C언어 - HW3

임베디드스쿨2기

Lv1과정

2021. 4. 02

이충재

1. 비트연산자

1) And 연산자(&)

상태가 1인 두 비트를 연산시 1,
한 비트라도 0이면 0

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

2) OR 연산자 (|)

연산비트가 하나라도 1이면 1,
모든 비트가 0이면 0

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

3) NOT연산자(~)

비트반전 0 → 1
1 → 0

A	Y
0	1
1	0

4)XOR 연산자(^)

서로 다른 비트연산시 1
같은 비트 연산시 0

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

5) 쉬프트 연산자

부호 방향으로 숫자크기만큼 비트이동.

오른방향 비트이동: >>
왼 방향 비트이동: <<

ex) num1 = 0100
Num1 >> 2 // 오른쪽으로 두 칸 이동
num1 = 0001

** 부호 변환 과정
1. 뒤쪽에서 가장 먼저 나오는 1을 찾는다.
2. 맨 뒤에서 처음 나오는 1까지는 그대로 두고 나머지는 반전

ex) 20 = 0001 0100
-20 = 1110 1100
---
0 = 1 0000 0000
ㄴ 맨앞비트는 버린다.

* and not : 2^n 단위 정렬 하고자 할때 유용하다.

ex) 1000 & ~(2^6 – 1)

1000 = 1111 0100
& ~(2^6 -1) = 1110 0000   // 2^6자리 아래로는 모두 0이된다

1000 & ~(2^6 -1) = 1110 0000

1100 0000 = 2^6 + 2*2^6 + 4*2^6 = 960
2^6단위로 1000을 정렬하였다.

2^n -1: n번째 비트부터 모두 1이된다.

ex) 2^3 = 1000
2^3 – 1 = 0111
3번째 비트부터 모두 1이되었다.

함수의 기본형태

```
리턴타입 함수이름 (인자타입)
{
        내용
}
```

리턴타입: 어떠한 데이터 타입을 return 할 것인지

인자타입: 어떠한 데이터 타입을 입력으로 할 것인지

```
ex) int my_function (int num)
    {
        return num + 6;
    }
```

입력을 정수로 받고 반환값은 정수로 반환.
num + 6을 반환하는 함

1. 0 ~ 100까지 숫자중 홀수만 출력하시오

```c
#include <stdio.h>

void print_oddnumber(int range)
{
        int i, count = 0;

        if(range < 0)
        {
                printf("잘못된 입력값입니다.");
        }
        for(i = 0; i < range; i++)
        {
                if(i % 2)
                {
                        printf("%3d", i);
                        count++;

                        if((count % 10) == 0)
                        {
                                printf("\n");
                        }
                }
```

```c
int main(void)
{
        int range;

        printf("출력범위를 입력하시오: ");
        scanf("%d", &range);

        print_oddnumber(range);
        printf("\n");

        return 0;
}
```

```
출력범위를 입력하시오: 100
  1  3  5  7  9 11 13 15 17 19
 21 23 25 27 29 31 33 35 37 39
 41 43 45 47 49 51 53 55 57 59
 61 63 65 67 69 71 73 75 77 79
 81 83 85 87 89 91 93 95 97 99
```

2. ~ 50까지 숫자의 합을 구하는 프로그램을 작성하세요.

```c
#include <stdio.h>

void add_function(int range)
{
        int i, sum = 0;
        for(i = 1; i <= range; i++)
        {
                sum += i;
        }

        printf("%d\n", sum);
}
```

```c
int main(void)
{
        int range;

        printf("범위입력하세요: ");
        scanf("%d", &range);

        add_function(range);

        return 0;
}
```

```
범위입력하세요: 50
1275
```

3. 1 ~ 33까지의 숫자중 3의 배수의 합만 구해보세요.

```c
void three(int range)
{
        int i, sum = 0;

        if(range <= 0)
        {
                printf("잘못된 입력값입니다.\n");
        }

        else
        {
                for(i = 1; i <= range; i++)
                {
                        if((i % 3) == 0)
                        {
                                sum += i;
                        }
                }

                printf("%d\n", sum);
        }
}
```

```c
int main(void)
{
        int range;

        printf("범위를 입력하시오: ");
        scanf("%d", &range);

        three(range);

        return 0;
}
```

```
범위를 입력하시오: 33
198
```

4. 1 ~ 100의 숫자중 2의 배수의 합과 3의 배수의 합을 각각 구해봅시다.

```c
void multisum(int range, int multiple)
{
        int i, sum = 0;

        if(range <= 0)
        {
                printf("잘못된 입력값입니다.\n");
        }

        for(i = 1; i <= range; i++)
        {
                if((i % multiple) == 0)
                {
                        sum += i;
                }
        }

        printf("%d\n", sum);
}
```

```c
int main(void)
{

        multisum(100, 2);
        multisum(100, 3);

        return 0;
}
```

```
2550
1683
```

5. 피보나치 수열의 n 번째 항을 구하는 프로그램을 만들어봅시다.

```c
void Fibonacci(int n)
{
        int a = 1, b = 1, result, i;

        if(n <= 0)
        {
                printf("잘못된 입력값입니다.\n");
        }


        if((n == 1) || (n == 2))
        {
                printf("1\n");
        }

        if(n >= 3)
        {
                for(i = 3; i <= n; i++)
                {
                        result = a + b;
                        a = b;
                        b = result;
                }

                printf("n번째항은 %d입니다.\n", result);
        }


}
```

```c
int main(void)
{

        int n;

        printf("알고싶은 항을 입력하세요: ");
        scanf("%d", &n);

        Fibonacci(n);

        return 0;
}
```

```
알고싶은 항을 입력하세요: 5
n번째항은 5입니다.
```

6. 1, 1, 1, 1, 2, 3, 4, 5, 7, 10, 14, 19, 26, 36, 50, ... 으로 진행되는 수열이 있다.

```c
void function(int n)
{
        int a = 1, b = 1, c = 1, d = 1;
        int i, result;

        if(n <= 0)
        {
                printf("잘못된 입력값입니다.");
        }

        if((n > 0) && (n < 5))
        {
                printf("1\n");
        }

        if(n >= 5)
        {
                for(i = 5; i <= n; i++)
                {
                        result = a + d;
                        a = b;
                        b = c;
                        c = d;
                        d = result;
                }

                printf("%d\n", result);
        }

}
```

```c
int main(void)
{
        function(25);

        return 0;
}
```

```
1252
```

7. 대소문자를 전환하는 프로그램에 함수 개념을 적용하여 풀어보세요.

```c
#include <stdio.h>

char change_alphabet(char alp)
{
        alp ^= 32;

        return alp;
}


int main(void)
{
        char alp1, alp2;

        printf("알파벳 하나를 입력하시오:");
        scanf("%c", &alp1);

        alp2 = change_alphabet(alp1);

        printf("%c\n", alp2);

        return 0;
}
~
```

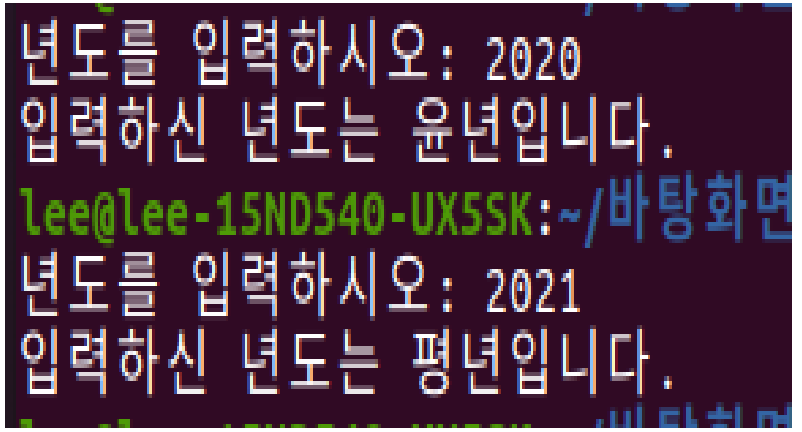
```
알파벳 하나를 입력하시오:E
e
lee@lee-15ND540-UX5SK:~/바탕화면
알파벳 하나를 입력하시오:e
E
```

아스키코드에서
소문자 알파벳과 대문자 알파벳은
십진수로 32차이가 납니다.

대문자일때는 +32를 하고
소문자일때는 -32를 하기 위하여
XOR연산을 이용하였습니다.

8. 윤년을 계산하는 프로그램을 만드세요.

```c
void leapYear(int year)
{
        if(year % 4 == 0 && year % 100 != 0 || year % 400 == 0)
                printf("입력하신 년도는 윤년입니다.\n");

        else
                printf("입력하신 년도는 평년입니다.\n");
}

int main(void)
{
        int year;

        printf("년도를 입력하시오: ");
        scanf("%d", &year);

        leapYear(year);

        return 0;
}
```



윤년의 조건

1. 4의배수 이면서 100의 배수는 아니다.

2. 400의배수이다.

위 두 조건중 하나만 만족하면 됩니다.

조건문을 사용하여 윤년 판별함수를 만들었습니다.

9. 0 ~ 3 까지 x^2에 대한 정적분을 수행하는 프로그램을 작성하세요

```c
int square(int x, int y)
{
        int i;
        int result = x;

        for(i = 0; i < y - 1; i++)
        {
                result *= x;
        }

        return result;
}
```

반복문을 사용하여 x ^ y를 계산하는
거듭제곱 함수를 만들었습니다.

```c
void integral(int start, int end, int n)
{
        float a, b, c, result;

        if(n == -1)
        {
                printf("ln %f \n", (float) end / start);
        }
        else
        {
                n += 1;
                printf("(1 / %d) * x ^ %d \n", n, n);
                a = (float) 1 / n;
                b = a * square(start, n);
                c = a * square(end, n);
                result = c - b;

                printf("%f\n", result);
        }
}
```

x^n 의 부정적분은
(1 / n+1) * x ^ (n+1)이고,

그리고 정적분 결과는
부정적분식에 마지막값을 대입한것
- 처음값을 대입한것

위의 두 공식을 이용하여
X^n의 정적분 함수를 만들었습니다.

```c
int main(void)
{
        int start, end, n;

        printf("적분하고자 하는 범위를 입력하시오.\n");
        printf("시작: ");
        scanf("%d", &start);

        printf("끝: ");
        scanf("%d", &end);

        printf("적분하고자 하는 x의계수 n을 입력하세요. 조건: n은 정수\nn: ");
        scanf("%d", &n);

        integral(start, end, n);

        return 0;
}
```

```
적분하고자 하는 범위를 입력하시오.
시작: 1
끝: 3
적분하고자 하는 x의계수 n을 입력하세요. 조건: n은 정수
n: 2
(1 / 3) * x ^ 3
8.666667
```