


C 언어 EXPRESS(개정3판)



제 7장 반보문



이번 장에서 학습할 내용

- 
- 반복의 개념 이해
 - while 반복문
 - do-while 반복문
 - for 반복문
 - break와
continue문



왜 반복이 중요한가?

```
printf("Hello World! \n");  
printf("Hello World! \n");  
printf("Hello World! \n");  
printf("Hello World! \n");  
printf("Hello World! \n");
```



```
for (i = 0; i < 5; i++)  
    printf("Hello World! \n");
```



while 문

Syntax: while 문

예

```
while( i < 10 )  
    printf("Hello World!\n");
```

조건식

조건식이 참이면 문장을 반복 실행한다.



예제

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

```
int main(void)
```

```
{
```

```
    int i = 0;
```

```
    while( i < 5 )
```

```
    {
```

```
        printf("Hello World! \n");  
        i++;
```

```
    }
```

```
    return 0;
```

```
}
```

반복 조건

반복 내용



예제 #1

```
// while 문을 이용한 구구단 출력 프로그램
#include <stdio.h>

int main(void)
{
    int n;
    int i = 1;

    printf("출력하고 싶은 단: ");
    scanf_s("%d", &n);

    while (i <= 9)
    {
        printf("%d * %d = %d \n", n, i, n*i);
        i++;
    }

    return 0;
}
```



예제 #2

```
// while 문을 이용한 제곱값 출력 프로그램
#include <stdio.h>

int main(void)
{
    int n;

    printf("=====\n");
    printf("  n      n의 제곱 \n");
    printf("=====\n");

    n = 1;
    while (n <= 10)
    {
        printf("%5d   %5d\n", n, n*n);
        n++;
    }

    return 0;
}
```



예제 #3

- 1부터 n까지의 합 계산하는 프로그램
- $1 + 2 + 3 + \dots + n$
- 1씩 증가하는 변수 필요
- 증가한 값을 더할 변수 필요



예제 #3

```
#include <stdio.h>

int main(void)
{
    int i, n, sum;                // 변수 선언

    printf("정수를 입력하시오:"); // 입력 안내 메시지 출력
    scanf_s("%d", &n);           // 정수값 입력

    i = 1;                        // 변수 초기화
    sum = 0;

    while(i <= n)
    {
        sum += i;                // sum = sum + i;와 같다.
        i++;                    // i = i + 1과 같다.
    }

    printf("1부터 %d까지의 합은 %d입니다\n", n, sum);
    return 0;
}
```



예제 #4 입력한 수까지의 짝수합

```
#include <stdio.h>

int main(void)
{
    int i, n, sum;                // 변수 선언

    printf("정수를 입력하시오:"); // 입력 안내 메시지 출력
    scanf_s("%d", &n);           // 정수값 입력

    i = 1;                        // 변수 초기화
    sum = 0;

    while(i <= n)
    {
        sum += i;                // sum = sum + i;와 같다.
        i = i + 2;

    }

    printf("1부터 %d까지의 짝수합은 %d입니다\n", n, sum);
    return 0;
}
```



예제 #5 입력을 5번 받아 입력한 수 합계구하기

```
// while 문을 이용한 합계 프로그램
#include <stdio.h>

int main(void)
{
    int i, n, sum;

    i = 0;                // 변수 초기화
    sum = 0;              // 변수 초기화
    while (i < 5)
    {
        printf("값을 입력하시오: ");
        scanf_s("%d", &n);
        sum = sum + n;    // sum += n;과 같다.
        i++;
    }
    printf("합계는 %d입니다.\n", sum);

    return 0;
}
```



if 문과 while 문의 비교

```
if( 조건 )
```

```
{
```

```
...
```

```
...
```

```
}
```



조건이 만족되면
한번만 실행
된다.

```
while( 조건 )
```

```
{
```

```
...
```

```
...
```

```
}
```



조건이 만족되면
여러 번 반복 실행
된다.



참과 거짓

```
#include <stdio.h>
int main(void)
{
    int i = 3;
    while (i)
    {
        printf("%d은 참입니다.", i);
        i--;
    }
    printf("%d은 거짓입니다.", i);
}
```

3은 참입니다.
2은 참입니다.
1은 참입니다.

- 숫자 0은 거짓으로 인식
- 숫자 0 이외의 수는 모두 참으로 인식



while의 끝 체크하기

```
#define _CRT_SECURE_NO_WARNINGS
// while 문을 이용한 성적의 평균 구하기 프로그램
#include <stdio.h>

int main(void)
{
    int grade, n;
    float sum, average;

    // 필요한 변수들을 초기화한다.
    n = 0;
    sum = 0;
    grade = 0;

    printf("성적 입력을 종료하려면 음수를 입력하시오\n");
```



센티넬 예제 2/2

```
// 성적을 입력받아서 합계를 구하고 학생 수를 센다.
while (grade >= 0)
{
    printf("성적을 입력하시오: ");
    scanf_s("%d", &grade);

    sum += grade;
    n++;
}

sum = sum - grade; // 마지막 데이터를 제거한다.
n--;              // 마지막 데이터 개수 제거한다.

average = sum / n; // 평균을 계산하고 화면에 출력한다
printf("성적의 평균은 %f입니다.\n", average);

return 0;
}
```



lab: 최대 공약수 찾기

- 유클리드 알고리즘

- ① 두 수 가운데 큰 수를 x , 작은 수를 y 라 한다.
- ② y 가 0이면 공약수는 x 와 같다.
- ③ $r \leftarrow x \% y$
- ④ $x \leftarrow y$
- ⑤ $y \leftarrow r$
- ⑥ 단계 ②로 되돌아간다.



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

```
int main(void)
```

```
{
```

```
    int x, y, r;
```

```
    printf("두개의 정수를 입력하시오(큰수, 작은수): ");
```

```
    scanf_s("%d%d", &x, &y);
```

```
    while (y != 0)
```

```
    {
```

```
        r = x % y;
```

```
        x = y;
```

```
        y = r;
```

```
    }
```

```
    printf("최대 공약수는 %d입니다.\n", x);
```

```
    return 0;
```

```
}
```



lab: 반감기

- 반감기: 어떤양이 초기값의 절반이 되는데 걸리는 시간
- 물질의 양이 반으로 줄어드는데 걸리는 시간을 입력받아 물질의 양이 초기의 1/10이 되는데 걸리는 시간을 구하기

반감기를 입력하시오(년): 10
10년 후에 남은 양=50.000000
20년 후에 남은 양=25.000000
30년 후에 남은 양=12.500000
40년 후에 남은 양=6.250000
1/10 이하로 되기 까지 걸린 시간=40년



사용자로부터 반감기를 입력받는다.

while(물질의 양 > 초기 물질의 양*0.1)

반감기만큼 시간을 더한다.

물질의 양은 1/2로 줄어든다.

현재 물질의 양을 출력한다.

10% 이하로 되기까지 걸린 시간을 출력한다.



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

```
int main(void)
```

```
{
```

```
    int halflife;
```

```
    double initial;
```

```
    double current;
```

```
    int years=0;
```

```
    printf("반감기를 입력하시오(년): ");
```

```
    scanf_s("%d", &halflife);
```

```
    initial = 100.0;
```

```
    current = initial;
```

```
    while( current > initial/10.0 ){
```

```
        years += halflife;
```

```
        current = current / 2.0;
```

```
        printf("%d년 후에 남은 양=%f", years, current);
```

```
    }
```

```
    printf("1/10 이하로 되기까지 걸린 시간=%d년", years);
```

```
    return 0;
```

```
}
```



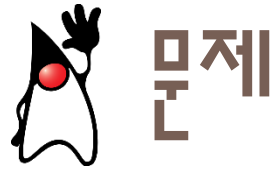
문제

정수를 입력받아서 3의 배수가 아닌 경우에는 아무 작업도 하지 않고
3의 배수인 경우에는 3으로 나눈몫을 출력하는 작업을 반복하다가
-1이 입력되면 종료하는 프로그램을 작성하시오.

* 입출력예의 진한 글씨는 실행값이다.

입·출력 예

```
5
12
4
21
7
100
-1
```



문제

삼각형의 밑변의 길이와 높이를 입력 받아 넓이를 출력하고, “ Continue? ” 에서 하나의 문자를 입력 받아 그 문자가 ‘Y’ 나 ‘y’ 이면 작업을 반복하고 다른 문자이면 종료하는 프로그램을 작성하시오.

(넓이는 반올림하여 소수 첫째자리까지 출력한다.)

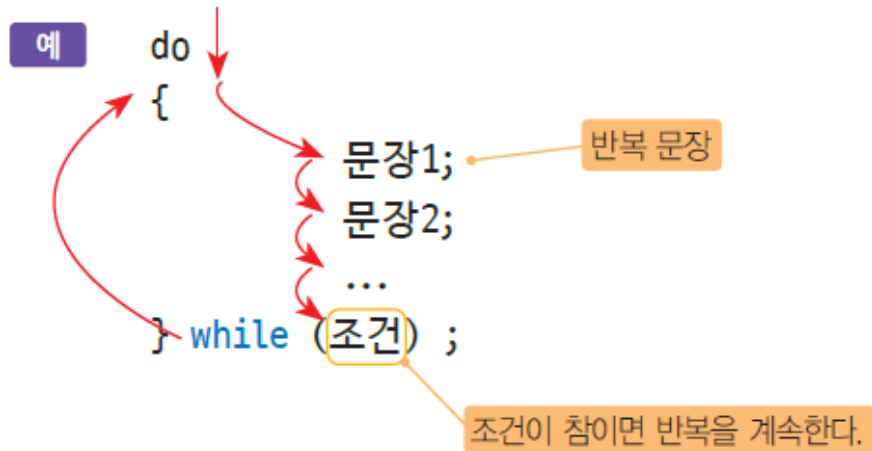
입·출력 예

```
Base = 11
Height = 5
Triangle width = 27.5
Continue? Y
Base = 10
Height = 10
Triangle width = 50.0
Continue? N
```



do...while문

Syntax: do...while문





// 사용자가 0을 입력할 때까지 숫자를 더한다.

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

```
int main(void)
```

```
{
```

```
    int number, sum = 0;
```

// 루프 몸체가 적어도 한번은 실행된다.

```
do
```

```
{
```

```
    printf("정수를 입력하시오: ");
```

```
    scanf_s("%d", &number);
```

```
    sum += number;
```

```
} while (number != 0);
```

```
printf("숫자들의 합 = %d \n", sum);
```

```
return 0;
```

```
}
```

정수를 입력하시오: 10

정수를 입력하시오: 20

정수를 입력하시오: 30

정수를 입력하시오: 0

숫자들의 합 = 60



예제 #2

1---새로만들기
2---파일열기
3---파일닫기
하나를 선택하시요: 1
선택된 메뉴=1



정가점거 중간문제

1. 다음 코드의 출력을 쓰시오.

```
int n = 0;  
do {  
    printf("%d\n", n);  
    n = n + 1;  
} while( n < 3 );
```



lab: 숫자 추측 게임

- 프로그래밍이 가지고 있는 정수를 사용자가 알아맞히는 게임

정답을 추측하여 보시오: 10
제시한 정수가 낮습니다.
정답을 추측하여 보시오: 30
제시한 정수가 낮습니다.
정답을 추측하여 보시오: 60
제시한 정수가 높습니다.
정답을 추측하여 보시오: 59
축하합니다. 시도횟수=4





알고리즘

do

사용자로부터 숫자를 `guess`로 입력받는다.

시도횟수를 증가한다.

if(`guess < answer`)

숫자가 낮다고 출력한다.

if(`guess > answer`)

숫자가 높다고 출력한다.

while(`guess != answer`);

“축하합니다”와 시도횟수를 출력한다.



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

```
#include <time.h>
```

```
int main(void)
```

```
{
```

```
    int answer = srand((unsigned int) time(NULL)); //정답을 난수로 구하기
```

```
    int guess;
```

```
    int tries = 0;
```

```
    do {
```

```
        printf("정답을 추측하여 보시오: ");
```

```
        scanf_s("%d", &guess);
```

```
        tries++;
```

```
        if (guess > answer) // 사용자가 입력한 정수가 정답보다 높으면
```

```
            printf("제시한 정수가 높습니다.");
```

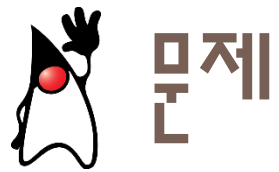
```
        if (guess < answer) // 사용자가 입력한 정수가 정답보다 낮으면
```

```
            printf("제시한 정수가 낮습니다.");
```

```
    } while (guess != answer);
```

```
        printf("축하합니다. 시도횟수=%d", tries);
```

```
    return 0;
```



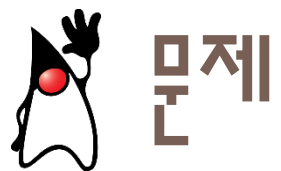
문제

do~while문을 이용하여 사용자가 0을 입력할 때까지 입력된 숫자들을 더하는 프로그램을 작성해 하시오.

결과

```
C:\Users\sjcom1-4\Desktop\sjcom\Project1.exe
정수들을 입력하시오: 10
정수들을 입력하시오: 20
정수들을 입력하시오: 30
정수들을 입력하시오: 0
숫자들의 합 = 60

-----
Process exited after 4.761 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```



do~while문을 이용하여 5개의 난수를 출력 하시오. 중복된 수는 허용하지 않습니다.



for문 구조 : 초기식, 조건식, 증감식

- 초기식

- 초기식은 반복 루프를 시작하기 전에 한번만 실행된다. 주로 변수 값을 초기화하는 용도로 사용된다.

- 조건식

- 반복의 조건을 검사하는 수식이다. 이 수식의 값이 거짓이 되면 반복이 중단된다.

- 증감식

- 한 번의 루프 실행이 끝나면 증감식이 실행된다.

```
for (i = 0; i < 5; i++)  
    printf("Hello World!\n");
```




예제 1~10까지의 합구하기

```
// 반복을 이용한 정수합 프로그램
#include <stdio.h>

int main(void)
{
    int i, sum;

    sum = 0;
    for(i = 1; i <= 10; i++)
        sum += i;                // sum = sum + i;와 같음

    printf("1부터 10까지의 정수의 합= %d\n",sum);

    return 0;
}
```



예제 1 ~ 입력받은 수까지 수들의 세제곱 구하기

```
// 반복을 이용한 세제곱값구하기
```

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

```
int main(void)
```

```
{
```

```
    int i, n;
```

```
    printf("정수를 입력하시요:");
```

```
    scanf_s("%d", &n);
```

```
    printf("=====\n");
```

```
    printf("   i   i의 세제곱\n");
```

```
    printf("=====\n");
```

```
    for(i = 1; i <= n; i++)
```

```
        printf("%5d   %5d\n", i, i*i*i);
```

```
    return 0;
```

```
}
```

정수를 입력하시요:5

=====

i	i의 세제곱
---	--------

=====

1	1
---	---

2	8
---	---

3	27
---	----

4	64
---	----

5	125
---	-----



예제 수를 입력받고 factorial 구하기

// 반복을 이용한 팩토리얼 구하기

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

```
int main(void)
```

```
{
```

```
    long fact=1;
```

```
    int i, n;
```

```
    printf("정수를 입력하시요:");
```

```
    scanf_s("%d", &n);
```

```
    for(i = 1; i <= n; i++)
```

```
        fact = fact * i;
```

```
    printf("%d!은 %d입니다.\n",n,fact);
```

```
    return 0;
```

```
}
```

정수를 입력하시요: 10
10!은 3628800입니다.

while 루프와 for 루프와의 관계

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

```
int main()
{
    int i=1;
    while (i <= 5)
    {
        printf("%d ", i);
        i++;
    }
    return 0;
}
```

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

```
int main()
{
    int i;
    for (i = 1; i <= 5; i++)
    {
        printf("%d ", i);
    }
    return 0;
}
```



팩토리얼 계산 예제(while 버전)

```
// 반복을 이용한 팩토리얼 구하기
#include <stdio.h>
int main(void)
{
    long fact = 1;
    int i = 1, n;
    printf("정수를 입력하세요: ");
    scanf_s("%d", &n);
    while (i <= n)
    {
        fact = fact * i;
        i++;
    }
    printf("%d!은 %d입니다.", n, fact);
    return 0;
}
```

정수를 입력하세요: 10
10!은 3628800입니다.



다양한 증가수식의 형태

```
for (int i = 10; i > 0; i-- )  
    printf("Hello World!\n");
```

뺄셈 사용

```
for (int i = 0; i < 10; i += 2 )  
    printf("Hello World!\n");
```

2씩 증가

```
for (int i = 1; i < 10; i *= 2 )  
    printf("Hello World!\n");
```

2를 곱한다.

```
for (int i = 0; i < 100; i = (i * i) + 2 )  
    printf("Hello World!\n");
```

수식도 가능



다양한 증가수식의 형태

```
for ( ; ; )  
    printf("Hello World!\n");
```

무한 반복 루프

```
for ( ; i < 100; i++ )  
    printf("Hello World!\n");
```

한 부분이 없을 수도 있다.

```
for (i = 0, k = 0; i < 100; i++ )  
    printf("Hello World!\n");
```

2개 이상의 변수 초기화

```
for (printf("반복시작"), i = 0; i < 100; i++ )  
    printf("Hello World!\n");
```

어떤 수식도 가능

```
for (i = 0; i < 100 && sum < 2000; i++ )  
    printf("Hello World!\n");
```

어떤 복잡한 수식도 조건
식이 될 수 있다.



정가점거

1. 다음 코드의 출력을 쓰시오.

```
for(i = 1; i < 5; i++)  
    printf("%d ", 2 * i);
```

2. 다음 코드의 출력을 쓰시오.

```
for(i = 10; i > 0; i = i - 2)  
    printf("Student%d\n", i);
```




문제

10 이하의 과목수 n 이 주어진다.

정수로 주어진 n 개 과목의 점수를 입력받아서 실수 평균을 구하여 출력하고

평균이 80점이상이면 "pass", 80점 미만이면 "fail"이라고 출력하는 프로그램을 작성하시오.

평균은 반올림하여 소수 첫째자리까지 출력한다.

입력 예

```
4
75 80 85 90
```

출력 예

```
avg : 82.5
pass
```



문제

한 개의 자연수를 입력받아 그 수의 배수를 차례로 10개 출력하는 프로그램을 작성하시오.

입력 예

5

출력 예

5 10 15 20 25 30 35 40 45 50



- 중첩 반복문(nested loop): 반복문 안에 다른 반복문이 위치

$i = 2; i \leq 9; i++$

$k = 1; k \leq 9; k++$

$i * k$



예제

// 중첩 for 문을 이용하여 *기호를 사각형 모양으로 출력하는 프로그램

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

```
int main(void)
```

```
{
```

```
    int x, y;
```

```
    for(y = 0; y < 5; y++)
```

```
    {
```

```
        for(x = 0; x < 10; x++)
```

```
        {
```

```
            printf("*");
```

```
        }
```

```
        printf("\n");
```

```
    }
```



```
*****  
*****  
*****  
*****  
*****
```



예제

```
#include <stdio.h>

int main(void)
{
    int x, y;
    for(y = 1; y <= 5; y++)
    {
        for(x = 0; x < y; x++)
            printf("*");
        printf("\n");      // 내부 반복문이 종료될 때마다 실행
    }

    return 0;
}
```



```
*
**
***
****
*****
```



문제

아래와 같이 출력되는 프로그램을 작성하시오.

출력 예

```
2 3 4 5 6
3 4 5 6 7
4 5 6 7 8
5 6 7 8 9
6 7 8 9 10
```



문제

정수를 입력받아 다음과 같이 순서쌍을 출력하는 프로그램을 작성하시오.

* 주의

')'와 '(' 사이에 공백이 1칸 있다.

(1, _1) 처럼 출력한다 : '_'는 공백

입력 예

4

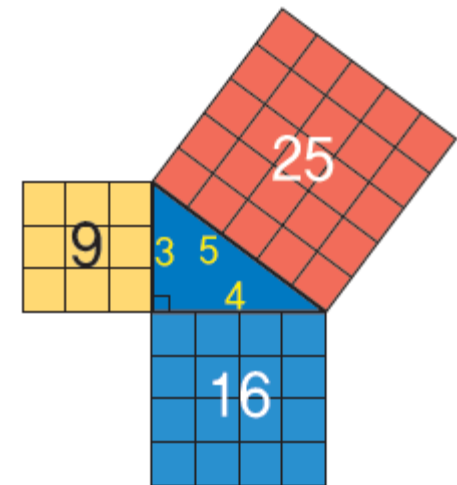
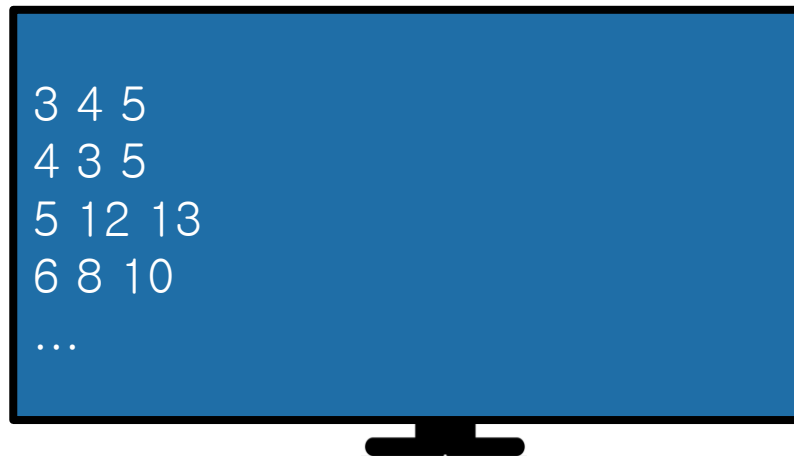
출력 예

```
(1, 1) (1, 2) (1, 3) (1, 4)
(2, 1) (2, 2) (2, 3) (2, 4)
(3, 1) (3, 2) (3, 3) (3, 4)
(4, 1) (4, 2) (4, 3) (4, 4)
```



실습: 직각 삼각형 찾기

- 각 변의 길이가 100보다 작은 삼각형 중에서 피타고라스의 정리가 성립하는 직각 삼각형은 몇 개나 있을까?





알고리즘

```
for(a=1;a<=100;a++)
```

```
    for(b=1;b<=100;b++)
```

```
        for(c=1;c<=100;c++)
```

```
            if(  $a*a + b*b == c*c$  )
```

a와 b와 c를 화면에 출력한다.




```
#include <stdio.h>
int main(void)
{
    for(int a=1; a<=100; a++)
        for(int b=1; b<=100; b++)
            for(int c=1; c<=100; c++)
                if( (a*a+b*b)==c*c )
                    printf("%d %d %d\n", a, b, c);
    return 0;
}
```



도전문제

- 위의 문제의 실행 결과를 자세히 보면 (3, 4, 5), (4, 3, 5), (5, 3, 4)와 같이 동일한 삼각형이 되풀이하여 출력되는 것을 알 수 있다. (3, 4, 5)와 같은 삼각형이 한번만 출력되게 하려면 소스의 어떤 부분을 수정하여야 할까?



```
3 4 5
5 12 13
6 8 10
7 24 25
8 15 17
9 12 15
...
```



Solution

```
#include <stdio.h>
int main(void)
{
    for(int a=1; a<=100; a++)
        for(int b=a; b<=100; b++)
            for(int c=b; c<=100; c++)
                if( (a*a+b*b)==c*c )
                    printf("%d %d %d\n", a, b, c);
    return 0;
}
```



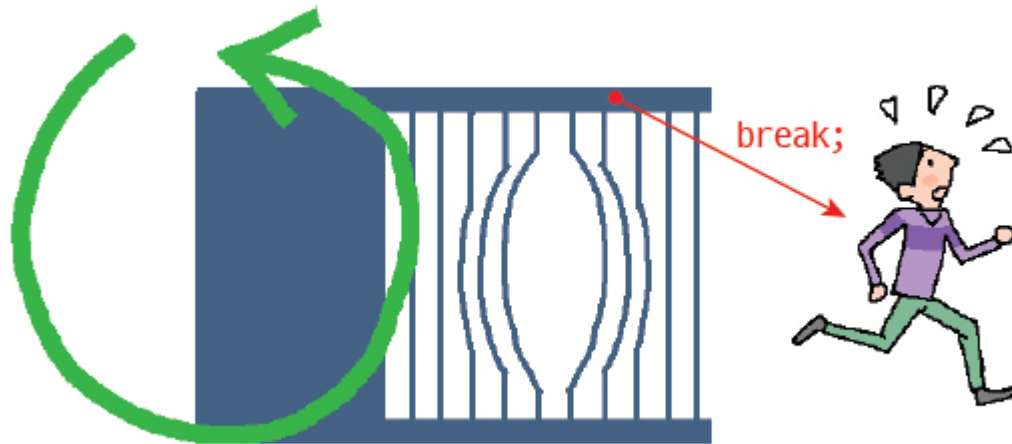
도전문제

- 위와 비슷한 문제를 하나 더 작성해보자. 라스베가스와의 같은 도박장에 가면 주사위 게임이 있다. 주사위 2개를 던졌을 때, 합이 6이 되는 경우를 전부 출력하여 보자. 예를 들어서 (1, 5), (2, 4),...와 같이 출력되면 된다. 또 주사위 3개를 사용하여서 합이 10이 되는 경우를 전부 출력하여 보자.



break 문

- break 문은 반복 루프를 빠져 나오는데 사용된다.





예제

- 100만원으로 재테크를 시작한 사람이 1년에 30%의 수익을 얻는다면 몇 년 만에 원금의 10배가 되는지를 계산하여 보자.
- 이런 경우에는 무한 반복 구조를 사용하고 조건이 만족되었을 때 `break`문이 실행되도록 하면 좋다.



예제

```
#include <stdio.h>
#define SEED_MONEY 1000000

int main(void)
{
    int year=0, money=SEED_MONEY;
    while(1)
    {
        year++;
        money += money*0.30;
        if( money > 10*SEED_MONEY )
        {
            break;
        }
    }
    printf("%d", year);
    return 0;
}
```

원금의 10배가 되면



예제

- 여기서는 무한 루프를 만들어서 사용자로부터 입력받은 실수의 제곱근을 구하여 출력하는 프로그램을 작성하여 보자.
- 허수는 생각하지 않는다고 하면 제곱근은 양의 실수에 대해서만 계산할 수 있으므로 만약 입력된 값이 음수이면 무한 루프를 종료하도록 하자. 무한 루프를 종료하는데 **break** 문을 사용한다.



// break를 이용하여 무한루프를 탈출한다.

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

```
#include <math.h>
```

```
int main(void)
```

```
{
```

```
    double v;
```

```
    while(1)
```

```
    {
```

```
        printf("실수값을 입력하시오: ");
```

```
        scanf("%lf", &v);
```

```
        if( v < 0.0 )
```

```
            break;
```

```
        printf("%f의 제곱근은 %f입니다.\n", v, sqrt(v));
```

```
    }
```

```
    return 0;
```

```
}
```

실수값을 입력하시오: 9.0

9.000000의 제곱근은 3.000000입니다.

실수값을 입력하시오: 25.0

25.000000의 제곱근은 5.000000입니다.

실수값을 입력하시오: -1



goto문이 필요한 유일한 경우

- 중첩 루프 안에서 어떤 문제가 발생했을 경우, goto를 이용하면 단번에 외부로 빠져 나올 수 있다.
- break를 사용하면, 하나의 루프만을 벗어 날 수 있다.

```
for(i=0;i<10;i++){
```

```
    for(j=1;j<=10;j++){
```

```
        // 어떤 작업
```

```
        break;
```

```
        // 어떤 작업
```

```
    }
```

```
}
```





goto문의 사용

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

```
int main(void)
```

```
{
```

```
    int x, y;
```

```
    for(y = 1; y < 10000; y++)
```

```
    {
```

```
        for(x = 1; x < 50; x++)
```

```
        {
```

```
            if( kbhit() ) goto OUT;
```

```
            printf("*");
```

```
        }
```

```
        printf("\n");
```

```
    }
```

```
OUT:
```

```
    return 0;
```

```
}
```

```
*****  
*****  
*****
```



continue

- 0부터 10까지의 정수 중에서 3의 배수만 제외하고 출력하는 예제를 살펴보자.

```
#include <stdio.h>
int main(void)
{
    int i;

    for(i=0 ; i<10 ; i++)
    {
        if( i%3 == 0 )
            continue;
        printf("%d ", i);
    }
    return 0;
}
```

continue 문을 만나면 다음
반복을 즉시 시작한다.



continue 문

```
while ( 조건식 )  
{  
    문장 ;  
    문장 ;  
    continue  
    문장 ;  
}
```

```
do  
{  
    문장 ;  
    문장 ;  
    continue  
    문장 ;  
} while ( 조건식 );
```

```
for ( 초기식 ;  
      조건식 ;  
      증감식 )  
{  
    문장 ;  
    문장 ;  
    continue  
    문장 ;  
}
```



예제

- 사용자로부터 알파벳 소문자를 받아서 대문자로 바꾸는 다음의 프로그램을 살펴보자. 만약 사용자로부터 받은 문자가 소문자가 아니면 사용자로부터 다시 문자를 입력받는다.



예제

```
// 소문자를 대문자로 변경한다.  
#include <stdio.h>  
  
int main(void)  
{  
    char letter;  
  
    while(1)  
    {  
        printf("소문자를 입력하시오: ");  
        scanf_s(" %c", &letter);  
  
        if( letter == 'Q' )  
            break ;  
        if( letter < 'a' || letter > 'z' )  
            continue ;  
  
        letter -= 32;  
        printf("변환된 대문자는 %c입니다.\n", letter);  
    }  
  
    return 0;  
}
```




lab: 복리 이자 계산

=====

연도 원리금

=====

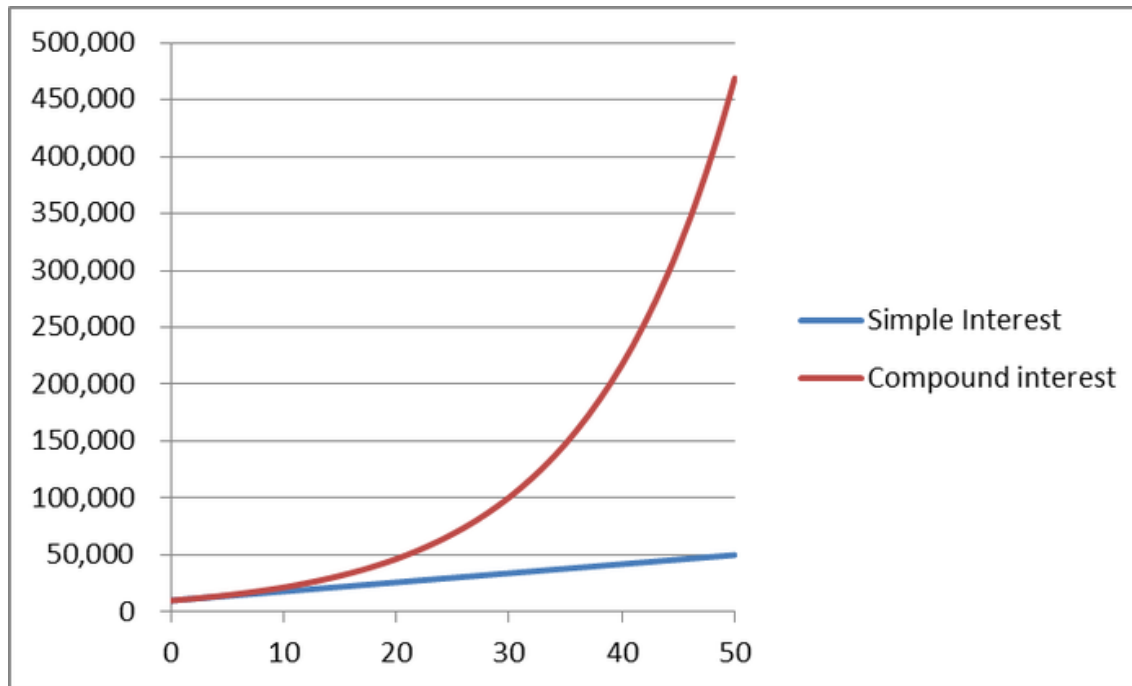
1	1050000.0
2	1102500.0
3	1157625.0
4	1215506.3
5	1276281.6
6	1340095.6
7	1407100.4
8	1477455.4
9	1551328.2
10	1628894.6





복리에서 원리금 합계

$$\text{원리합계} = \text{원금} \times (1 + \text{이율})^{\text{기간}}$$





복리에서 원리금 합계

```
#include <stdio.h>

#define RATE 0.07 // 이율
#define INVESTMENT 10000000 // 초기 투자금
#define YEARS 10 // 투자 기간

int main(void)
{
    int i;
    double total = INVESTMENT; // 원리금 합계

    printf("=====\n");
    printf("연도 원리금\n");
    printf("=====\n");

    for (i = 1; i <= YEARS; i++)
    {
        total = total * (1 + RATE); // 새로운 원리금 계산
        printf("%2d%10.1f\n", i, total);
    }

    return 0;
}
```



Lab: 자동으로 수학문제 생성하기

$$3 + 7 = 10$$

맞았습니다.

$$9 + 3 = 12$$

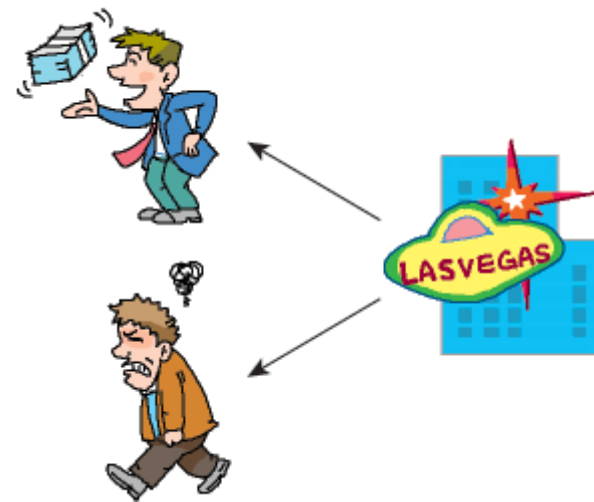
맞았습니다.

$$8 + 3 = _$$



Lab: 도박사의 확률

- 어떤 사람이 50달러를 가지고 라스베가스에서 슬롯 머신 게임을 한다고 하자. 한 번의 게임에 1달러를 건다고 가정하자. 돈을 딸 확률은 0.5이라고 가정하자(현실과는 많이 다르다). 라스베가스에 가면, 가진 돈을 다 잃거나 목표 금액인 250달러에 도달할 때까지 게임을 계속한다 (while 루프가 생각나지 않은가?). 어떤 사람이 라스베가스에 100번을 갔다면 몇 번이나 250달러를 따서 돌아올 수 있을까?





```
#include <stdio.h>
#include <stdlib.h>
```

```
int main(void)
```

```
{
```

```
    int initial_money = 50;
```

```
    int goal = 250;
```

```
    int i;
```

```
    int wins = 0;
```

```
    for (i = 0; i < 100; i++) {
```

```
        int cash = initial_money;
```

```
        while (cash > 0 && cash < goal) {
```

```
            if (((double)rand() / RAND_MAX) < 0.5) cash++;
```

```
            else cash--;
```

```
        }
```

```
        if (cash == goal) wins++;
```

```
    }
```

```
    printf("초기 금액 $%d \n", initial_money);
```

```
    printf("목표 금액 $%d \n", goal);
```

```
    printf("100번 중에서 %d번 성공\n", wins);
```

```
    return 0;
```

```
}
```

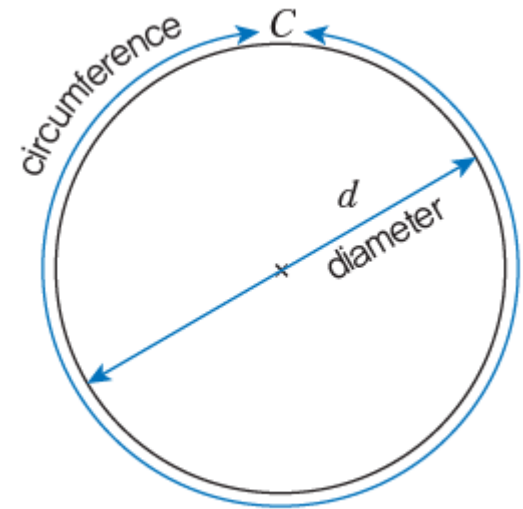
초기 금액 \$50
목표 금액 \$250
100번 중에서 20번 성공



Lab: 파이 구하기

- 파이를 계산하는 가장 고전적인 방법은 Gregory-Leibniz 무한 수열을 이용하는 것

$$\pi = \frac{4}{1} - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \dots$$





실행 결과

반복횟수:100000

$\text{Pi} = 3.141583$

계속하려면 아무 키나 누르십시오 . . .



알고리즘

사용자로부터 반복횟수 `loop_count`를 입력받는다.

```
분자 = 4.0;
```

```
분모 = 1.0;
```

```
sum = 0.0;
```

```
while(loop_count > 0)
```

```
    sum = sum + 분자 / 분모;
```

```
    분자 = -1.0* 분자;
```

```
    분모 = 분모 + 2.0;
```

```
    --loop_count;
```

`sum`을 출력한다.



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

```
int main(void)
```

```
{
```

```
    double divisor, dividend, sum;
```

```
    int loop_count;
```

```
    divisor = 1.0;
```

```
    dividend = 4.0;
```

```
    sum = 0.0;
```

```
    printf("반복횟수:");
```

```
    scanf_s("%d", &loop_count);
```

```
    while(loop_count > 0) {
```

```
        sum = sum + dividend / divisor;
```

```
        dividend = -1.0 * dividend;
```

```
        divisor = divisor + 2;
```

```
        loop_count--;
```

```
    }
```

```
    printf("Pi = %f", sum);
```

```
    return 0;
```

```
}
```