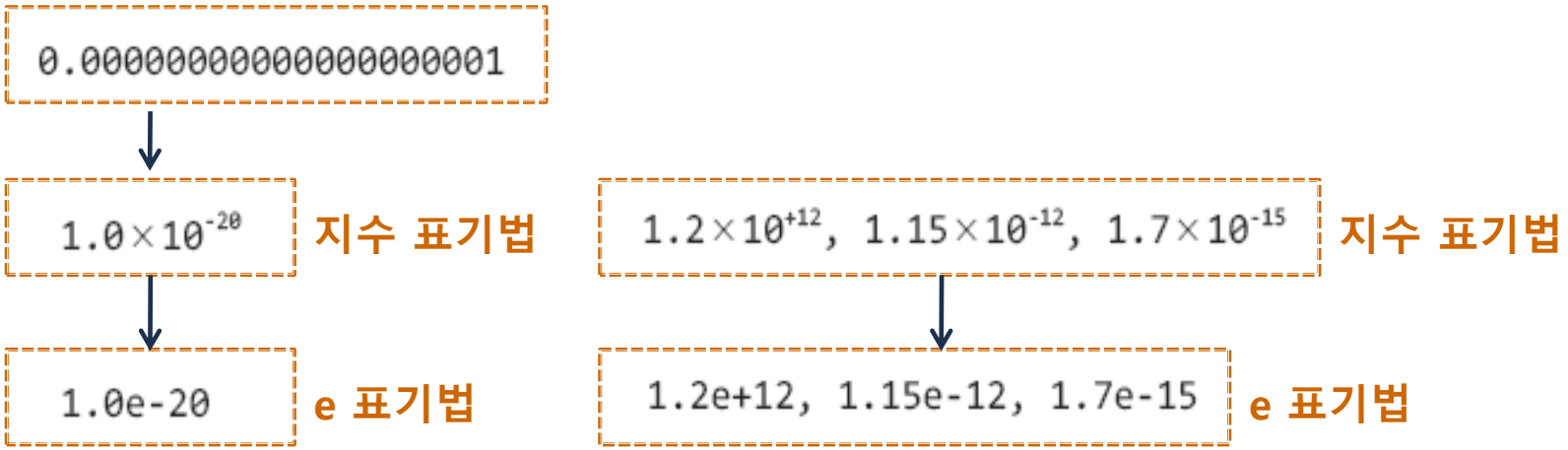
```
int main(void)
{
    int num1 = 7, num2 = 13;

    printf("%o %#o \n", num1, num1);
    printf("%x %#x \n", num2, num2);
}
```

```
7 07
d 0xd
```

실수 : %f, %e (1)

- %f : 일반 10진수 표기, %e : 지수 형식의 과학적 표기




```
int main(void)
{
    printf("%f \n", 0.1234);
    printf("%e \n", 0.1234);
    printf("%f \n", 0.12345678);
    printf("%e \n", 0.12345678);
}
```

```
0.123400
1.234000e-001
0.123457
1.234568e-001
```

실수 : %f, %e (2)

- %.3f : 소수점 이하 셋째 자리까지 표기
  - 이하는 반올림
  - 디폴트값 : 6 (소수점 이하 여섯 자리까지 표기)
- %.3e : 과학적 표기법도 동일

```
int main(void)
{
    printf("%f\n", 123.1234567);
    printf("%.4f\n", 123.1234567);
    printf("%.10f\n", 123.1234567);
    printf("%.0f\n", 123.1234567);

    printf("%e\n", 123.1234567);
    printf("%.4e\n", 123.1234567);
    printf("%.10ef\n", 123.1234567);
    printf("%.0e\n", 123.1234567);
}
```

```
123.123457
123.1235
123.1234567000
123
1.231235e+002
1.2312e+002
1.2312345670e+002f
1e+002
```

실수 : %g

- %g : %f와 %e 사이에서 적절한 형태(?)의 출력 선택
  - %e로 표기되는 경우 : 지수값이 -4보다 작을 때(짧은 표현 선택) 또는 지수값이 유효 자리 수 이상일 때
    - 디폴트 유효 자리 수 : 6 → %.4g (유효 자리 수는 4)

```
int main(void)
{
    double d1 = 0.00012345678;  // 1.2345678e-4
    double d2 = 0.000012345678; // 1.2345678e-5
    printf("%g \n", d1);        // 0.000123457 출력
    printf("%g \n", d2);        // 1.23457E-005 출력

    double d3 = 1230.12;        // 1.23012e+3;
    double d4 = 12300.12;       // 1.230012e+4;
    double d5 = 123000.12;      // 1.2300012e+5;
    printf("%.4g \n", d3);      // 1230
    printf("%.4g \n", d4);      // 1.23e+004 출력
    printf("%.4g \n", d5);      // 1.23e+005 출력
}
```

소수점 이하 마지막에 나타나는 0(들)은 표기하지 않음

```
0.000123457
1.23457e-005
1230
1.23e+004
1.23e+005
```

문자열 : %s

- 문자열 상수의 예 : “hello”, “c programming”
  - 쌍따옴표로 표현
  - 문자열에 대한 보다 자세한 내용은 10장(10주차)에서 학습함

```
int main(void)
{
    printf("%s, %s, %s \n", "AAA", "BBB", "Hello C World!");
}
```

```
AAA, BBB, Hello C World!
```

필드 폭 지정 : 정돈된 출력 보이기

- 필드 폭 지정 방법
  - %와 서식 문자 사이에 필드의 폭 기술
  - 기본적으로 오른쪽 정렬

%8d
필드 폭을 8칸 확보하고, 오른쪽 정렬해서 출력을 진행한다.

%-8d
필드 폭을 8칸 확보하고, 왼쪽 정렬해서 출력을 진행한다.

```
이  름          전공학과  학년
김동수          전자공학     3
이을수        컴퓨터공학     2
한선영        미술교육학     4
```

```
int main(void)
{
    printf("%-8s %14s %5s \n", "이  름", "전공학과", "학년");
    printf("%-8s %14s %5d \n", "김동수", "전자공학", 3);
    printf("%-8s %14s %5d \n", "이을수", "컴퓨터공학", 2);
    printf("%-8s %14s %5d \n", "한선영", "미술교육학", 4);
}
```

필드 폭 지정 예

```
int main(void)
{
    printf("%8s %8s %8s %8s \n", "이름", "국어", "수학", "평균");
    printf("%8s %8d %8d %8.1f \n", "홍길동", 80, 90, 85.0);
    printf("%8s %8d %8d %8.1f \n", "김유신", 90, 95, 92.5);
    printf("%8s %8d %8d %8.1f \n", "이순신", 100, 97, 98.5);
}
```

%8.1f : 실수 출력
- 필드 폭 : 8, 소수점 이하 출력 자리 수 : 1

```
    이름       국어       수학       평균
  홍길동         80         90       85.0
  김유신         90         95       92.5
  이순신        100         97       98.5
```

scanf 함수 기초

- scanf 함수에게 전달해야 할 2가지 정보
  - 입력의 형식 : 어떻게 받아들일 거니?
    - 기본적으로 사용자가 입력하는 정보는 문자(들)임

      → 어떻게 해석할 것인가?
  - 입력의 장소 : 어디에 저장할까?

```
int num;
scanf("%d", &num);
```

서식 지정
→ 해석 방식

입력 장소
→ num 변수가 있는 메모리

정수 기반의 입력 서식 지정

- %d : 10진수
- %o : 8진수
- %x : 16진수

printf 함수의 출력 서식 지정과 동일

```
int main(void)
{
    int num1, num2, num3;
    printf("세 개의 정수 입력: ");
    scanf("%d %o %x", &num1, &num2, &num3);

    printf("입력된 정수 10진수 출력: ");
    printf("%d %d %d \n", num1, num2, num3);
}
```

```
세 개의 정수 입력: 12 12 12
입력된 정수 10진수 출력: 12 10 18
```

실수 기반의 입력 서식 지정 (1)

- 주의 : 입력 서식과 출력 서식이 다름
  - printf : float – %f, double – %f, long double – %Lf
    - printf 함수 호출 시 float, double 값 모두 double로 전달됨
      - %f와 %lf의 의미 동일 → float, double 모두 %f로 사용하면 됨
  - scanf : float – %f, double – %lf, long double – %LF
    - float, double의 저장 장소 크기 및 저장 방식이 다르므로 반드시 구별해야 함
    - 데이터 입력 시 e 표기법도 사용 가능
    - scanf에서는 %f, %e, %g 모두 의미가 동일

실수 기반의 입력 서식 지정 (2)

```
int main(void)
{
    float num1;
    double num2;
    long double num3;

    printf("실수 입력1: ");
    scanf("%f", &num1);
    printf("입력된 실수 %f \n", num1);

    printf("실수 입력2(e 표기법으로): ");
    scanf("%lf", &num2);
    printf("입력된 실수 %f \n", num2);

    printf("실수 입력3(e 표기법으로): ");
    scanf("%Lf", &num3);
    printf("입력된 실수 %Lf \n", num3);
}
```

```
실수 입력1: 0.0011
입력된 실수 0.001100
실수 입력2(e 표기법으로): 1.1e-3
입력된 실수 0.001100
실수 입력3(e 표기법으로): 0.1e+2
입력된 실수 10.000000
```

(참고) 서식 문자 사이의 공백의 의미

- scanf("%d %d") : 데이터 사이에 공백이 하나 나옴
  - scanf("%d#%d") : 데이터 사이에 # 문자가 하나 나옴

```
int main(void)
{
    int num1, num2, num3;

    printf("정수 3개 입력 : ");
    scanf("%d, %d, %d", &num1, &num2, &num3);
    printf("%d, %d, %d \n", num1, num2, num3);
}
```

```
정수 3개 입력 : 100, 200, 300
100, 200, 300
```

입력 시 반드시 , 문자 삽입

scanf("%d%d%d")와같이 공백을 넣지 않아도 됨
- 데이터 입력 시에는 공백으로 구분. 기본적으로 공백은 데이터로 취급하지 않음

이번 장에서 배운 것

- printf 함수에서 ₩n, ₩t, ₩" 등의 특수 문자를 사용할 수 있다.
- 정수 출력을 위해 %d, %o, %x 서식 문자를 사용할 수 있다.
- 실수 출력을 위해 %f, %e, %g를 사용할 수 있다.
- 출력 시 필드 폭을 잡기 위해 %10d와 같이 사용하며 왼쪽 정렬을 위해서는 %-10d와 같이 사용한다.
- scanf 함수에서도 정수 입력을 위해 %d, %o, %x를 사용할 수 있다.
- 실수 입력 시 float과 double을 위해 각각 %f, %lf를 사용한다.
- printf와 scanf 함수에는 더 많은 기능들이 포함되어 있다. 필요할 때 MSDN 도움말 등을 참고하여 프로그램을 작성하면 된다.