

2010-1학기 프로그래밍입문(1)

1장. C의 개요

박 종 혁

Tel: 970-6702

Email: jhpark1@snut.ac.kr

C의 개요

- C-Language란?

- 원하는 결과를 얻어내기 위한 Program 작성시 필요한 일종의 언어
- Unix 운영 체제하에서 시스템 프로그래밍을 하기위해 개발된 언어
- 구조적인 언어, 강력한 기능, 빠른 속도

- C 언어의 역사

- ALGOL60 (ALGOritmic Language): 1960년 국제 위원회에서 발표
- CPL (Combined Programing Language) :1963년 영국 캠브리지 런던 대학
- BCLP (Basic CLP) : 영국 캠브리지 런던대학
- B : 미국 AT&T Bell 연구소의 Ken Thompson, 1970년 발표
- C : 미국 AT&T Bell 연구소의 Dennis Ritchie, 1972
- C++ : 표준 C에서 OOP(Object Oriented Programing, 객체지향 프로그램)의 개념 추가

- C 언어의 종류

- ANSI C (American National Standard Institute) : 표준화 작업, "Standard C"
- Visual C : Microsoft사에서 개발
- Borland C : Microsoft사에서 개발

- 프로그램이 만들어지는 절차

- Source Code --> Source File (*.c) --> 선행처리 지시어가 번역
- Source File --> Object File (*.obj) --> Executable File (*.exe)

프로그래밍 예

- “from sea to shining C”를 출력하는 프로그램 작성
1. 문자 편집기를 사용하여 다음과 같은 내용을 가지는 파일을 작성하고 파일 확장자가 .c인 파일이름을 줌 (예, sea.c)

```
#include <stdio.h>
int main(void){
    printf("from sea to shining C\n");
    return 0;
}
```

(참고) 파일 이름은 프로그램 성격에 맞는 것으로 선택해야 함

sea 프로그램 분석(1)

- `/* In file sea.c */`
 - `/* */`은 주석을 나타내며 컴파일러는 이것을 무시함
- `#include <stdio.h>`
 - `#`으로 시작하는 행은 전처리기에 의해 처리됨
 - 프로그램을 컴파일하면 컴파일 전에 전처리기가 먼저 동작됨.
- `int main(void)`
 - 식별자 다음에 괄호()가 오면 그 식별자는 함수라는 것을 나타냄
 - 모든 C 프로그램은 반드시 `main()` 함수를 포함함
 - 프로그램의 수행은 항상 `main()` 함수로부터 시작함

sea 프로그램 분석(2)

- { }
 - 중괄호는 여러 문장들을 그룹화하기 위해 사용됨
 - 즉, 중괄호를 둘러싸인 것은 하나의 단위로 취급됨
- "from sea to shining C\n"
 - 큰따옴표로 둘러싸인 일련의 문자들을 문자열 상수라고 함
 - 문자열 상수를 이루는 단어들은 그 본래의 의미를 잃어버림
 - \n은 개행 문자를 나타냄
 - 프로그램 상에서 일반 문자로 표현할 수 없는 것을 표현하고 싶을 때 역슬래시 \와 결합된 문자를 사용함

sea 프로그램 분석(3)

- `printf("from sea to shining C\n");`
 - `printf` 뒤에 괄호가 있기 때문에 `printf()`는 함수임.
 - 제일 뒤에 세미콜론 `;`이 있기 때문에 이것은 문장임.
 - 즉, C에서 모든 문장은 세미콜론으로 끝남.

printf()

- 화면에 출력하는 함수
- 연속적으로 printf()가 있을 경우, 뒤에 나오는 printf()의 출력은 바로 앞 printf()의 마지막 출력 위치에서부터 시작하여 출력한다.
- 즉, 다음 printf()는
`printf("from sea to shining C\n");`
다음과 같이 사용해도 같은 출력을 낸다.
`printf("from sea to ");`
`printf("shining C");`
`printf("\n");`

printf() 예제

```
#include <stdio.h>
int main(void){
    printf("\n\n\n\n\n\n\n\n\n\n");
    printf("          *****\n");
    printf("          *   from sea          *\n");
    printf("          *   to shining C      *\n");
    printf("          *****\n");
    printf("\n\n\n\n\n\n\n\n\n\n");
    return 0;
}
```

1.3 변수, 수식, 배정

```
/* The distance of a marathon in kilometers. */
#include <stdio.h>
int main(void)
{
    int miles, yards;
    float kilometers;
    miles=26;
    yards=385;
    kilometers=1.609*(miles + yards / 1760.0 );
    printf("\n A marathon is %f kilometers. \n\n", kilometers);
    return 0;
}
```

==> A marathon is 42.185970 kilometers.

▶ `/* The distance of a marathon in kilometers. */`

□ 주석문은 `/*` 와 `*/` 로 둘러싼다

▶ `int miles, yards;`

선언문, 변수 `miles, yards`가 정수값을 가지는 변수
//선언문과 문장은 세미콜론으로 끝남.

▶ `float kilometers;`

선언문, 변수 `kilometers`가 실수값을 가지는 변수

□ 변수는 선언하고 나서 사용한다

▶ `miles=26;`

`yards=385;`

배정문, 정수형 상수 26과 385가 변수 `miles`과 `yards`에 각각 배정된다.

▶ `kilometers=1.609*(miles + yards / 1760.0 );`

배정문 `/, *, +` : 연산자 (`-`, `%`, `...`)

▶ `printf("Wn A marathon is %f kilometers. WnWn", kilometers);`

변환형식 `%f` 과 인자 `kilometers`는 짝을 이루며, `kilometers`의 값이 부동 소수점(float) 형식 `%f`의 위치에 출력됨.

□ 변수의 값을 출력하려면 서식지정이 필요하다.

▶ 수식의 변환 규칙 (Conversion rule)

`7/2 --> 3,    7.0/2 --> 3.5`

1.4 #define 과 #include의 사용

```
#define LIMIT 100  
#define PI 3.14159  
#define C 299792.458
```

```
/* speed of light in km/sec */
```

▶ # : 전처리기 지시자 (Preprocessing Directive)

이문장이 포함된 파일을 컴파일 하면, 전처리기는 문자열 상수나 주석을 제외한 곳에서 대체함.

```
#include "my_file.h"
```

▶ file에 my_file.h 파일의 사본 포함

▶ C에서 제공하는 표준 헤더파일 : stdio.h, string.h, math.h, <xxx.h>

15페이지 예제)

```
/*In file pacific_sea.h */
```

1.5 printf() 와 scanf()의 사용

- printf() : 화면 출력
 - printf(“서식지정문자열“, ”변수“);
- 서식지정 문자열
 - 일반문자열, 변환문자열(%), 확장문자열(W)
- printf()의 변환문자열
 - printf(“%변환문자“, ”변수“); ”변수“를 변환형식에 맞추어 화면 출력

- 변환문자 (Conversion Character) - printf()

변환문자	대응되는 인자가 출력되는 형식
c	as a character (문자)
d	as a decimal integer (10진 정수)
ld	as a long type decimal integer (long형 10진 정수)
e	as a floating point number in scientific notation
f	(지수형..)
g	as a floating point number (float, double)
s	in the e-format or f-format, whichever is shorter as a string (문자열)

- 화면에 abc 출력하는 방법

```
printf("abc");
```

```
printf("%s", "abc");
```

```
printf("%c%c%c", 'a','b','c');
```

- `printf()`의 옵션 지정

%필드 폭자리수변환문자

%d -> 123

%5d -> _ _ 123

%10d -> _ _ _ _ _ 123

%2d -> 123 (지정폭이 작아도 필요한 폭은 확보)

%f -> 654.321000 (표준폭으로 출력)

%12f -> _ _ 654.321000

(소수점 넣어 12자리로 출력, 이하는 표준폭으로 출력)

%9.2f -> _ _ _ 654.32

(소수점 넣어 9자리로 출력, 이하는 2자리로 출력)

- scanf() : 키보드 입력
- scanf()의 변환문자열
 - printf(“%변환문자“, &변수); 변환문자 형식으로 입력 받아 들임.
- 변환문자 (Conversion Character) - scanf()

변환문자	입력스트림의 문자가 변환되는 형식
c	to a character (문자)
d	to a decimal integer (10진 정수)
ld	to a long type decimal integer (long형 10진 정수)
f	to a floating point number (float)
lf	to a floating point number (double)
LF	to a floating point number (long double)
s	to a string (문자열)

실습

```
#include <stdio.h>
int main(void)
{
    char c1,c2,c3;
    int i;
    float x;
    double y;
    printf("\n%s\n%s", "Input three character,"
           "an int, a float, and a double: ");
    scanf("%c%c%c%d%f%lf", &c1, &c2, &c3, &i, &x, &y);
    printf("\nHere is the data that you typed in:\n");
    printf("%3c%3c%3c%5d%17e%17e\n\n", c1,c2,c3,i,x,y);
}
```

입력 : ABC 3 55 77.7

➔ ??

출력

Input three character,
an int, a float, and a double: ABC 3 55 77.7

Here is the data that you typed in:

```
_ _ A_ _ B_ _ C_ _ _ _ 3_ _ _ _ 5.500000e+ 01_ _ _ _ 7.770000e+ 01
```

1.6 제어의 흐름

선택 : if, if-else

반복 : for, while, do-while

if 문

- 일반적인 형태 : if (조건식) { 문장1 }
-> 조건식이 참(true)이면 (0이 아니면) 문장1 실행, 단문이면 {} 생략
- a=1
if (b==3) a=5; /* == : '--와 같다' 연산자 */
printf("%d", a);
-> b가 3이면 a=5
-> b가 3이 아니면 문장(a=5) 실행 안함, printf() 문 실행 1 출력

if-else 문

- 일반적인 형태 : if (조건식) { 문장1 }
 else { 문장2 }

-> 조건식이 참(true)이면 (0이 아니면) 문장1 실행
 그렇지 않으면 문장2 실행

예)

```
if(cnt==0){  
    a=2;  
    b=3;  
    c=5;  
}  
else {  
    a=-1;  
    b=-2;  
    c=-3;  
}  
printf("%d", a+ b+ c);
```

==> cnt 가 0값을 가지면 10 출력, 그렇지 않으면 -6 출력

■ while 루프

- 일반적인 형태 : while (조건식) { 문장 }

```
#include<stdio.h>
int main(void)
{
    int i=1, sum=0;
    while (i<=5) {
        sum+=i;
        ++i;
    }
    printf("sum= %d \n", sum);
    return 0;
}
```

▶ 참고

++i, i++; 증가 --i, i--; 감소
i=i+1; i=i-1;

■ while 루프

- 일반적인 형태 : for (조건식) { 문장 }

```
#include<stdio.h>
int main(void)
{
    int sum=0;
    for (i=1; i<=5; ++i ) {
        sum+=i;
    }
    printf("sum= %d \n", sum);
    return 0;
}
```

질의 및 응답

- 끝 -