

<LAB 6>

Ch3 : 변수및자료형 ~

Ch6 : 조건문 (종합)

제출일 : 2022/10/05

이름 : 고려욱

학번 : 201810528

1. Lab 6-1)

1) 실습 문제

1) 두개의 16진수를 키보드로부터 입력 받고 서로 더한 결과를 다음과 같이 화면에 16진수로 출력하는 프로그램을 작성하라



2) 16진수 형태로 2개의 정수를 키보드로부터 입력 받고 덧셈한 결과를 다음과 같이 10진수와 16진수로 출력하는 프로그램을 작성하라.



2) 배경 지식

[1] 정수의 형식지정자

기본적으로 정수를 출력하기 위해서는 10진수는 %d, 8진수는 %o, 16진수는 %x를 사용한다. 이때 추가적인 옵션으로 출력시의 출력폭을 지정해 줄 수 있다. 형식지정자에서 %와 알파벳 사이에 원하는 길이의 출력폭을 넣어 출력을 진행할 수 있다. 또한 출력폭 앞에 +/-0/#을 사용하여 각각 +기호 출력/ 좌측정렬/ 남아있는 폭을 0으로 채움/ 8진수,16진수의 경우 0또는 0x를 채움 의 옵션을 붙여 출력할 수 있다.

[2] 입출력함수

c언어의 표준 입출력함수는 printf함수와 scanf함수가 존재한다. Printf("<출력내용>",<출력목록>), Scnaf("<형식지정자>",<저장할 변수의 주소>)와 같이 사용한다. Scnaf 사용시에는 저장할 변수의 주소가 사용되어야 하므로 주소가 저장된 변수가 아닌 변수를 출력하기 위해서는 변수명 앞에 '&'를 사용하여 주소임을 나타내 주어야 한다.

3) 소스 코드

[1]

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    int num1, num2; //16진수 저장 변수 선언
    scanf("%x %x", &num1, &num2); //변수에 16진수 2개 입력
    printf("%#x\n", num1 + num2); //두 변수를 더한값을 출력

    return 0;
}
```

[2]

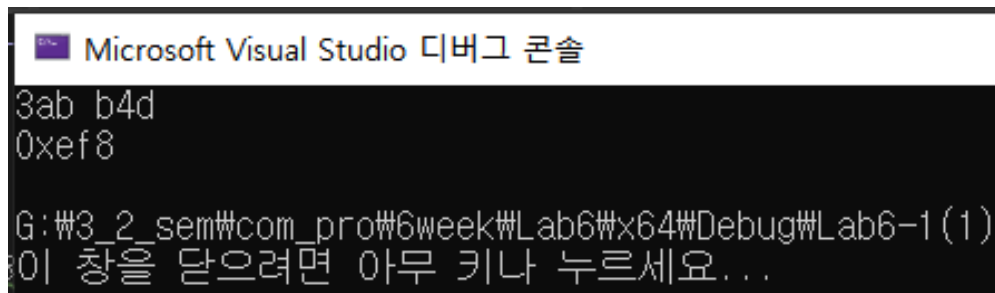
```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    int num1, num2; //16진수 저장 변수 선언
    printf("정수를 입력하시오:");
    scanf("%x", &num1); //num1에 16진수 1개 입력
    printf("정수를 입력하시오:");
    scanf("%x", &num2); //num2에 16진수 1개 입력
    printf("덧셈 결과는 %d(10진수) 입니다.\n", num1 + num2);
    printf("덧셈 결과는 %x(16진수) 입니다.", num1 + num2);

    return 0;
}
```

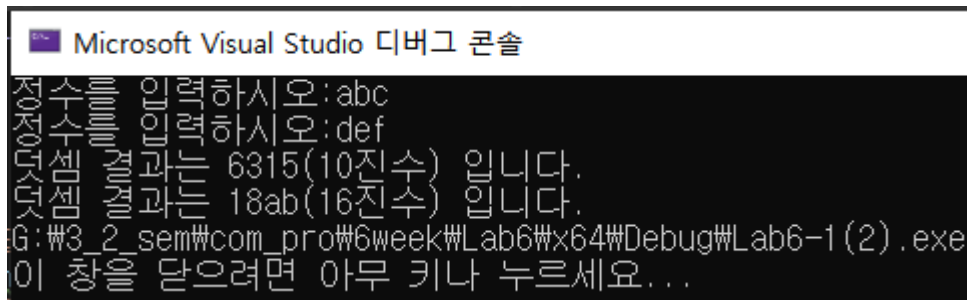
4) 실행결과

[1]



```
Microsoft Visual Studio 디버그 콘솔
3ab b4d
0xef8
G:\#3_2_sem\com_pro\#6week\#Lab6\#x64\Debug\Lab6-1(1)
이 창을 닫으려면 아무 키나 누르세요...
```

[2]



```

Microsoft Visual Studio 디버그 콘솔
정수를 입력하시오:abc
정수를 입력하시오:def
덧셈 결과는 6315(10진수) 입니다.
덧셈 결과는 18ab(16진수) 입니다.
G:\#3_2_sem\com_pro\6week\Lab6\x64\Debug\Lab6-1(2).exe
이 창을 닫으려면 아무 키나 누르세요...

```

5) 결과분석

- 출력시 16진수 앞에 0x를 붙이고 싶다면 형식지정자에 #옵션을 사용하여야 한다.
- 정수형을 입력받는 것이므로 int형을 사용하여 입력을 받되, %x형식지정자를 통해 키보드로 입력된 값이 16진수로 인지되게 하여야 함

2. Lab 6-2)

1) 실습 문제

다음과 같은 출력을 생성하는 프로그램을 작성하시오.



```

C:\Windows\system32\cmd.exe
x(이전) 수식      식의 값 x(이후)
5      x++      5      6
5      x--      5      4
5      ++x      6      6
5      --x      4      4
계속하려면 아무 키나 누르십시오 . . .

```

2) 배경 지식

[1] 전위/후위연산

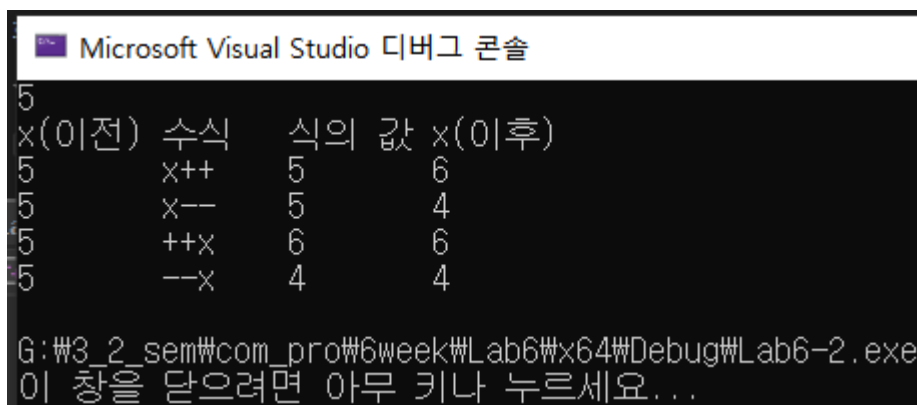
증감연산자에는 전위(prefix)연산과 후위(postfix)연산이 존재한다. (변수)++, --(변수)등으로 사용하며 기본적으로의 기능은 변수값을 각각 1씩 증가시키거나 1씩 감소시킨다. 그러나 둘의 차이는 전위연산의 경우 연산의 결과값이 증가/감소되기 전의 값이고 후위연산의 경우는 연산의 결과값이 증가/감소된 후의 값이라는 것이다. 즉, 연산 이후 변수에 저장되는 값은 동일하지만, 연산식의 연산값이 달라진다.

3) 소스 코드

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    int x, y, z, k; //변수 4개 설정
    scanf("%d", &x); //x의 값 입력
    // 전위, 후위 연산시에 x에 저장된 값 변경되므로
    // 값이 유지되도록 하기 위해서 3개의 변수에 x 저장
    y = x;
    z = x;
    k = x;
    printf("x(이전) 수식   식의 값 x(이후)\n");
    //x++ 후위연산
    printf("%d          ", x);
    printf("x++ %4d %7d\n", x++, x);
    //x-- 후위연산
    printf("%d          ", y);
    printf("x-- %4d %7d\n", y--, y);
    //++x 전위연산
    printf("%d          ", z);
    printf("++x %4d %7d\n", ++z, z);
    //--x 전위연산
    printf("%d          ", k);
    printf("--x %4d %7d\n", --k, k);
    return 0;
}
```

4) 실행결과



```
Microsoft Visual Studio 디버그 콘솔
5
x(이전) 수식   식의 값 x(이후)
5      x++    5      6
5      x--    5      4
5      ++x    6      6
5      --x    4      4

G:\3_2_sem\com_pro\6week\Lab6\x64\Debug\Lab6-2.exe
이 창을 닫으려면 아무 키나 누르세요...
```

5) 검토의견

- 먼저, 반복문을 이용하였다면 코드가 더 간추려져 작성할 수 있었을 것이라고 생각한다.

- 동일한 x변수만을 가지고 이용하고 싶었지만, 전위, 후위연산 후에 x변수에 저장된 값이 달라지므로 출력 이후에 변수를 다시 원래 입력한 값으로 되돌리는 식이 필요했다. 각 경우마다 식을 추가하기 보다 미리 입력값을 3개의 다른 변수에 대입하고 각 변수마다 다른 전위, 후위연산을 사용하고자 하였다.
- 한글로 출력한 출력부와 변수를 출력하는 부분의 간격을 동일하게 하기 위해 여러 번 디버깅을 거쳤다. 한글, 영어, 문자, 숫자의 간격이 서로 동일하지 않다는 것을 느꼈다.

3. Lab 6-3)

1) 실습 문제

연산자 +, -, *, / , %와 두개의 피연산자를 받아 연산결과를 출력하는 간단한 산술계산기를 만들어 본다.

- 모든 값을 정수로 입력 받고 처리

2) 배경 지식

[1] if else 조건문

조건 만족 여부에 따라 if else문이 존재한다. If (cond) stmt1; else if(cond2) stmt2; else if(cond3) stmt3; else stmt4;로 작성한다. 순차적으로 가장 처음에 존재하는 cond 조건부터 조건 만족여부를 점검하며 만족되는 조건이 있는 곳의 statement를 실행하고 현재 조건문에 해당하는 구역 다음 statement를 실행한다.

3) 소스 코드

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    int num1, num2; //피연산자 입력 변수 2개
    char op;        //연산자 입력 변수
    printf("정수 2개를 입력하십시오 :");
```

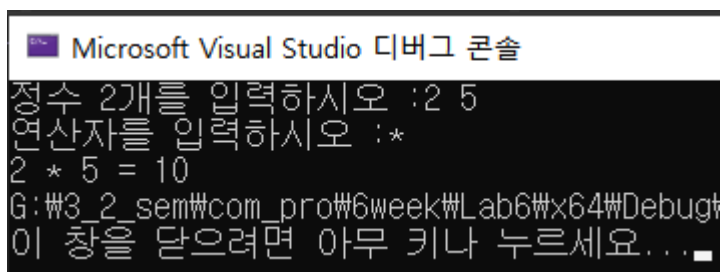
```

scanf("%d %d", &num1, &num2); // 피연산자 2개 입력
printf("연산자를 입력하십시오 :");
scanf(" %c", &op); // 연산자 입력

if (op == '+') { //연산자가 +라면
    printf("%d + %d = %d", num1, num2, num1 + num2);
}
else if (op == '-') { //연산자가 -라면
    printf("%d - %d = %d", num1, num2, num1 - num2);
}
else if (op == '*') { //연산자가 *라면
    printf("%d * %d = %d", num1, num2, num1 * num2);
}
else if (op == '/') { //연산자가 /라면
    printf("%d / %d = %d", num1, num2, num1 / num2);
}
else if (op == '%') { //연산자가 %라면
    printf("%d % %d = %d", num1, num2, num1 % num2);
}
else
    printf("error!");
return 0;
}

```

4) 실행결과

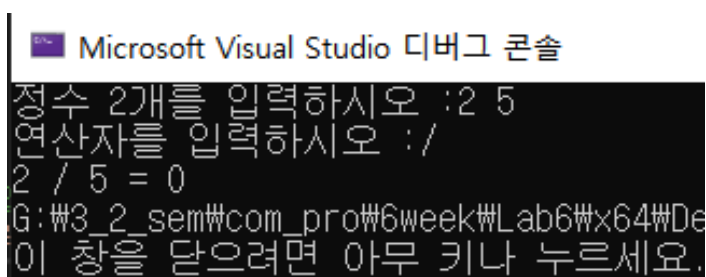


Microsoft Visual Studio 디버그 콘솔

```

정수 2개를 입력하십시오 :2 5
연산자를 입력하십시오 :*
2 * 5 = 10
G:\#3_2_sem\com_pro\#6week\#Lab6\#x64\#Debug
이 창을 닫으려면 아무 키나 누르세요...

```



Microsoft Visual Studio 디버그 콘솔

```

정수 2개를 입력하십시오 :2 5
연산자를 입력하십시오 :/
2 / 5 = 0
G:\#3_2_sem\com_pro\#6week\#Lab6\#x64\#De
이 창을 닫으려면 아무 키나 누르세요.

```

5) 검토의견

- 입력부분에서 scanf(%c)를 사용하면 문제가 있었다. 문자 하나를 받아오기 위해서 scanf(%c)를 사용하였는데 입력이 진행되지 않고 넘어가 error가 출력되었다. 이는 scanf와 개행문자의 오류이다. 키보드의 입력이 입력버퍼에 들어가고 scanf는 그것을

읽어오는데 위의 정수 2개 입력시에 눌렀던 키보드의 엔터가 'wn'의 형태로 입력버퍼에 들어가 있었기 때문이다. 즉, `scanf(%c)`는 바로 'wn'을 읽어 들어와 변수 `op`에 넘겼고 이것이 오류를 발생시킨 것이다.

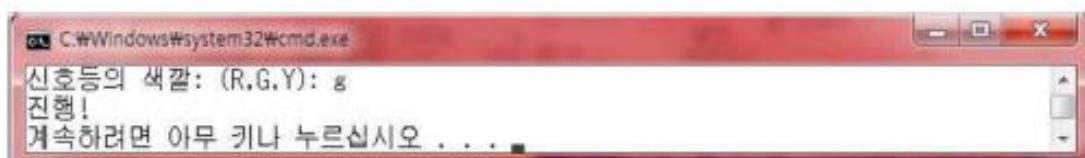
- 따라서, 이 오류를 해결시키기 위해서는 한번에 피연산자와 연산자를 입력 받거나 `%c`앞에 띄어쓰기 하나를 위치시켜 입력을 받을 수 있다. 이번 프로그램에서는 후자의 방식을 사용하였다.
- 마지막으로 올바르지 않은 연산자가 입력되었을 경우 error메시지를 표현한 뒤에 프로그램이 종료되도록 하였다.

4. Lab 6-4)

1) 실습 문제

[1]

사용자가 신호등의 색깔을 입력하면 "정지", "주의", "진행"와 같은 문장을 출력하는 프로그램을 작성하여 보자. If-else 문을 사용한다.



[2]

위의 프로그램을 switch 문을 사용하여 작성하라.

- 주의 : 대문자(G) 대신 소문자(g)를 입력해도 동작하도록 코딩

2) 배경 지식

[1] switch case 조건문

If else 조건문이 여러 번 계속 반복되는 구문을 더 간략하게 표현하기 위해서 사용한다. Switch(exp) { case 상수1 : stmt1; break; case 상수2: stmt2; break; default: stmt; break;}로 작성한다. Exp에 해당하는 값에 들어가는 상수에 해당하는 case문을 찾아 문장이 실행된다. Break는 속해있는 제어문을 탈출하는 문장으로 만약 case문 안에

break가 존재하지 않는다면 만족하는 case문 아래의 모든 문장이 실행되게 된다.

[2] 논리 연산자

논리 연산자는 세가지가 존재한다. '&&', '||' ! 으로 각각 and, or, not을 의미한다. 각 연산자의 결과는 0 또는 1이 출력된다. 0은 거짓을 의미하고 1은 참을 의미한다. && 와 ||는 이항 연산자로 두 피연산자의 논리관계가 지정한 연산자와 부합한지의 여부를 결과로 내보내고 !는 단항연산자로 피연산자가 0이면 결과를 1 0이 아니면 0을 출력한다.

3) 소스 코드

[1]

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    char color; //신호등 색깔 저장 변수 선언
    printf("신호등의 색깔: (R,G,Y): ");
    scanf("%c", &color); //색깔 입력 및 변수 저장

    if (color == 'g' || color == 'G') {
        printf("진행!");
    }
    else if (color == 'r' || color == 'R') {
        printf("정지!");
    }
    else {
        printf("주의!");
    }
    return 0;
}
```

[2]

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main(void)
{
    char color; //신호등 색깔 저장 변수 선언
    printf("신호등의 색깔: (R,G,Y): ");
    scanf("%c", &color); //색깔 입력 및 변수 저장
```

```
switch (color) {  
case 'g': case 'G':  
    printf("진행!");  
    break;  
case 'r': case 'R':  
    printf("정지!");  
    break;  
case 'y': case 'Y':  
    printf("주의!");  
    break;  
default:  
    break;  
}  
return 0;  
}
```

4) 실행결과

[1]

A screenshot of the Microsoft Visual Studio Debug Console. The title bar says "Microsoft Visual Studio 디버그 콘솔". The text in the console is: "신호등의 색깔: (R,G,Y): G", "진행!", "G:\3_2_sem\com_pro\6week\Lab6\64\Debug\Lab6-4(1).exe(", and "이 창을 닫으려면 아무 키나 누르세요...".

[2]

A screenshot of the Microsoft Visual Studio Debug Console. The title bar says "Microsoft Visual Studio 디버그 콘솔". The text in the console is: "신호등의 색깔: (R,G,Y): y", "주의!", "G:\3_2_sem\com_pro\6week\Lab6\64\Debug\Lab6-4(2)", and "이 창을 닫으려면 아무 키나 누르세요...".

5) 검토의견

- if-else의 경우에는 논리연산자를 활용하여 or논리 연산자를 통해 소문자/대문자 모두 입력 시 동작하도록 하였다.
- switch의 경우에는 논리연산자 없이 조건에 해당하는 문자들을 각각 case를 같은 문장에 두고 한번에 처리하였다.