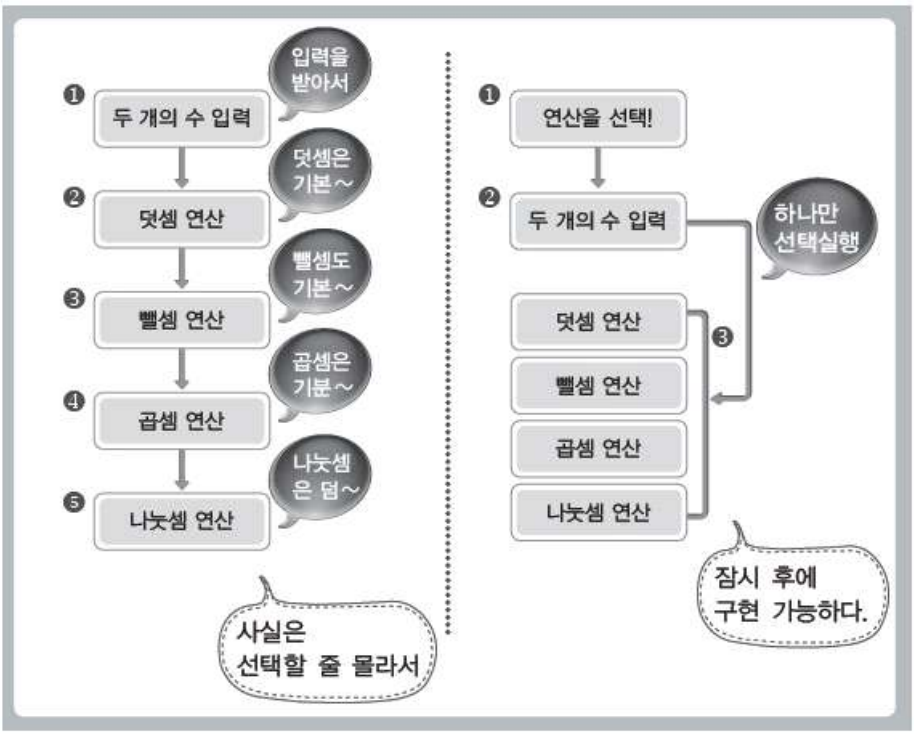
5주차 1차시 조건에 따른 분기

【학습목표】

1. 조건에 따른 흐름 분기를 설명할 수 있다.
2. 반복문의 생략과 탈출을 설명할 수 있다.

학습내용1 : 조건적 실행과 흐름의 분기

1. 흐름의 분기가 필요한 이유



√분기하지 못하면 프로그램 사용자는 사칙연산 중 하나를 선택하지 못한다!
√프로그램을 구현하다 보면 상황에 따라서 선택적으로 실행해야 하는 영역도 존재하기 마련!

2. if문을 이용한 조건적 실행

```
if(num1>num2)
{
    printf("num1이 num2보다 큽니다. \n");
    printf("%d > %d \n", num1, num2);
}
```

num1이 num2보다 크면 실행

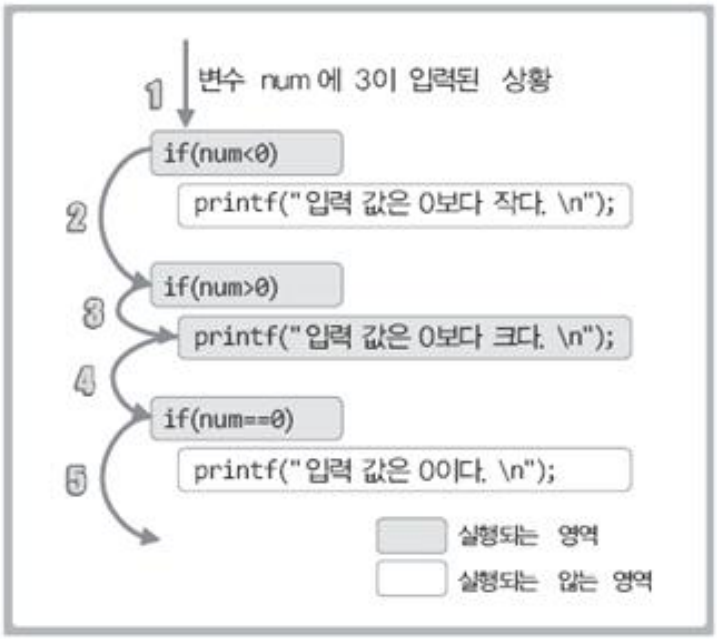
```
if(num1>num2)
    printf("num1이 num2보다 큽니다. \n");
```

한 줄이면 중괄호 생략가능

```
int main(void)
{
    int num;
    printf("정수 입력: ");
    scanf("%d", &num);

    if(num<0)    // num이 0보다 작으면 아래의 문장 실행
        printf("입력 값은 0보다 작다. \n");

    if(num>0)    // num이 0보다 크면 아래의 문장 실행
        printf("입력 값은 0보다 크다. \n");

    if(num==0)    // num이 0이면 아래의 문장 실행
        printf("입력 값은 0이다. \n");

    return 0;
}
```

```
정수 입력: 3
입력 값은 0보다 크다.
```

```
정수 입력: 0
입력 값은 0이다.
```



3. if문을 이용한 계산기 프로그램

```
int main(void)
{
    int opt;
    double num1, num2;
    double result;

    printf("1.덧셈 2.뺄셈 3.곱셈 4.나눗셈 \n");
    printf("선택? ");
    scanf("%d", &opt);
    printf("두 개의 실수 입력: ");
    scanf("%lf %lf", &num1, &num2);

    if(opt==1)
        result = num1 + num2;
    if(opt==2)
        result = num1 - num2;
    if(opt==3)
        result = num1 * num2;
    if(opt==4)
        result = num1 / num2;

    printf("결과: %f \n", result);
    return 0;
}
```

```
1.덧셈 2.뺄셈 3.곱셈 4.나눗셈
선택? 3
두 개의 실수 입력: 2.14 5.12
결과: 10.956800
```

이제 계산기 프로그램에 실질적으로 더 가까운 형태가 되었다.
프로그램 구성상 사칙연산 중 하나만 실행이 된다. 그럼에도 불구하고 프로그램 사용자가
덧셈연산을 선택할지라도 총 4번의 조건검사(if문을 통한)를 진행한다는 불합리한 점이 존재한다.
이러한 불합리한 점의 해결에 사용되는 것이 if~else문이다.

4. if~else문을 이용한 흐름의 분기

```
if(num1>num2)      num1이 num2보다 크면 실행
{    // if 블록
     printf("num1이 num2보다 큽니다. \n");
     printf("%d > %d \n", num1, num2);
}
else      num1이 num2보다 크지 않으면 실행
{    // else 블록
     printf("num1이 num2보다 크지 않습니다. \n");
     printf("%d <= %d \n", num1, num2);
}
```

if~else문은 하나의 문장임에 주목하자!
따라서 if와 else 사이에 다른 문장이 삽입될 수 없다.

```
int main(void)
{
    int num;
    printf("정수 입력: ");
    scanf("%d", &num);
    if(num<0)
        printf("입력 값은 0보다 작다. \n");
    else
        printf("입력 값은 0보다 작지 않다. \n");

    return 0;
}
```

```
정수 입력: 7
입력 값은 0보다 작지 않다.
```

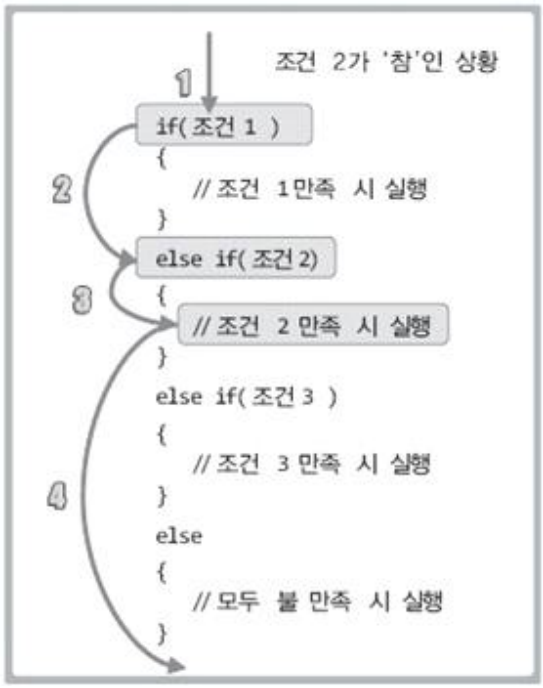
5. if...else if...else의 구성

```
if( 조건 1 )
{
   // 조건 1 만족 시 실행
}
else if( 조건 2 )
{
   // 조건 2 만족 시 실행
}
else if( 조건 3 )
{
   // 조건 3 만족 시 실행
}
else
{
   // 모두 불 만족 시 실행
}
```

얼마든지 추가 삽입 가능!

```
else if( 조건 4 )
{
   // 조건 4 만족 시 실행
}
```

if...else if...else문의 구성



if...else if...else문의 흐름

6. if...else if...else문의 적용

```
int main(void)
{
    int opt;
    double num1, num2;
    double result;
    printf("1.덧셈 2.뺄셈 3.곱셈 4.나눗셈 \n");
    printf("선택? ");
    scanf("%d", &opt);
    printf("두 개의 실수 입력: ");
    scanf("%lf %lf", &num1, &num2);

    if(opt==1)
        result = num1 + num2;
    else if(opt==2)
        result = num1 - num2;
    else if(opt==3)
        result = num1 * num2;
    else
        result = num1 / num2;

    printf("결과: %f \n", result);
    return 0;
}
```

합리적으로 완성된 사칙연산 계산기 프로그램

7. if...else if...else의 진실

if~else문은 하나의 문장임을 상기!

```
if(num<0)
    printf("입력 값은 0보다 작다. \n");
else if(num>0)
    printf("입력 값은 0보다 크다. \n");
else
    printf("입력 값은 0이다. \n");
```

```
if(num<0)
   printf("입력 값은 0보다 작다. \n");
else
      if(num>0)
   printf("입력 값은 0보다 크다. \n");
else
   printf("입력 값은 00이다. \n");
```



```
if(num<0)
{
    printf("입력 값은 0보다 작다. \n");
}
else
{
    if(num>0)
        printf("입력 값은 0보다 크다. \n");
    else
        printf("입력 값은 0이다. \n");
}
```

else에 하나의 if~else문이 속한 상황.
속한 문장이 하나일 때에는 중괄호를 생략할 수 있다!

8. 조건 연산자: 피 연산자가 세 개인 '삼 항 연산자'

```
(num1>num2) ? (num1) : (num2);

(조건) ? data1 : data2
```

조건이 참이면 data1 반환, 거짓이면 data2 반환

```
int num3 = (num1>num2) ? (num1) : (num2);

int num3 = num1;   num1>num2가 참이면 n
int num3 = num2;   um1>num2가 거짓이면
```

```
int main(void)
{
    int num, abs;
    printf("정수 입력: ");
    scanf("%d", &num);

    abs = num>0 ? num : num*(-1);
    printf("절댓값: %d \n", abs);
    return 0;
}
```

```
정수 입력: -79
절댓값: 79
```

학습내용2 : 반복문의 생략과 탈출

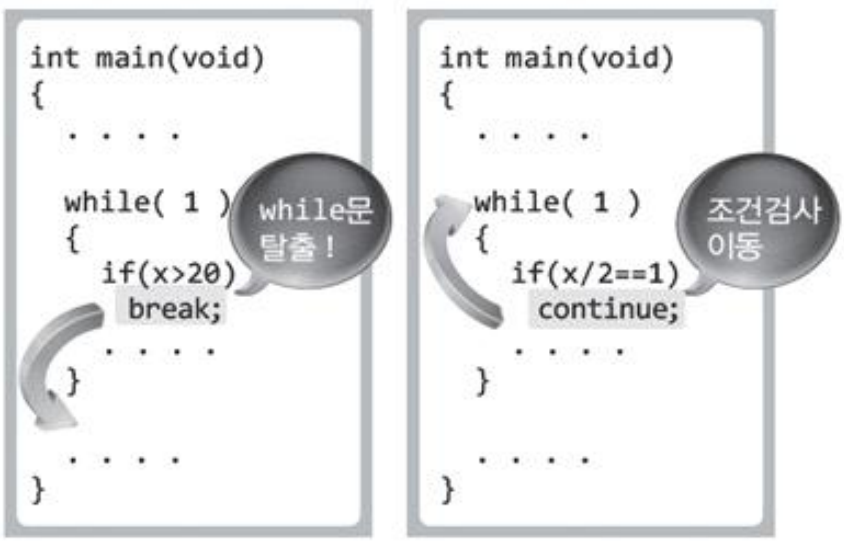
1. break! 이제 그만 빠져나가자!

```
int main(void)
{
    int sum=0, num=0;

    while(1)
    {
        sum+=num;
        if(sum>5000)
            break; // break문 실행! 따라서 반복문 탈출
        num++;
    }

    printf("sum: %d \n", sum);
    printf("num: %d \n", num);
    return 0;
}
```

```
sum: 5050
num: 100
```

break문은 자신을 감싸는 반복문 하나를 빠져 나간다.
if문과 함께 사용이 되어서 특정 조만이 만족될 때 반복문을 빠져나가는 용도로 주로 사용된다.

2. continue! 나머지 생략하고 반복조건 확인하러



continue문은 반복문을 빠져나가지 않는다! 다만 반복조건을 확인하러 올라갈 뿐이다.
그리고 반복조건이 여전히 '참'이라면 반복영역을 처음부터 실행하게 된다.

```c
int main(void)
{
    int num;
    printf("start! ");

    for(num=1; num<20; num++)
    {
        if(num%2==0 || num%3==0)
            continue;
        printf("%d ", num);
    }
    printf("end! \n");
    return 0;
}
```

```
start! 1 5 7 11 13 17 19 end!
```

【학습정리】

1. 프로그램의 흐름을 원하는 형태대로 컨트롤하기 위해서는 흐름의 분기를 사용하면 된다.

2. 분기의 가장 기본은 두 개의 키워드 if와 else로 구성되는 if~else문이다.

3. 조건 연산자는 간단한 if~else문을 대신할 수 있는 연산자로 C프로그래머들이 선호하는 연산자이다.