Formatted Input / Output

소프트웨어학부
박영훈 교수

이 단원의 목표

- **`printf`** 함수와 **`scanf`** 함수의 원리를 알아본다.
- **`printf`** 함수에서 출력되는 변수를 정렬하는 법, 자릿수를 정하는 법을 알아본다.
- **`scanf`** 함수에서 입력 포맷이 정해진 상태에서 변수를 입력하는 방법을 알아본다.

printf 함수의 고급 용법

- **printf** 함수는 다음과 같은 형식으로 사용된다.

```
printf("문자열", expression1, expression2, …);
```

(여기서 expression은 상수와 변수, 수식을 모두 포함한다)

- 문자열은 문자들(한글 포함)과 %로 시작되는 변수 표시자들로 구성되어 있다.
- 이 때, expression의 개수와 변수 표시자의 개수는 서로 같아야 한다.
- expression의 개수와 변수 표시자의 개수가 서로 다르다고 해서 컴파일 에러가 뜨지는 않는다. 하지만, 원하는 결과가 나오지 않을 뿐 아니라 warning메시지가 뜨게 된다.

  잘못된 예:

```
printf("%d, %d", i);  또는  printf("%d", i, j);
```

- 또한, 변수 표시자의 타입과 expression의 타입이 서로 다르다고 해서도 컴파일 에러가 뜨지는 않는다. 하지만 역시 원하는 결과가 나오지 않을 수 있고, warning이 뜬다.

  잘못된 예:

```
int i = 10;
float x = 3.4f;
printf("%d, %f", x, i);
```

변수 표시자 (Conversion Specification)

- `int i = 123; float x = 4.327f;` 로 선언되어 있다고 하자.
- 변수 표시자의 일반적인 형태는 `%`m`.`p`T` 또는 `%-`m`.`p`T` 이다.

- m
  - 출력되는 변수가 차지하는 칸 수를 말 한다. 이 때, `-`가 없으면 오른쪽 정렬, `-`가 있으면 왼쪽 정렬이다.
  - 예

    `printf("|%10d|%-10d|%10f|%-10f|", i, i, x, x);` 의 결과는 다음과 같다.

    `|•••••••123|123•••••••|••4.327000|4.327000••|` (단, •는 빈 칸을 나타냄)

- p
  - 실수형을 출력할 경우, 소수점 이하 표시될 자릿수의 개수를 지정한다. 만일 소수점 이하 자릿수의 개수가 p보다 많으면 p의 자리까지 반올림하여 출력한다.
  - 정수형을 출력할 경우, 표시될 자릿수의 개수를 지정한다. 이 때, 정수의 자릿수 (마이너스 기호 포함)가 p보다 크거나 같을 경우 그 정수를 그대로 출력하지만, 반대의 경우는 정수 앞을 `0`으로 채운다.
  - 예

    `printf("|%.5d|%.2d|%.5f|%.2f|", i, i, x, x);` 의 결과는 다음과 같다.

    `|00123|123|4.32700|4.33|`

변수 표시자 (Conversion Specification)

- $m.p$ 또는 $-m.p$
  - m과 p를 둘 다 쓰게 되면 두 기능을 동시에 사용하게 된다. 즉, 총 **m** 칸을 차지하며, 정수의 경우 **p**자리를, 소수의 경우 소수점 **p**째자리까지 반올림하여 표시하되, **-**가 있으면 왼쪽 정렬, **-**가 없으면 오른쪽 정렬이 된다.

    printf("|%2.2d|%8.2d|%8.5d|%-8.5d|", i, i, i, i); 의 결과는 다음과 같다.

    |123|•••••123|•••00123|00123•••|

    printf("|%2.2f|%8.2f|%8.5f|%-8.5f|", x, x, x, x); 의 결과는 다음과 같다.

    |4.33|••••4.33|•4.32700|4.32700•|

  - 만일 **m** 또는 **p**를 비워놓으면 **0**을 썼을 때와 같은 효과가 난다.

    printf("|%2.d|%8.d|%8.f|%-8.f|", i, i, x, x); 의 결과는 다음과 같다.

    |123|•••••123|•••••••4|4•••••••|

- **T**
  - 정수형을 출력할 때는 **d**, 실수형을 출력할 때는 **f**, **e** (**E**), **g**를 쓴다.
  - **e** (**E**): 지수 형태로 표시한다. **printf("%.3e", 40.2345);** 의 결과는 **4.023e+01**이다.
  - **g**: **%f**나 **%e**로 표현할 수 있는 방법 중 더 간단한 방법으로 표시된다.

Escape Sequence

- 특수문자나 C에서 이미 다른 기능이 있는 문자를 출력하기 위하여 사용되는 \으로 시작하는 2개짜리 문자열.

- 많이 쓰이는 Escape Sequence

Symbol	의미	Symbol	의미
`\a`	Beep	`\\`	역슬래시 출력
`\b`	Backspace	`\"`	큰 따옴표 출력
`\n`	한 줄 바꿈	`\'`	작은 따옴표 출력
`\t`	탭 (Tab)	`%%`	%를 출력

따옴표안에 넣을때 단독으로 쓰면 문자열이 닫히게되므로.

(8칸의 8의배수번째칸으로)

- `printf("\"Hello\tWorld\"");` 를 실행하면 다음과 같은 결과가 나온다.

```
"Hello••World"
```

\0 : 끝내는것

↳ 여기까지라는 알림!

scanf 함수

- **scanf** 함수 역시 **printf** 함수와 마찬가지로 입력받을 포맷을 정할 수 있다.
- **scanf** 함수 안의 따옴표 안에 conversion specification 외에 일반 문자도 넣을 수 있다.

- conversion specification만 있는 경우

```
int i, j;
float x, y;
scanf("%d%d%f%f", &i, &j, &x, &y);
```

다음과 같이 입력했다고 하자.

```
1-20.3-4.0e3
```

그러면 **i**, **j**, **x**, **y**에 각각 **1**, **-20**, **0.3**, **-4000.0**이 저장된다.

scanf 함수의 동작 원리

- **scanf** 함수 안의 따옴표 안에 conversion specification만 여러 개 있는 경우, 키보드로 입력한 것들 중 white space로 입력한 문자들을 구분한다.

white space란?: 스페이스, 탭, 줄바꿈 등과 같이 눈에 보이진 않지만 문자로 취급 되는 것들

1 -20 .3 -4.0e3

- **scanf("%d%d%f%f", &i, &j, &x, &y);** 에서 다음과 같이 입력해도 같은 값이 입력된다.

```
•••••1
-20•••••••.3
••••••••••4.0e3
```

왜냐하면 **scanf** 함수에 들어가는 문자열은 다음과 같은데(단, •는 스페이스, ¤는 줄바꿈), 너는 이렇게받아들임

```
•••••1¤-20•••••••.3¤••••••••••4.0e3¤
```

↑줄바꿈

이 입력에서 스페이스와 엔터는 각 입력한 값들을 구분해주는 역할을 해 주기 때문이다.

* float연산은 오차가 발생할수있음.

scanf 함수

- **scanf** 함수 안의 따옴표 안에 conversion specification 와 일반 문자가 둘 다 있는 경우
  일반 문자를 모두 정확한 위치에, 정확히 써 줘야 한다.

- 예

```
int m, s;
scanf("%dmin%dsec", &m, &s);
```

  일 때, 키보드로는 **30min25sec** 와 같이 입력해야 한다.

  단, 변수 부분 앞에는 white space가 와도 무방하다. 즉, **••30min•••25sec** 와 같이 입력해도 된다.

- 만일 정확하게 입력하지 않은 경우, 앞에서부터 정상적으로 입력한 곳 까지만 받아들이고, 그 다음부터는 무시해 버린다. 예를 들어, **30mi25sec** 와 같이 입력하면 변수 **m**에 **30**이 들어가긴 하지만 변수 **s**에는 아무 값도 들어가지 않게 된다.

실습

- 분자, 분모가 모두 양의 정수로 이루어진 두 개의 분수를 입력받아, 그것을 더하는 프로그램을 만들어보자.

- 조건
  - 입력 포맷: `%d/%d+%d/%d`
  - 출력 포맷: `%d/%d`

- 실행 예

  분수의 합을 입력하시오: `5/6+3/4`
  분수의 합 `= 38/24`

```
int a,b,c,d;
int m,n;
printf("Enter two fractions: ");
scanf("%d/%d+%d/%d", &a, &b, &c, &d);
n=b*d;
m=a*d+b*c;
printf("Sum=%d/%d\n");
return 0;
```