

C – 제어문(조건, 반복)



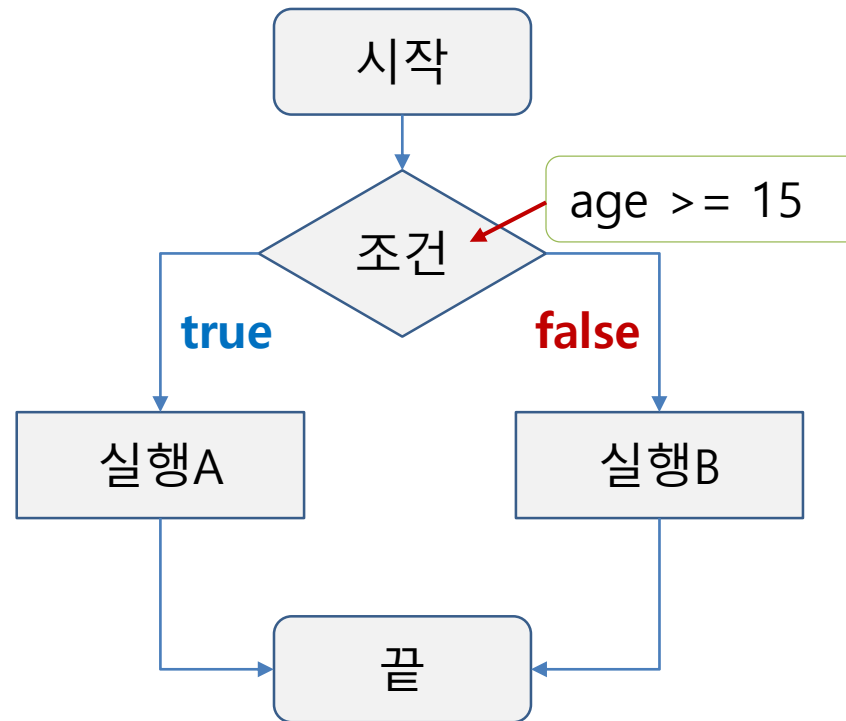
Visual Studio 2022



조건문(if문)

❖ 조건문이란?

주어진 조건에 따라 다른 수행문이 실행되도록 프로그래밍하는 명령문
if문, switch문이 대표적이다.



조건문(if문)

▪ if문

```
if(조건식){  
    수행문;  
}
```

//조건식이 참이면 수행문 실행

▪ if-else 문

```
if(조건식){  
    수행문1;  
}  
else{  
    수행문2  
}
```

//조건식이 참이면 수행문1 실행,
아니면 수행문2 실행

```
/*  
    나이가 15세 이상이면 관람가,  
    아니면 관람불가 출력  
*/  
int age = 14;  
  
/*if (age >= 15) {  
    printf("관람가입니다.\n");  
}  
printf("나이는 %d세입니다.\n", age);*/  
  
if (age >= 15) {  
    printf("관람가입니다.\n");  
}  
else {  
    printf("관람 불가입니다.\n");  
}  
printf("나이는 %d세입니다.\n", age);
```



if문 – 연습 예제

입장객 수에 따른 좌석 줄 수를 계산하는 프로그램을 작성하세요.

```
int customer; //입장객
int column;   //좌석 열
int row;      //줄(행)

printf("입장객 수 입력: ");
scanf_s("%d", &customer);

printf("좌석열 수 입력: ");
scanf_s("%d", &column);

if (customer % column == 0) //나누어 떨어짐
{
    row = customer / column; //기본이 int형임
}
else
{
    row = (int)(customer / column) + 1;
}
printf("%d개의 줄이 필요합니다.\n", row);
```

입장객 수 입력: 20
좌석열 수 입력: 5
4개의 줄이 필요합니다.

입장객 수 입력: 23
좌석열 수 입력: 5
5개의 줄이 필요합니다.



if문 – 연습 예제

윤년을 계산하는 프로그램.

```
/*
    윤년 (2월이 29일까지 있는 해) 판정하기
    - 4년에 한번 윤년이 있다.(4의 배수)
    - 100의 배수는 윤년이 아니다.
    - 400의 배수는 윤년이다.
*/
//int year = 1900;
int year;
printf("연도를 입력하세요: ");
scanf_s("%d", &year);

if ((year % 4 == 0 && year % 100 != 0) || year % 400 == 0)
{
    printf("윤년입니다.\n");
}
else
{
    printf("윤년이 아닙니다.\n");
}
```

연도를 입력하세요: 2024
윤년입니다.

연도를 입력하세요: 1900
윤년이 아닙니다.



if ~ 내부 if

- if ~ 내부 if문

```
/*  
어떤 수가 10보다 큰지 작은지 구분하고,  
10보다 큰 경우 짝수와 홀수를 구분하고,  
10보다 작은 경우 짝수와 홀수를 구분하는 프로그램  
*/  
  
int num;  
printf("수를 입력하세요: ");  
scanf_s("%d", &num);
```

```
수를 입력하세요 : 13  
13는(은) 10보다 큰 홀수입니다.
```



if ~ 내부 if

- if ~ 내부 if문

```
if (num > 10) {  
    if (num % 2 == 0) {  
        printf("%d는(은) 10보다 큰 짝수입니다.\n", num);  
    }  
    else {  
        printf("%d는(은) 10보다 큰 홀수입니다.\n", num);  
    }  
}  
else {  
    if (num % 2 == 0) {  
        printf("%d는(은) 10보다 작은 짝수입니다.\n", num);  
    }  
    else {  
        printf("%d는(은) 10보다 작은 홀수입니다.\n", num);  
    }  
}
```



if ~ else if ~ else문

▪ if - else if – else 문(다중 조건문)

```
if(조건 1){  
    수행문1;  
}  
else if(조건 2)  
    수행문2  
}  
else{  
    수행문3  
}
```

// 조건 1이 참이면 수행문1 실행, 조건2가 참이면 수행문2
실행, 조건1,2가 모두 거짓이면 수행문3 실행



if ~ else if ~ else문

▪ if - else if – else 문(다중 조건문)

```
if(조건 1){  
    수행문1;  
}  
else if(조건 2)  
    수행문2  
}  
else{  
    수행문3  
}
```

// 조건 1이 참이면 수행문1 실행, 조건2가 참이면 수행문2
실행, 조건1,2가 모두 거짓이면 수행문3 실행



if ~ else if ~ else문

▪ if - else if – else 문(다중 조건문)

```
/*  
과목의 점수에 따른 학점 계산하기  
- 90 ~ 100점 : A  
- 80 ~ 90점 : B  
- 70 ~ 80점 : C  
- 70점 미만 : F  
*/
```

```
int score; //과목 점수  
char grade; //학점
```

```
printf("점수 입력: ");  
scanf_s("%d", &score);
```

점수 입력 : 87
점수는 87점 이고 , 학점은 B입니다 .



if ~ else if ~ else문

▪ if - else if – else 문(다중 조건문)

```
if (score >= 90 && score <= 100)
{
    grade = 'A';
}
else if(score >= 80)
{
    grade = 'B';
}
else if (score >= 70)
{
    grade = 'B';
}
else
{
    grade = 'F';
}

printf("점수는 %d점이고, 학점은 %c입니다.\n", score, grade);
```



조건문(SWITCH – CASE)

▪ switch-case문

조건식의 결과가 정수 또는 문자 값이고 그 값에 따라 수행문이 결정될때 if~else if ~ else문을 대신하여 switch-case문을 사용

```
switch(변수){  
    case 변수값:  
        실행문  
        break;  
    ...  
    default:  
        실행문  
}
```

//변수값에 해당하는 case 이면 실행문
수행, 해당 값이 없으면 default 수행



조건문(SWITCH – CASE)

▪ switch ~ case문

```
/*  
    엘리베이터 타기: 1 ~ 3층까지 있는 건물  
*/  
int floor;  
  
printf("가고 싶은 층을 누르세요: ");  
scanf_s("%d", &floor);  
  
switch (floor) {  
case 1:  
    printf("1층을 눌렀습니다.\n");  
    break;  
case 2:  
    printf("2층을 눌렀습니다.\n");  
    break;  
case 3:  
    printf("3층을 눌렀습니다.\n");  
    break;  
default:  
    printf("건물에 없는 층입니다.\n");  
    break;  
}
```



조건문(SWITCH – CASE)

▪ case문 동시에 사용하기

```
int month = 10;
int day;

switch (month) {
case 1: case 3: case 5: case 7: case 8: case 10: case 12:
    day = 31;
    break;
case 2:
    day = 28;
    break;
case 4: case 6: case 9: case 11:
    day = 30;
    break;
default:
    day = 0;
    break;
}
printf("%d월은 %d일 까지 있습니다.\n", month, day);
```



반복문(while문)

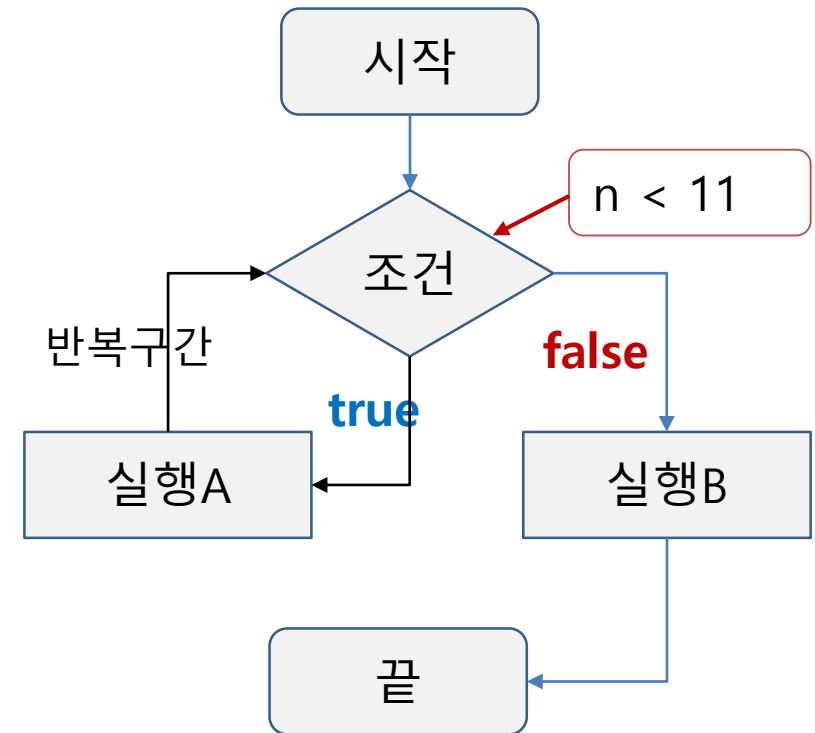
❖ 반복문

- 주어진 조건이 만족할 때까지 수행문을 반복적으로 수행함
- while, do~while, for 문이 있음

▪ while 문

조건식이 참인 동안 반복 수행

```
while(조건식){  
    수행문1;  
    ...  
}
```



반복문(while문)

▪ while문

```
int a = 1;
printf("a = %d\n", a);

a++; //a = a + 1
printf("a = %d\n", a);

a++; //a += 1
printf("a = %d\n", a);
```

```
// "안녕~"을 10번 반복하기
int n = 1; //초기값

printf("안녕~ %d\n", n++);
printf("안녕~ %d\n", n++);
printf("안녕~ %d\n", n++);
printf("안녕~ %d\n", n++);
printf("안녕~ %d\n", n++);
printf("안녕~ %d\n", n++);
printf("안녕~ %d\n", n++);
printf("안녕~ %d\n", n++);
printf("안녕~ %d\n", n++);
printf("\n");
```

```
안녕~ 1
안녕~ 2
안녕~ 3
안녕~ 4
안녕~ 5
안녕~ 6
안녕~ 7
안녕~ 8
안녕~ 9
안녕~ 10
```



반복문(while문)

■ while문

```
int a = 1;
printf("a = %d\n", a);

a++; //a = a + 1
printf("a = %d\n", a);

a++; //a += 1
printf("a = %d\n", a);
```

```
/*
   "안녕~"을 10번 반복하기
*/
int n = 1; //초기값
while (n <= 10) //종료값
{
    printf("안녕~ %d\n", n);
    n++; //증감값
}
```

```
n = 10;
while (n > 0)
{
    printf("안녕~ %d\n", n);
    n--;
}
```

```
안녕~ 1
안녕~ 2
안녕~ 3
안녕~ 4
안녕~ 5
안녕~ 6
안녕~ 7
안녕~ 8
안녕~ 9
안녕~ 10
안녕~ 10
안녕~ 9
안녕~ 8
안녕~ 7
안녕~ 6
안녕~ 5
안녕~ 4
안녕~ 3
안녕~ 2
안녕~ 1
```



반복문(while문)

▪ while문

```
/*  
    1부터 10까지 더하기  
*/  
int n = 1;  
int sum = 0; //합계 저장 변수  
  
while (n <= 10)  
{  
    sum += n;  
    printf("n=%d, sum=%d\n", n, sum);  
    n++;  
}  
printf("합계: %d\n", sum);  
  
return 0;
```

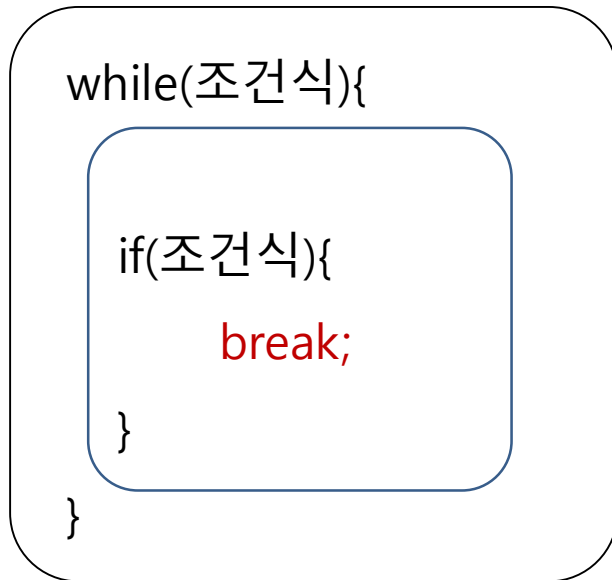
```
n=1, sum=1  
n=2, sum=3  
n=3, sum=6  
n=4, sum=10  
n=5, sum=15  
n=6, sum=21  
n=7, sum=28  
n=8, sum=36  
n=9, sum=45  
n=10, sum=55  
합 계 : 55
```



기타 제어 -break 문

▪ break 문

반복문에서 break 문을 만나면 더 이상 반복을 수행하지 않고 반복문을 빠져 나옴



while(**1**){ } – 무한 반복문
1이면 참, 1이 아니면 거짓



break 예제

■ 반복 조건문

```
/*
    1부터 10까지 더하기
    while ~ break 문
*/
int n = 1;
int sum = 0;

while (1)
{
    if (n > 10)
        break;
    printf("%-3d", n);
    n++;
}
```

```
while (1)
{
    if (n > 10)
        break;
    sum += n; //sum = sum + n
    n++;
}

printf("합계: %d\n", sum);
```

1 2 3 4 5 6 7 8 9 10

합계 : 55



반복 조건문

■ 숫자 입력 시 문자 입력으로 인한 오류 처리

```
int num;  
  
printf("숫자를 입력하세요: ");  
scanf_s("%d", &num);  
  
printf("%d\n", num);
```

숫자를 입력하세요: abc
-858993460

✓ getchar()의 반환 타입은 int인데 문자가 입력되어 오류 발생

코드

역할

```
while (getchar() != '\n');
```

입력 버퍼를 완전히 비움
(엔터(\n)가 나올 때까지 모든 문자 제거)



반복 조건문

■ 숫자 입력 시 문자 입력으로 인한 오류 처리

```
int num;
/*printf("숫자를 입력하세요: ");
scanf_s("%d", &num);
printf("%d\n", num);*/

char ch;
/*printf("1개의 문자 입력: ");
ch = getchar(); // getchar()는 char를 반환
printf("입력한 문자: %c\n", ch);
...
int ch2;
printf("1개의 문자 입력: ");
ch2 = getchar(); // getchar()는 int를 반환 (EOF(-1) 처리 위해)
printf("입력한 문자: %c\n", (char)ch2);*/
```

```
숫자를 입력하세요: ab
잘못된 입력입니다!
숫자를 입력하세요: 100
100
```



반복 조건문

■ 숫자 입력 시 문자 입력으로 인한 오류 처리

```
//문자 입력시 오류 처리
while (1) {
    printf("숫자를 입력하세요: ");
    if (scanf_s("%d", &num) == 1) { // 정상 입력 시
        printf("%d\n", num);
        break;
    }
    // getchar()의 반환 타입은 int인데 문자가 입력되어 오류 발생
    while (getchar() != '\n'); //엔터('\n')가 나올때까지 모든 문자 제거
    printf("잘못된 입력입니다!\n");
}
```

```
숫자 를 입력 하세요 : ab
잘 못 된 입력입 니다!
숫자 를 입력 하세요 : 100
100
```



do ~ while 문

▪ do ~ while 문

- while문과 동일하게 반복 기능을 수행한다.
- 꼭 한 번은 실행될 수 있도록 할때 while문 대신 사용할 수 있다.

초기값 선언

do{

// 실행문

}while(조건);

세미콜론(;)을 꼭 붙임

```
int n = 1;
```

```
do {
```

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

```
    n++;
```

```
} while (n <= 10);
```



do ~ while 문

■ 숫자 입력 받기

//1. while문 사용

```
int num = -1;
```

```
/*while (num != -1)
```

```
{
```

```
    printf("-1 입력시 종료: ");
```

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

```
}*/
```

//2. do ~ while문 사용

```
do {
```

```
    printf("-1 입력시 종료: ");
```

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

```
} while (num != -1);
```

//3. while ~ break

```
while (1)
```

```
{
```

```
    printf("-1 입력시 종료: ");
```

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

```
    if (num == -1) break;
```

```
}
```

```
printf("프로그램을 종료합니다.\n");
```

```
-1 입력시 종료: 12
-1 입력시 종료: 100
-1 입력시 종료: 8
-1 입력시 종료: -1
프로그램을 종료합니다.
```



do ~ while 문

■ 문자 입력 받기

```
// 문자 입력 받기 - 'q'를 입력하면 종료
char ch;

do {
    printf("q 입력시 종료: ");
    ch = getchar(); //문자 입력

    // 입력 버퍼 비우기
    while (getchar() != '\n');
} while (ch != 'q');

printf("프로그램 종료!\n");
```

```
q 입력시 종료: s
q 입력시 종료: k
q 입력시 종료: hello
q 입력시 종료: 코리아
프로그램 종료: q
```



자동 판매기 프로그램

▪ 커피 자판기 프로그램

- 동전 500원을 넣으면 커피가 나온다.
- 500원을 초과하면 거스름돈이 나오고 커피가 나온다.
- 500원보다 작으면 커피가 나오지 않음
- 커피는 총 5개이고 모두 소진되면 판매를 중지한다.

```
동전을 넣어주세요 : 500
커피가 나옵니다.
동전을 넣어주세요 : 500
커피가 나옵니다.
동전을 넣어주세요 : 600
커피가 나오고, 거스름돈 100원을 돌려받습니다.
동전을 넣어주세요 : 500
커피가 나옵니다.
동전을 넣어주세요 : 300
커피가 나오지 않고, 돈을 돌려줍니다.
동전을 넣어주세요 : 500
커피가 나옵니다.
커피가 모두 소진되어 판매를 중단합니다.
```



자동 판매기 프로그램

▪ 커피 자판기 프로그램

```
int money;
int coffee = 5;

while (1) {
    printf("동전을 넣어주세요: ");
    scanf_s("%d", &money);

    if (money == 500)
    {
        printf("커피가 나옵니다.\n");
        coffee -= 1;
    }
    else if (money > 500)
    {
        printf("커피가 나오고, 거스름돈 %d원을 돌려받습니다.\n", (money - 500));
        coffee -= 1;
    }
}
```



자동 판매기 프로그램

▪ 커피 자판기 프로그램

```
else {  
    printf("커피가 나오지 않고, 돈을 돌려줍니다.\n");  
}  
  
if (coffee == 0)  
{  
    printf("커피가 모두 소진되어 판매를 중단합니다.\n");  
    break;  
}  
}  
  
return 0;
```



반복문(for문)

▪ for 문

주로 조건이 횟수인 경우에 사용하는 반복문이다.
초기화식, 조건식, 증감식을 한꺼번에 작성

```
for(초기화식; 조건식; 증감식){  
    수행문;  
}
```

▪ for문 수행 과정

```
int n;  
    ① → ② ← ④  
for(n = 1; n <= 5; n++) {  
    printf(n);  
    ③ ↗  
}
```



반복문(for문)

- 1부터 10까지 출력하기

```
/*  
    1 ~ 10 까지 출력, 합계  
*/  
for (int n = 1; n <= 10; n++)  
{  
    printf("%-3d", n);  
}  
printf("\n");
```

```
1  2  3  4  5  6  7  8  9 10
```



반복문(for문)

- 1부터 10까지의 합

실습) 1부터 10까지 홀수의 합 계산하기

```
int sum = 0;

for (int n = 1; n <= 10; n++)
{
    sum = sum + n;
}
printf("1부터 10까지의 합: %d\n", sum);
```

```
1부터 10까지의 합 : 55
1부터 10까지 홀수의 합 : 25
```



디버깅(Debugging)

◆ 디버깅 작업

- 중단점 설정(F9) : 디버그 > 중단점 설정
- 실행(F10) : 디버그 > 프로시저 단위 실행

중단점
설정

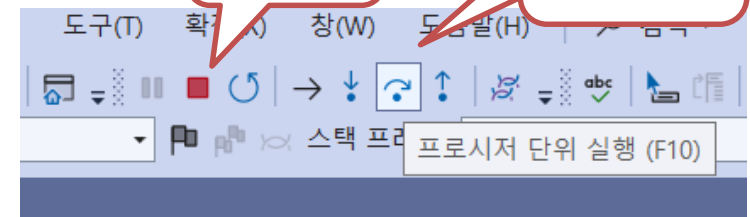
```
int sum = 0;

for (int n = 1; n <= 10; n++)
{
    sum = sum + n;
    printf("n=%d, sum=%d\n", n, sum);
}

printf("합계: %d\n", sum);
```

디버깅
중지

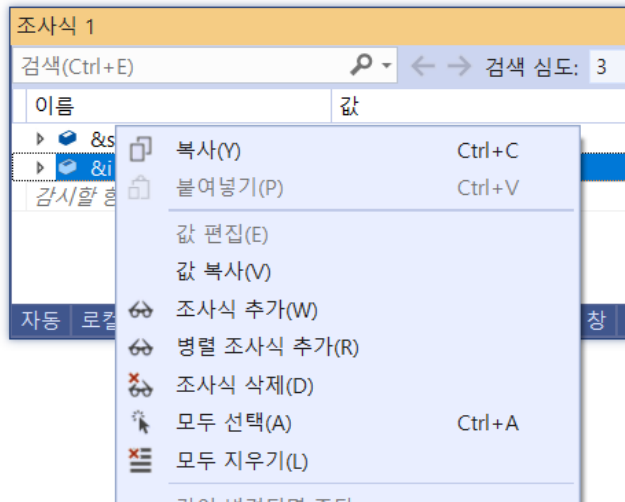
프로시저
단위실행



자동	
검색(Ctrl+E)	
← → 검색 심도:	
이름	값
n	2
sum	3
자동 로컬 조사식 1	

디버깅(Debugging)

◆ 디버깅 작업



단축메뉴 > 조사식 추가

&sum : 주소값 확인

조사식 1		
검색(Ctrl+E) 🔍 < > 검색 심도: 3		
이름	값	형식
▶ &sum	0x001df974 {3}	int *
▶ &i	0x001df980 {2}	int *
감시할 항목 추가		
자동 로컬 조사식 1 호출 스택 중단점 예외 설정 명령 창 직접 실행 창 출력 오류		



■ 구구단 프로그램 - 단을 입력받아 출력하기

```
단을 입력하세요 : 8
8 x 1 = 8
8 x 2 = 16
8 x 3 = 24
8 x 4 = 32
8 x 5 = 40
8 x 6 = 48
8 x 7 = 56
8 x 8 = 64
8 x 9 = 72
```

```
int dan;
printf("단을 입력하세요: ");
scanf_s("%d", &dan);

for (int i = 1; i <= 9; i++)
{
    printf("%d x %d = %d\n", dan, i, (dan * i));
}
```



기타 제어 – continue문

▪ continue 문

반복문과 함께 쓰이며, 반복문 내부 continue 문을 만나면 이후 반복되는 부분을 수행하지 않고 조건식이나 증감식을 수행함.

```
for(조건식){  
  
    if(조건식){  
        continue;  
    }  
}
```



기타 제어 – continue문

▪ continue 문

```
//1~10까지의 자연수 중 5와 8을 제외하고 출력하기
for (int i=1; i<=10; i++)
{
    if (i == 5 || i == 8)
        continue;
    printf("%d ", i);
}
printf("\n");
```

```
1 2 3 4 6 7 9 10
2 4 6 8 10
1부터 10까지의 짝수의 합 : 30
```



기타 제어 – continue문

▪ continue 문

```
//1~10까지의 자연수 중 짝수의 합 구하기
int n, sum;
for (n = 1, sum = 0; n <= 10; n++)
{
    if (n % 2 == 1)
        continue;
    sum += n;
    printf("%d ", n);
}
printf("\n1부터 10까지의 짝수의 합: %d\n", sum);
```



반복문(중첩 for문)

■ 중첩된 반복문(Nested Loop)

	열1	열2	열3	열4	열5
행1					
행2					
행3					
행4					
행5					

가
가
가
가
가



반복문(중첩 for문)

■ 중첩된 반복문(Nested Loop)

```
int i, j;  
for (i = 1; i <= 5; i++)  
{  
    for (j = 1; j <= 5; j++) {  
        printf("*");  
    }  
    printf("\n");  
}
```

```
/*  
    i=1, j=1  *  
           j=2  **  
           j=3  ***  
           j=4  ****  
           j=5  *****  
    i=2, j=1  *  
           j=2  **  
           j=3  ***  
           j=4  ****  
           j=5  *****  
    i=3,  
*/
```



반복문(중첩 for문)

■ 삼각형 모양의 별 찍기1

hint) 열(column)이 변하는 것에 주목한다.

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

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

```
//직각삼각형 모양의 별
for (i = 1; i <= 5; i++)
{
    for (j = 1; j <= i; j++) {
        printf("*");
    }
    printf("\n");
}
printf("\n");
```



반복문(중첩 for문)

■ 삼각형 모양의 별찍기2

hint) 공백과 별로 나눠서 생각함

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

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

```
for (i = 1; i <= 5; i++)
{
    for (j = 1; j <= 5-i; j++) {
        printf(" ");
    }
    for (j = 1; j <= i; j++) {
        printf("*");
    }
    printf("\n");
}
```



구구단 출력

■ 구구단 전체 출력

```
2단
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
```

```
3단
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
```

```
for (int i = 2; i <= 9; i++)
{
    printf("%d단\n", i);
    for (int j = 1; j <= 9; j++)
    {
        printf("%d x %d = %d\n", i, j, (i * j));
    }
    printf("\n");
}
```



구구단 출력

■ 구구단 전체 출력

```
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
```

```
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
```

```
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
```

```
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
```

```
// 3~6단까지 출력하고, 각 단은 5~8만 곱하기
for (int i = 3; i < 7; i++)
{
    for (int j = 5; j < 9; j++)
    {
        printf("%d x %d = %d\n", i, j, (i * j));
    }
    printf("\n");
}
```



반복문(for문)

- 1부터 순서대로 출력하기
- 특정 수에서 멈추기

```
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23
```

```
for (i = 0; i < 5; i++)
{
    for (j = 1; j <= 5; j++) {
        printf("%d ", i*5+j);
    }
    printf("\n");
}
```



자리배치도 프로그램

■ 자리배치도 프로그램

- 입장객 수와 좌석 열 수를 입력한다.
- 줄(행) 수에 따라 좌석 배치가 배열된다.

```
입장객 수 입력 : 23
좌석열 수 입력 : 4
좌석1 좌석2 좌석3 좌석4
좌석5 좌석6 좌석7 좌석8
좌석9 좌석10 좌석11 좌석12
좌석13 좌석14 좌석15 좌석16
좌석17 좌석18 좌석19 좌석20
좌석21 좌석22 좌석23
```



자리배치도 프로그램 만들기

■ 자리배치도 프로그램

```
int customer; //입장객
int column;   //좌석 열
int row;      //줄(행)

printf("입장객 수 입력: ");
scanf_s("%d", &customer);

printf("좌석열 수 입력: ");
scanf_s("%d", &column);

if (customer % column == 0)
{
    row = (int)(customer / column);
}
else
{
    row = (int)(customer / column) + 1;
}
```



자리배치도 프로그램 만들기

■ 자리배치도 프로그램

```
for (int i = 0; i < row; i++)
{
    for (int j = 1; j <= column; j++)
    {
        int seatNum = i * column + j; //좌석 번호
        if (seatNum > customer)
            break;
        printf("좌석%d ", seatNum);
    }
    printf("\n");
}
return 0;
```



실습 문제 1 – 반복 조건문(while)

1부터 더했을때 그 합이 100이 넘는 자연수와 합계를 구하세요.

(파일이름: SumOver100.c)

👉 실행 결과

```
N = 14  
SUM = 105
```



실습 문제2 – 중첩 for

구구단을 단보다 곱하는 수가 작거나 같은 경우까지 출력하는 프로그램
(파일이름: GugudanTest.c)

👉 실행 결과

```
2 x 1 = 2
2 x 2 = 4

3 x 1 = 3
3 x 2 = 6
3 x 3 = 9

4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16

5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
```

```
6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
6 x 9 = 54

7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
```

```
8 x 1 = 8
8 x 2 = 16
8 x 3 = 24
8 x 4 = 32
8 x 5 = 40
8 x 6 = 48
8 x 7 = 56
8 x 8 = 64
8 x 9 = 72

9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
```

