# 컴퓨터 프로그래밍 I
# (CSE2003-3)

**Mon/Wed 16:30-17:45 pm**

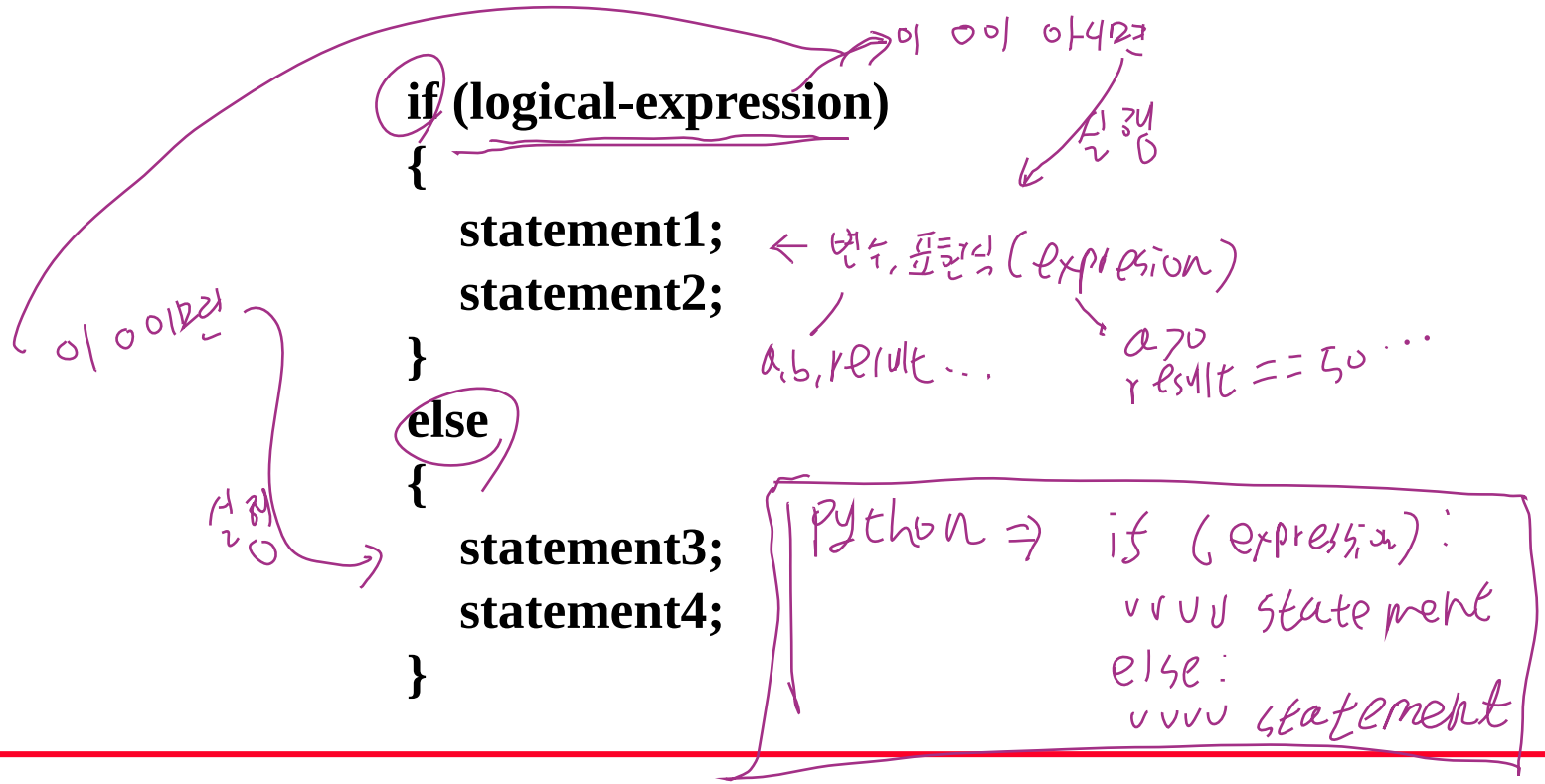
**Lecture 9**

Selection

# Selection 과 Logical Data(논리 데이터)의 값

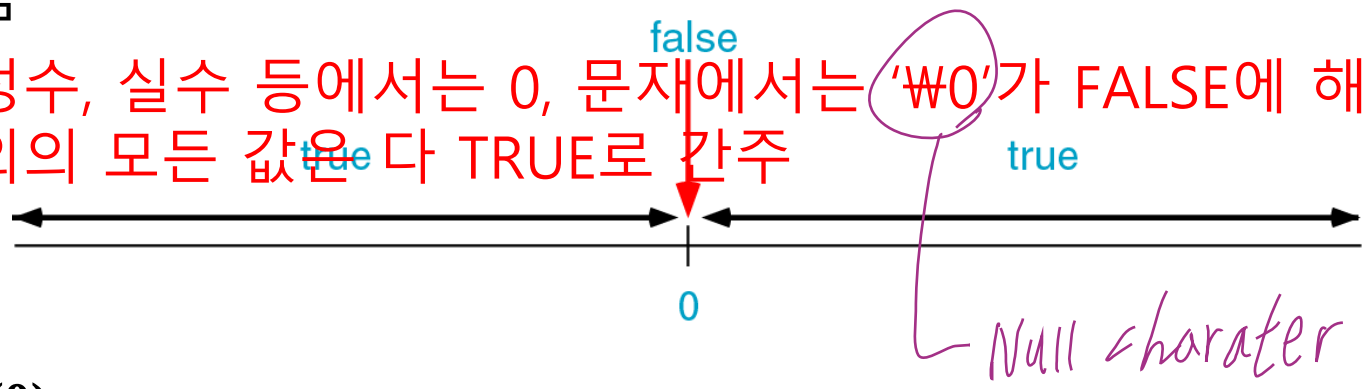
- 컴퓨터는 상황에 따라서 다른 행동을 하는 지능적인 전자 기계: Selection은 인공지능의 가장 기본적인 핵심 기능!!
- Selection은 현재 상황에서 다양한 조건들을 체크하여 조건에 따라 다른 작업을 하도록 프로그램을 만드는 방법
- Ex:

```
if (logical-expression)
{
    statement1;
    statement2;
}
else
{
    statement3;
    statement4;
}
```



# Selection 과 Logical Data(논리 데이터)의 값

- **Selection은 Logical Decision(논리적 판단)에 기반함**
  - 그러나 전통적으로 C에는 논리형 데이터(logical type)가 따로 없음
  - 데이터 값이 '0(zero)'이면 'FALSE', (nonzero) 이면 'TRUE'로 봄
  - 정수, 실수 등에서는 0, 문자에서는 '₩0'가 FALSE에 해당 그 외의 모든 값은 다 TRUE로 간주




```
if (0)
   printf("0 means TRUE")
else
   printf("0 means FALSE");
```

출력 결과: `0 means FALSE`

# Logical Expression을 만드는 방법

- **논리연산자와 비교연산자를 이용하여 logical expression을 만든다**
- **논리연산자(Logical Operator)**
  - and(&&), or(||), not(!)을 표현하는 연산자
  - 두 피연산자의 참과 거짓에 따라 연산의 결과값을 결정
    - && : 피연산자가 모두 참이면 true를 반환
    - || : 피연산자 중 하나라도 참이면 true를 반환
    - ! : 피연산자가 true이면 false를, false이면 true를 반환
- **논리 연산자의 연산 방법**
  - 연산에 참여하는 자료 값 중 0은 거짓(false)을 의미
  - 1 (또는, 0이 아닌 값)은 참(true)을 의미
  - 평가의 결과는 반드시 0(false)이거나 1(true)

# 논리연산자(Logical Operator)

| x | y | && 연산 결과 | \|\| 연산 결과 | ! 연산 결과 |
|---|---|---|---|---|
| | | x && y | x \|\| y | ! x |
| 0 (false) | 0 (false) | false | false | true |
| 0 (false) | 1 (true) | false | true | true |
| 1 (true) | 0 (false) | false | true | false |
| 1 (true) | 1 (true) | true | true | false |

<Logical Operators Truth Table>

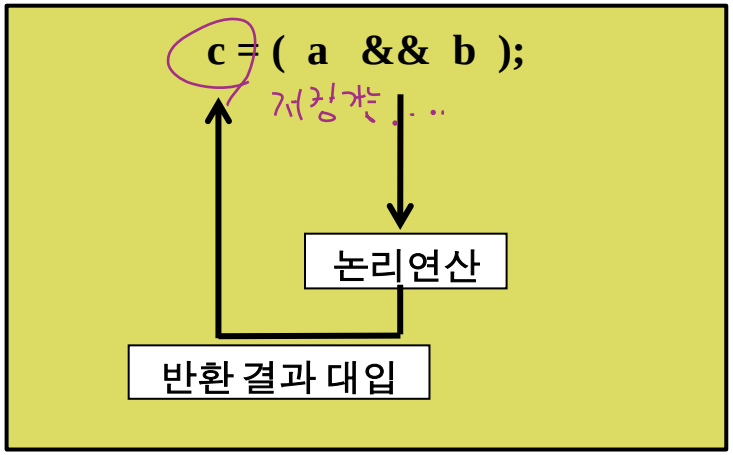
진리표.

# 논리연산자(Logical Operator)

## ◆ 예제프로그램 – 논리연산자

```
#include <stdio.h>

int main(void){
    int a = 1;
    int b = 1;
    int c = 0;

    printf("    %2d AND     %2d : %2d\n", a, b, a && b);
    printf("    %2d AND     %2d : %2d\n", a, c, a && c);
    printf("    %2d AND     %2d : %2d\n", c, a, c && a);
    printf("    %2d OR      %2d : %2d\n", a, b, a || b);
    printf("    %2d OR      %2d : %2d\n", c, a, c || a);
    printf("    %2d OR      %2d : %2d\n", c, c, c || c);
    printf("NOT %2d AND NOT %2d : %2d\n", a, c, !a && !c);
    printf("NOT %2d AND     %2d : %2d\n", a, c, !a && c);
    printf("    %2d AND NOT %2d : %2d\n", a, c, a && !c);

    return 0;
}
```

1(nonezero) –> true,
0(zero) –> false




(결과)

```
  1 AND   1 :  1
  1 AND   0 :  0
  0 AND   1 :  0
  1 OR    1 :  1
  0 OR    1 :  1
  0 OR    0 :  0
NOT 1 AND NOT  0 :  0
NOT 1 AND      0 :  0
    1 AND NOT  0 :  1
```

# 비교연산자(Comparative Operator)

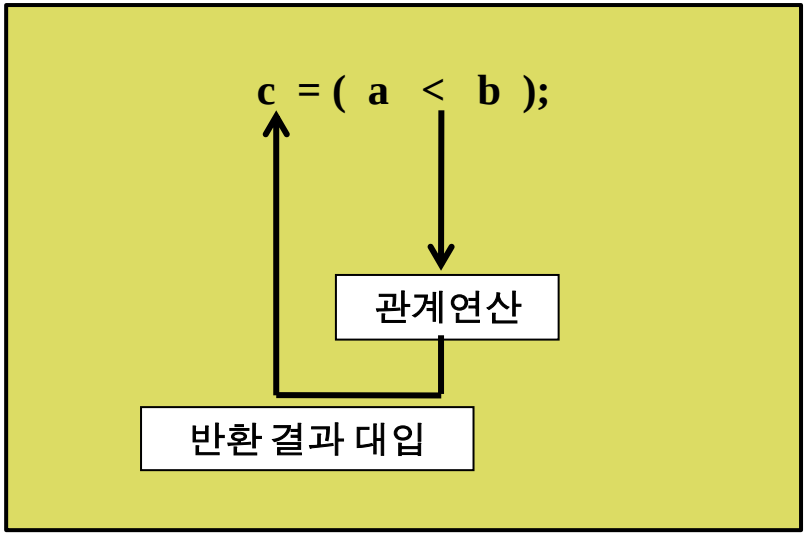
- **비교연산자 또는 관계연산자(Relational Operator)**
  - 두 값의 크기를 비교하는 연산자
    - < , > , >= , <= , == , !=
    - 두 기호의 순서가 바뀌면  에러 발생( =< , =! )

| 관계연산자 | 연산 결과 |
|---|---|
| > | 왼쪽의 값이 오른쪽의 값보다 크면 참 |
| < | 왼쪽의 값이 오른쪽의 값보다 작으면 참 |
| >= | 왼쪽의 값이 오른쪽의 값보다 크거나 같으면 참 |
| <= | 왼쪽의 값이 오른쪽의 값보다 작거나 같으면 참 |
| == | 두 값이 같으면 참 |
| != | 두 값이 같지 않으면 참 |

# 비교연산자 (Comparative Operator)

## ◆ 예제프로그램 – 비교연산자

```
#include <stdio.h>

int main(void){
        int a = 5;
        int b = -3;

        printf("%2d <  %2d is %2d\n",a,b,a <  b);
        printf("%2d == %2d is %2d\n",a,b,a == b);
        printf("%2d != %2d is %2d\n",a,b,a!=b);
        printf("%2d >  %2d is %2d\n",a,b,a > b);
        printf("%2d <= %2d is %2d\n",a,b,a <= b);
        printf("%2d >= %2d is %2d\n",a,b,a >= b);

        return 0;

}
```

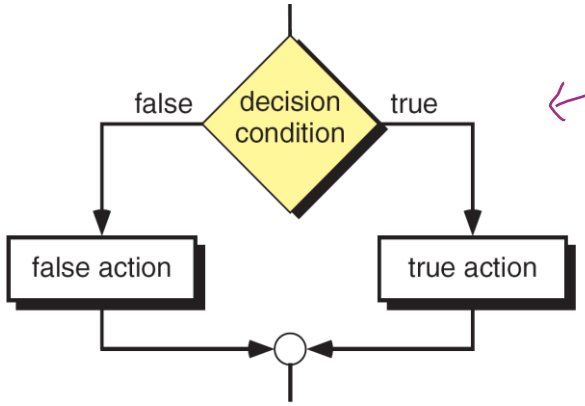



<결과>

```
5  < -3 is  0
5 == -3 is  0
5 != -3 is  1
5 >  -3 is  1
5 <= -3 is  0
5 >= -3 is  1
```

# 이중선택(Two-Way Selection)

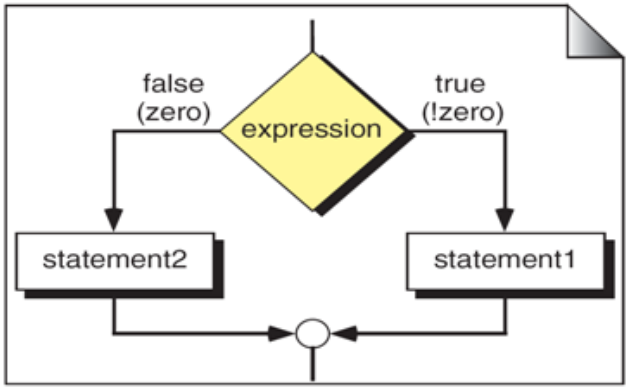
- 참(true)이나 거짓(false) 중 하나의 action만을 실행
- decision condition의 결과에 따라 제어 흐름 바꿈
- 이중선택(two-way selection)의 논리
  - 상태(condition)가 참(true)이면 true action을 취함
  - 상태(condition)가 거짓(false)이면 false action을 취함
- C언어에서는 if...else 문으로 two-way selection logic을 구현




← 순서도.

# 조건문 if...else

- **if (expression) statements**
  - expression의 결과값이 'true'인 경우
    - "statement-1" 실행
  - expression의 결과값이 'false'인 경우
    - "statement-2" 실행



(a) Logical Flow

**if 문은 선택적으로 한 번만 실행하고 넘어감**

- **if...else문의 statement는 들여쓰기 적용**
  - 관리자가 코드를 인식하기 쉽게 하기 위해 사
  - 실제 컴퓨터가 들여쓰기를 인식하는 것은 아

```
if (expression)
    statement1
else
    statement2
```
(b) Code

# if(condition)...else... (조건문)

◆ 예제 프로그램 – if...else 문의 예

```
#include <stdio.h>

int main(void){

    int a;

    printf("Please enter one integer: ");
    scanf("%d", &a);

    if(a)
            printf("%d is true\n",a);
    else
            printf("%d is false\n",a);

    return 0;
}
```

사용자로부터 임의의 값을 입력 받는다.

<결과>

```
Please enter one integer: 1
1 is true
kimjihwan@cspro:~/c_test_2$ ./test3
Please enter one integer: 0
0 is false
kimjihwan@cspro:~/c_test_2$ ./test3
Please enter one integer: 3
3 is true
```

0이 입력되면 false, 그 외 다른 수가 입력되면 true

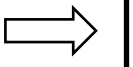
# 조건문 if...else

- 수행하는 statement가 두 줄 이상 일 때는 { }로 묶어줌
- else부분에 들어갈 statement가 없을 경우 else 부분 생략 가능 (Null else statement)

```
if (expression)
{
    statement1;
    statement2;
}
else
{
    statement3;
    statement4;
}
```

<Compound statements in an if...else>

```
if (expression)
{
    statement1;
    statement2;
}
else
    ;
```



```
if (expression)
{
    statement1;
    statement2;
}
```

이거 잘 반적.

<Null else statement>

# 조건문 if...else

```
if (j != 3)
   {
    b++;
    printf("%d", b);
   } // if
else
    printf( "%d", j );
```

Compound statements are treated as one statement

```
if (j != 5 && d == 2)
 {
  j++;
  d--;
  printf("%d%d", j, d);
 } // if
else
 {
  j--;
  d++;
  printf("%d%d", j, d);
 } // else
```

<Compound statements in an if...else>

```
if (j != 3)
   {
    b++;
    printf("%d", b);
   } // if
else
    ;
```

```
if (j != 3)
   {
    b++;
    printf("%d", b);
   } // if
```

<Null else statement>

# if(condition)...else... (조건문)

- 두 정수를 읽어서 큰 값과 작은 값을 계산

```
#include <stdio.h>

int main(void){
  int a, b, min, max;
  printf("input(1): ");
  scanf("%d", &a);
  printf("input(2): ");
  scanf("%d", &b);

  if(a < b){
    min = a;
    max = b;
  }
  else{
    min = b;
    max = a;
  }
  printf("minimum number : %d\n", min);
  printf("maximum number : %d\n", max);

  return 0;
}
```

if 조건이 참 일경우 실행

if 조건이 거짓 일경우 실행

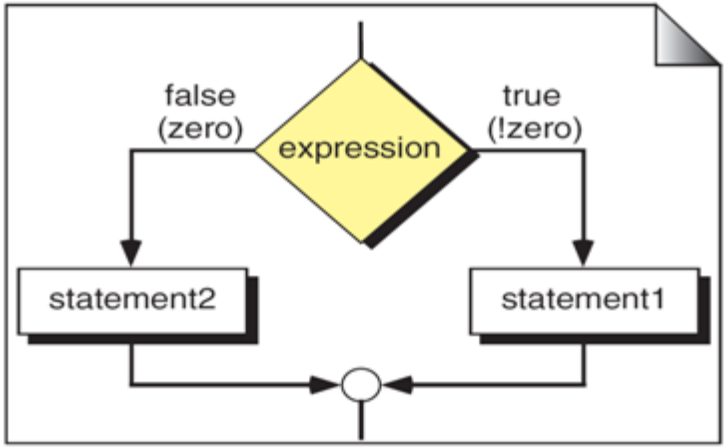
<결과>

```
input(1): 1
input(2): 7
minimum number : 1
maximum number : 7
kimjihwan@cspro:~/c_test_2$ ./test7
input(1): 5
input(2): 2
minimum number : 2
maximum number : 5
```

입력된 숫자를 비교해서 작은 수와 큰 수를 출력

# Conditional Expression(축약된 조건식)

| ‘조건’ ? A : B | x= (y>0) ? 10 : 20; | x = ((y>0) ? a*b : a/b); |
|---|---|---|
| ‘조건’이 true면 A 반환,<br>‘조건’이 false면 B반환 | y > 0가 true면 10을,<br>false면 20을 반환해서<br>x 값으로 대입 | y가 0보다 크면 a*b의 결과,<br>크지 않으면 a/b의 결과를 반환<br>x 값으로 대입 |




(a) Logical Flow

```
expression ? statement1 : statement2
```

(b) Code

# Conditional Expression(조건식)

- 양수, 음수, zero 구분하기

```
#include <stdio.h>

int main(void){
  int num;

  printf("Input your number: ");
  scanf("%d",&num);

  if(num==0)
    printf("zero\n");
  else
    (num>0)? printf("positive number\n"):printf("negative number\n");

  return 0;
}
```

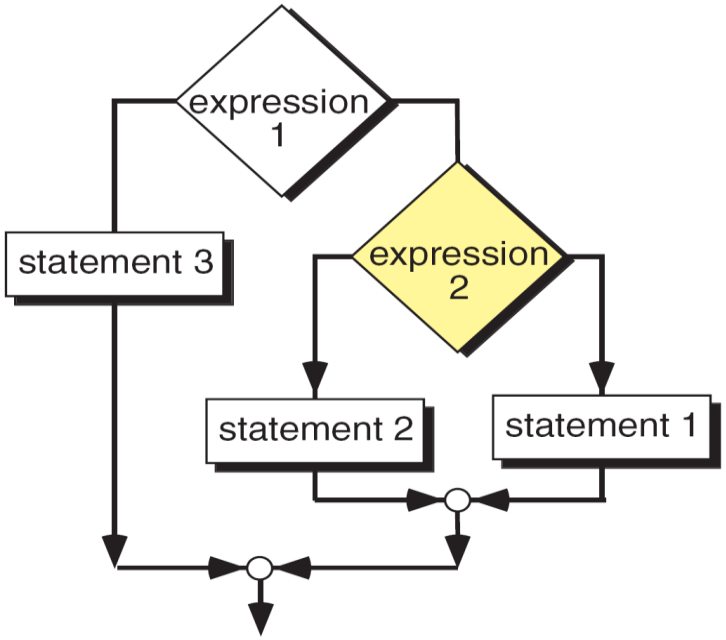
굳이 이런 축약된 조건식을 쓰지 않고 일반적인 if (cond) { } else { } 문장을 사용해도 무방함!!! 그냥 또 다른 다양한 C언어의 특화된 표현법!!

<결과>

```
Input your number: 1
positive number
kimjihwan@cspro:~/c_test_2$ ./test8
Input your number: -2
negative number
kimjihwan@cspro:~/c_test_2$ ./test8
Input your number: 0
zero
```

# Nested if statement (중첩 if 문)

- **Nested if Statements**
  - if 문 내부에 다른 if문이 중첩(nested)되어 나오는 문장
    - 조건 내에 다른 조건을 넣고 싶을 때 유용한 표현
  - 다음은 nested if문의 Logic 흐름과 실제 표기법임




(a) Logic flow

```
if (expression 1)
    if (expression 2)
        statement 1
    else
        statement 2
else
    statement 3
```

(b) Code

# Nested if statement (중첩 if 문)

- if...else 구문에서 여러 개의 if문이 중첩해서 나타날 때 else문이 생략될 경우 의미가 불분명해 질 수 있음
  - 컴퓨터는 if문에 가장 가까운 else를 하나의 쌍으로 처리
  - Dangling else problem 발생
    - [해결책] 의도한 if문을 compound statement { }로 처리

하나의 if..else쌍으로 인식(*)

```
if(alpha == 3)
        if(beta == 4)
            printf("alpha = 3, beta = 4\n");
else
        printf("alpha beta not valid\n");
```

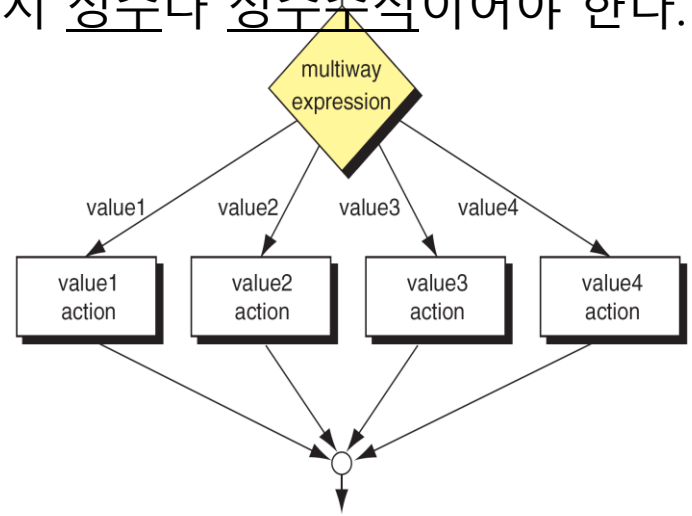
Compound statement { }

```
if(alpha == 3)
{
    if(beta == 4)
        printf("alpha = 3, beta = 4\n");
}
else
    printf("alpha is not 3\n");
```

(*) C언어는 들여쓰기(indentation)가 의미가 없음!!

# 다중선택(Multiway Selection)

- **Multiway selection에는 크게 switch 문과 else if 문이 존재한다.**
- **Switch문**
  - switch 문은 여러 개의 선택을 처리하는 구문
  - switch 이후의 괄호 ( ) 안의 표현식(expression)의 값 중에서 case의 값과 일치하는 것을 처리
    - Expression과 value는 반드시 정수나 정수수식이어야 한다.
- **switch 문의 logic flow**



# Switch statement

- **실행순서**
  - switch문의 조건식(expression)을 계산
  - 계산된 값과 일치하는 상수 값(constant)을 갖는 case를 위에서부터 찾음
  - 해당 case내부의 문장(statement) 실행
    - 문장이 여러 개면 블록 형태로 기술
  - case 내부에서 break statement을 만나면 switch문 종료
  - 일치된 case문을 만나지 못하여 default 실행

```
switch (expression)
  {
   case constant-1: statement
                    ⋮
                    statement
   case constant-2: statement
                    ⋮
                    statement
   case constant-n: statement
                    ⋮
                    statement
   default        : statement
                    ⋮
                    statement
  } // end switch
```

(optional)

**Default는 반드시 있어야 하는 것은 아니지만 switch문의 case들이 모든 가능한 상황을 다 커버하지 못할 경우 반드시 defaulf 조건문이 포함되어야 한다.**

# 스위치(switch)문

- **break문**
  - switch 문에서 break 문을 만나면 무조건 switch 문을 종료
  - 그러나 switch 문의 case 문 내부에 break 문이 없다면
    - 일치하는 case 문을 실행하고,
    - 계속해서 break 문을 만나기 전까지 무조건 다음 case 문 내부의 문장을 실행
  - break 문을 주로 사용하는 용도
    - Loop으로부터의 탈출
    - switch structure에서 나머지 실행문장을 건너 뛰기 위함

# 스위치(switch)문

- 예제 프로그램 – switch문의 예

```c
#include <stdio.h>

char scoreToGrade (int score);

int main(void)
{
    int  score;
    char grade;

    printf("Enter the test score (0-100); ");
    scanf("%d", &score);

    grade = scoreToGrade(score);
    printf("The grade is : %c\n", grade);

    return 0;
}
```

```c
char scoreToGrade(int score)
{
    char grade;
    int  temp;

    temp = score / 10;
    switch(temp)
    {
        case 10 :
        case 9  : grade='A'; break;
        case 8  : grade='B'; break;
        case 7  : grade='C'; break;
        case 6  : grade='D'; break;
        default : grade='F';
    }

    return grade;
}
```

<결과>

```
Enter the test score (0-100); 100
The grade is : A
kimjihwan@cspro:~/c test 2$ ./test5
Enter the test score (0-100); 80
The grade is : B
```

# else if문

- **다중판단을 할 경우에 사용 되는 형태**

  **(if…else if…else)**

  - if의 expression을 검사하여 참일 경우
    - 바로 아래의 statement1 실행
  - expression이 거짓일 경우
    - 다음의 else if 로 넘어가서 조건 검사
  - 마지막 else는 위의 조건들이 모두 참이 아닐 경우에 statement4 실행
  - 맞는 조건을 찾은 다음에는 나머지 조건검사 수행 하지 않음

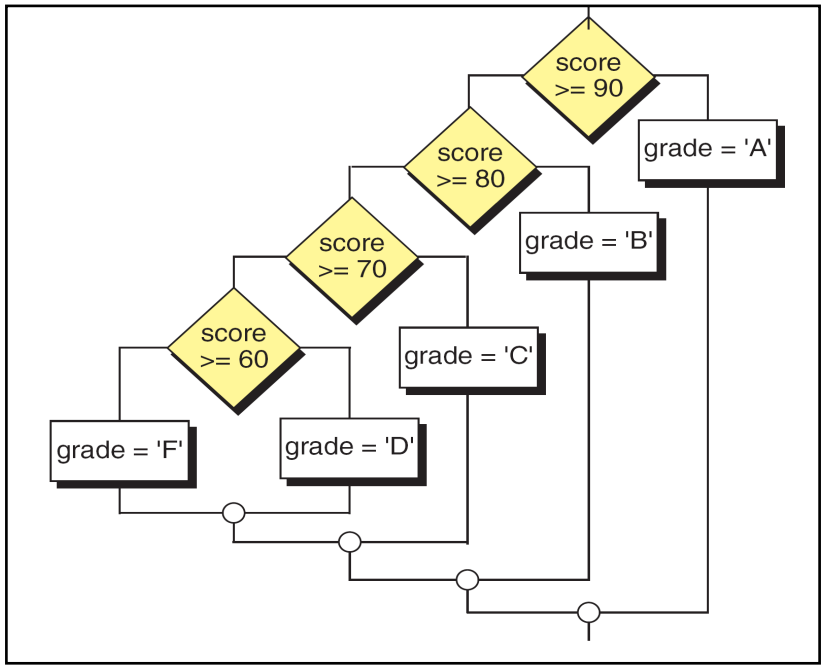
```
if(expression)
        statement1
else if(expression)
        statement2
else if(expression)
        statement3
else
        statement4
```

# else if statement

## ◆ 예제 프로그램 – else if 문의 예

```
#include <stdio.h>

int main(void) {
      int  score = 0;
      char grade;

      printf("Enter the test score(0-100); ");
      scanf("%d", &score);

      if(score >= 90)
            grade='A';
      else if(score >= 80)
            grade='B';
      else if(score >= 70)
            grade='C';
      else if(score >= 60)
            grade='D';
      else   grade='F';

      printf("The grade is %c\n", grade);
      return 0;
}
```

```
Enter the test score (0-100); 100
The grade is : A
kimjihwan@cspro:~/c test 2$ ./test5
Enter the test score (0-100); 80
The grade is : B
```



# switch vs. else if statement

- **switch문과 else if문 모두 조건에 따라 프로그램의 흐름을 분기시키는 목적으로 사용** (다중선택)
- **switch문에는 비교연산이 올 수 없음** (정수형 수식)
- **switch문으로 구현된 내용은 else if문으로 구현 가능**
  - 반대의 경우는 구현이 안되는 경우가 생김
    - 예 : 비교연산
- **분기의 횟수가 많을 수록 switch문이 else if문에 비해 간결한 코드 구현이 가능**

# 예제: Nested if statement

- 예제 프로그램 – Nested if statement (중첩 if문)의 예

```
#include <stdio.h>

int main(void)
{
  int a;
  int b;

  printf("Please enter two integers: ");
  scanf("%d %d",&a, &b);

  if(a  <=  b)
       if(a    <   b)
           printf("%d  <   %d\n",a, b);
       else
           printf("%d  ==  %d\n",a, b);
  else
       printf("%d  >   %d\n",a,b);
  return 0;

}
```

<결과>

```
Please enter two integers: 1 4
1       <       4
kimjihwan@cspro:~/c test 2$ ./test4
Please enter two integers: 1 1
1       ==      1
kimjihwan@cspro:~/c test 2$ ./test4
Please enter two integers: 4 1
4       >       1
```

# 예제

- 세 개의 점수를 입력 받아 평균을 구한 후 평균이 90이상이면 “A”, 80 이상이면 “B”, 70 이상이면 “C”, 60 이상이면 “D”, 그렇지 않으면 “F”를 출력
- 한 과목이라도  40점 미만이면 과락, 평균이 60점 이하면 탈락

<결과>

```
Input korean score :100
Input English score :100
Input mathematics score :100
Your average is 100.
grade A
 your test pass
kimjihwan@cspro:~/c_test_2$ ./test9
Input korean score :40
Input English score :50
Input mathematics score :40
Your average is 43.
grade F
 your test fail
kimjihwan@cspro:~/c_test_2$ ./test9
Input korean score :30
Input English score :100
Input mathematics score :100
Your average is 76.
grade C
 your test fail
```

- 정답은 다음 쪽에...

# 예제소스

```
#include <stdio.h>

int main(void){
  int kor, eng, mat, ave;

  printf("Input korean score :");
  scanf("%d", &kor);
  printf("Input English score :");
  scanf("%d", &eng);
  printf("Input mathematics score :");
  scanf("%d", &mat);
  ave=(kor+eng+mat)/3;
  printf("Your average is %d.\n", ave);

  switch(ave/10){
   case 10:
   case 9:
     printf("grade A");
     break;
   case 8:
     printf("grade B");
     break;
   case 7:
     printf("grade C");
     break;
   case 6:
     printf("grade D");
     break;
   default:
     printf("grade F");
  }
```

```
  if(ave>=60)
    if(kor>=40 && eng>=40 && mat>=40)
      printf("\n your test pass \n");
    else
      printf("\n your test fail \n");
  else
      printf("\n your test fail \n");

  return 0;

}
```