



Chapter 03. Control statements

Index

1. Understanding the control statements
2. Select statement
3. Loop
4. Auxiliary control statements
5. Endless loop

Objective

- The learning control statements for changing the program flow.
- Depending on the conditions, if the loop to selectively perform only one sentence, if else statements, and learning how to use a switch statement.
- for loop, while loop to repeatedly perform a specific sentence, learn the do ~ while control use.
- how to use the break statement in the auxiliary control statements

01 Understanding the control statements

[Table 3-1] Type of control statements

construction	Control Command
Select statement	If statement
	if~else statement
	Multi if~else statement
	Switch statement
Loop	For statement
	While statement
	do~while statement
Auxiliary control statements	Break statement
	Continue statement

02 Select statement

- Select statement is used when you want to determine the sentence in accordance with the conditions. Select statement if there is a loop if ~ else statement, multi-if ~ else statement, switch statement.

■ If statement

- if statement is a control statement to cause the specific sentence only to satisfy a given condition. If the result of the condition is true, do the sentence immediately following the if statement, and the results can be performed immediately without performing the following sentence: This sentence is false.

[sentence]

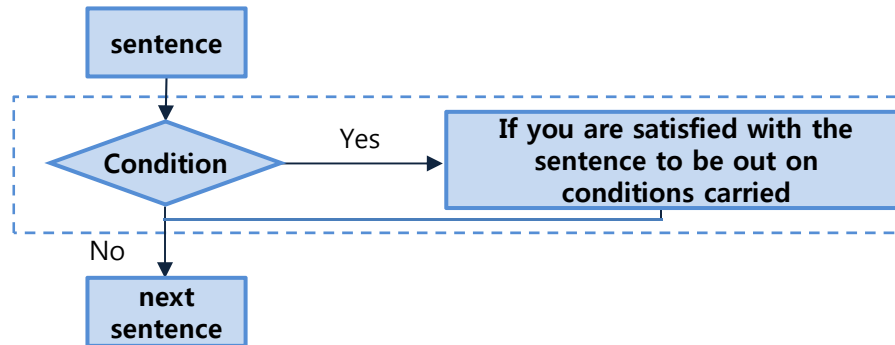
if(Condition){

 If you are satisfied with the condition statement
 it is performed;}

[next sentence]

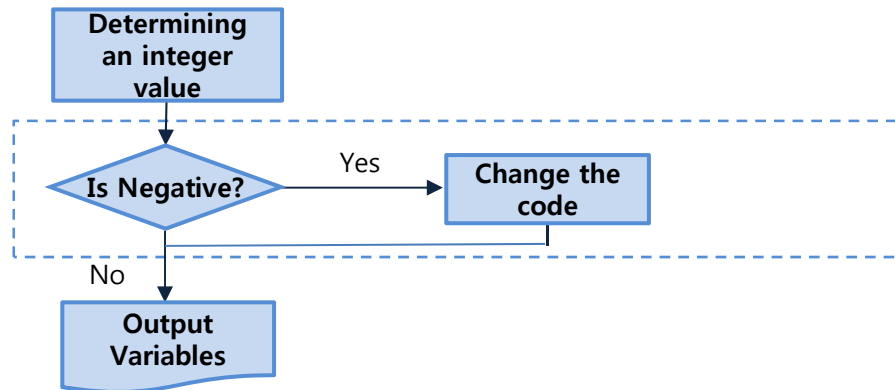
if문 기본 형식

02 Select statement



[Picture 3-1] if loop flowchart

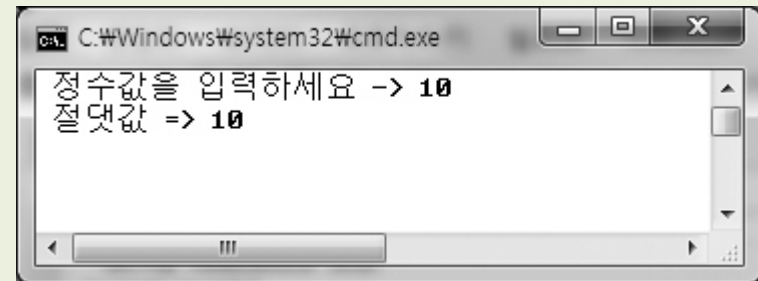
- Use the if loop statement to obtain the absolute value as an example, let's write a program.



[Picture 3-2] A flow chart of a program to obtain the absolute value

Example 3-1. Use the if statement to obtain the absolute value (03_01.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     int x;
06
07     cout << " 정수값을 입력하세요 -> ";
08     cin >> x;
09
10     if(x < 0) // 음수일 경우에만
11         x = -x; // 부호 변경
12
13     cout << " 절댓값 => " << x << "\n";
14 }
```



02 Select statement

■ if-else statement

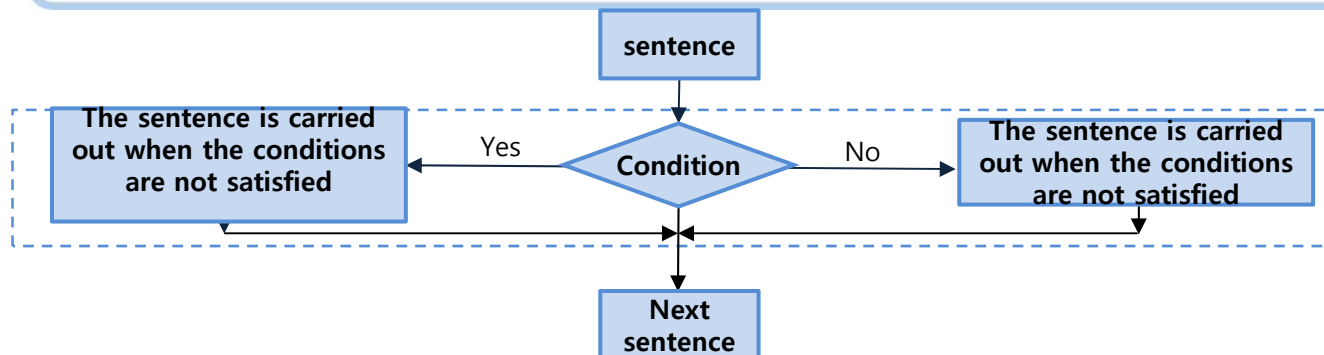
- An if-else loop statement format used by most of the series, and it is used to select only one of two cases.

```
[sentence]  
if(Condition){
```

```
    The sentence is carried out when the conditions are  
    satisfied;  
}
```

```
else{  
    The sentence is carried out when the conditions are not  
    satisfied;  
}  
[next sentence]
```

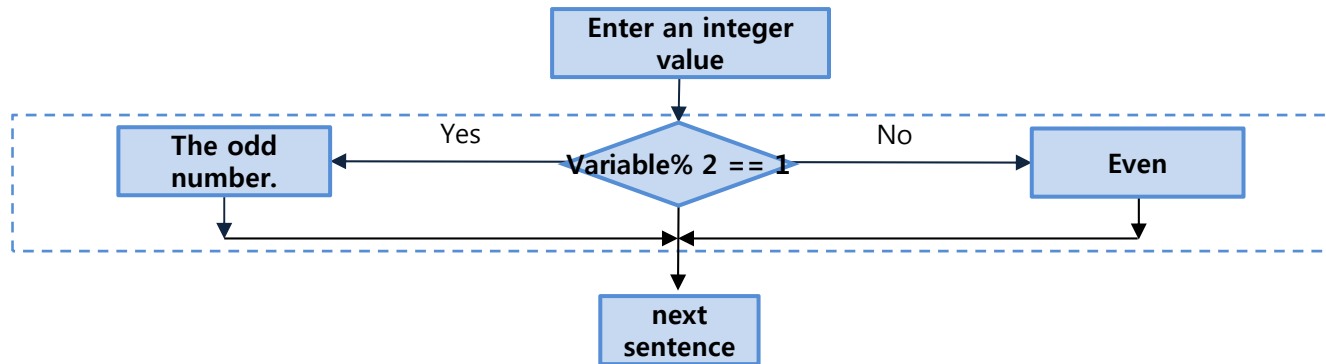
if~else문 기본 형식



[Picture 3-1] if~else loop flowchart

02 Select statement

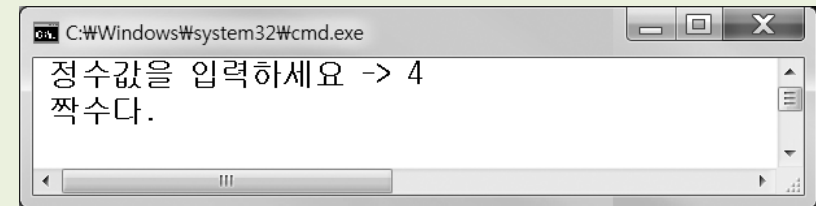
- By using if ~ else statement we will write a program to determine whether a given integer data is even or odd.



[Picture 3-2] A flow chart of a program to obtain the absolute value

Example 3-2. Even-odd determination to use the if ~ else Statement (03_02.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     int x;
06
07     cout << " 정수값을 입력하세요 -> ";
08     cin >> x;
09
10     if(x % 2 == 1) // 2로 나누어 나머지가 1이면
11         cout << " 홀수다. \n";
12     else
13         cout << " 짝수다. \n";
14 }
```



02 Select statement

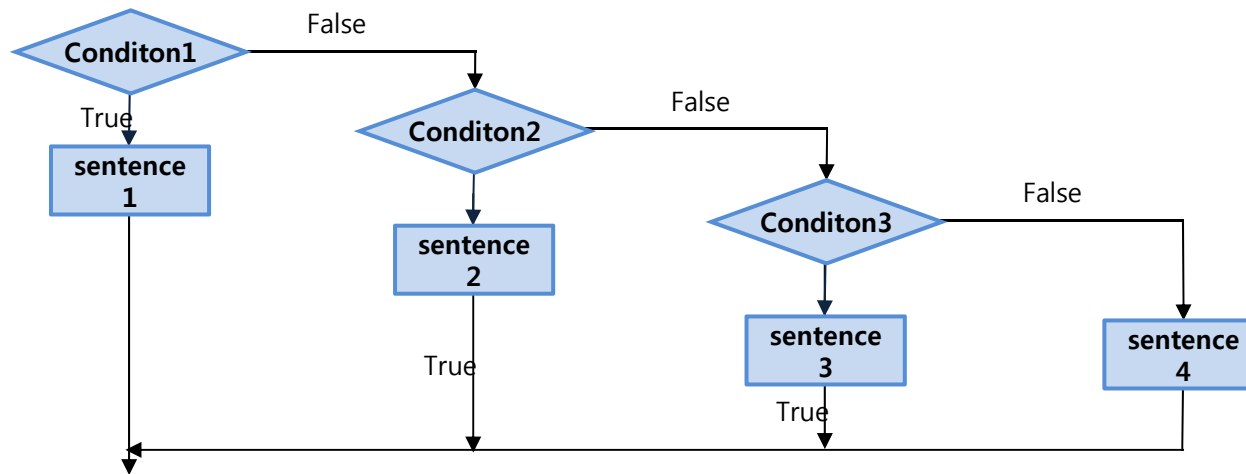
■ Multiple if-else statements

- if ~ else statement is true, but only once in the process of selecting a false, if the number of cases that you have to select from more than three non-overlapping two shall be used by the if ~ else statement.

다중 if~else문 기본 형식

```
[sentence]
if(Condition 1){
    Sentence to be processed when the conditions are satisfied. 1;
}
else if(Condition 2){
    1 sentence does not satisfy the conditions to be processed
    when the conditions are satisfied 2;}
...
else if(Condition n){
    Condition 1 does not satisfy the conditions n-1 from the sentence to be
    processed when the n conditions are satisfied;}
else{
    Sentence to be processed when they are not satisfied with all the
    conditions mentioned above;}
[next sentence]
```

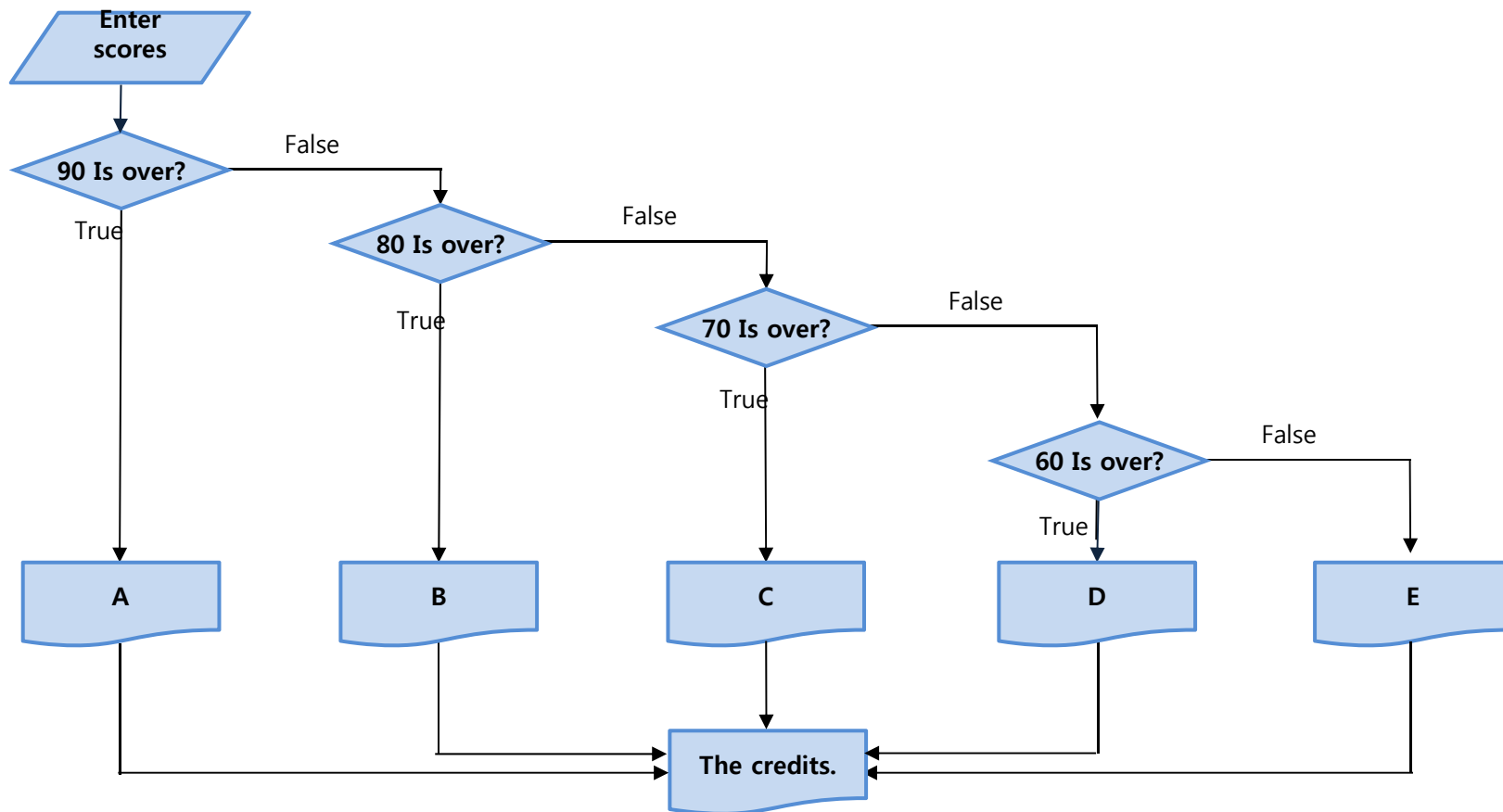
02 Select statement



[Picture 3-5] Multiple flowchart if ~ else Statement

02 Select statement

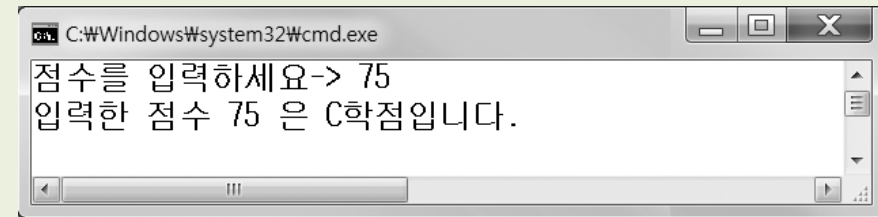
- Let receiving a score using multiple statements if ~ else fill out to obtain the credit program.



[Picture 3-6] To obtain the credit program Flowchart

Example 3-3. To take advantage of multiple calculations by credit if ~ else Statement (03_03.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     int score; // 입력받은 점수를 저장할 변수
06     char grade; // 구한 학점을 저장할 변수
07     cout<<"점수를 입력하세요 -> ";
08     cin>>score;
09     // 조건 검사
10     if(score>=90) // score가 90이상이나?
11         grade='A'; // 만족하면 grade='A'
12     else if (score>=80) // 아니면 score가 80이상이나?
13         grade='B'; // 만족하면 grade='B'
14     else if (score>=70) // 아니면 score가 70이상이나?
15         grade='C'; // 만족하면 grade='C'
16     else if (score>=60) // 아니면 score가 60이상이나?
17         grade='D'; // 만족하면 grade='D'
18     else // 아니면
19         grade='F'; // grade='F'
20     cout<<"입력한 점수 " <<score<<" 은 "<<grade<<"학점입니다.\n";
21 }
```



02 Select statement

■ Switch statement

- A switch statement for purposes such as multiple if-else statements, while a switch statement to perform multiple if-else statement is the result of true or false statements to the behavior of branches depending on a constant expression.

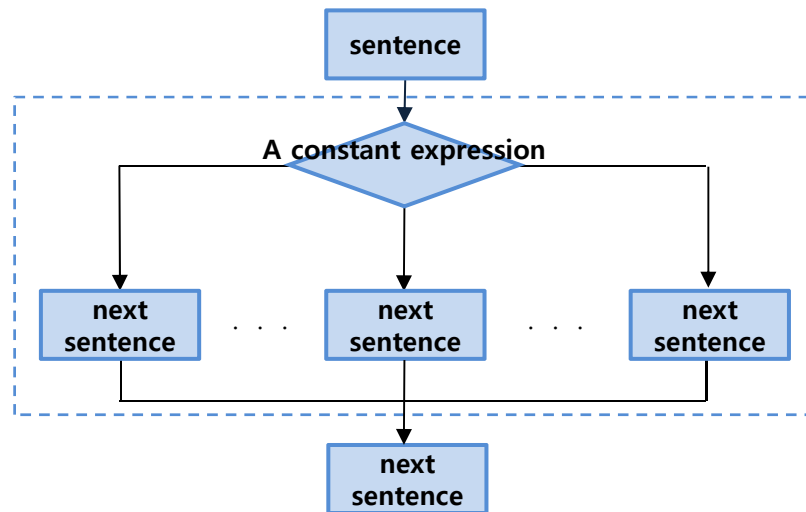
```

sentence 1;
Switch(A constant expression) {
    case Integer 1 : sentence 2; break;
    case Integer 2 : sentence 3; break;
    ...
    case Integer n : sentence n; break;
    ...
    default: sentence m;
}
next sentence;
```

switch문 기본 형식

- The Switch statement can omit the break statement. The case statement without a break statement Because it is using this a similar function as OR operator (||) in an if statement can make the program even more clear.

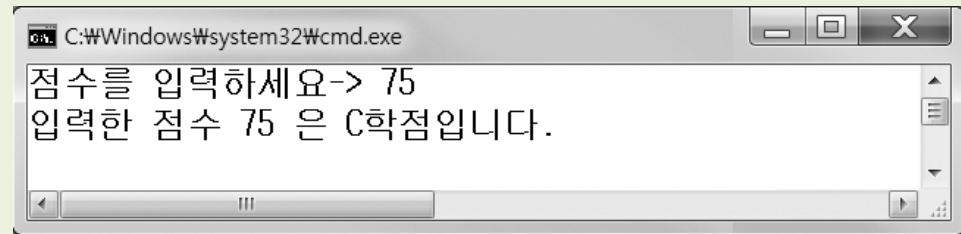
02 Select statement



[Picture 3-7] The switch statement Flowchart

Example 3-5. Credits to determine a switch statement (03_05.cpp)

```
01 #include<iostream>
02 using namespace std;
03 void main()
04 {
05     int score;    // 입력받은 점수를 저장할 변수
06     char grade; // 구한 학점을 저장할 변수
07     cout<<"점수를 입력하세요 -> ";
08     cin>>score;
09
10     switch(score/10) {    // 결과가 정수로 나오는 산술식
11     case 10 : grade='A'; break;
12     case 9  : grade='A'; break;
13     case 8  : grade='B'; break;
14     case 7  : grade='C'; break;
15     case 6  : grade='D'; break;
16     default : grade='F';
17     }
18     cout<<"입력한 점수 " <<score<<" 은 "<<grade<<"학점입니다.\n";
19 }
```



03 Loop

- It refers to a loop statement that is performed repeatedly to a particular part of the sentence. The for loop statement, while statement, do ~ while statements.

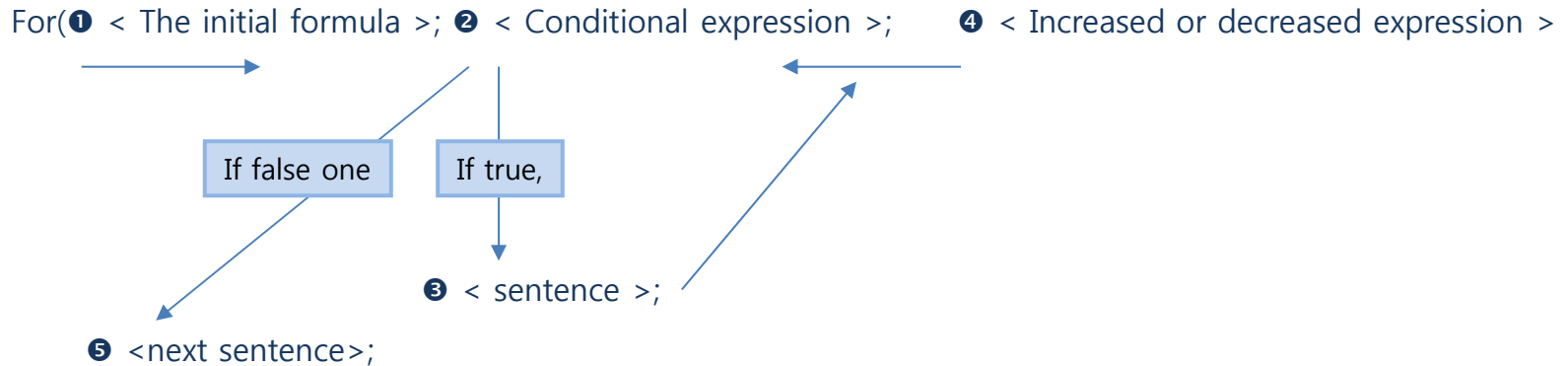
■ For statement

- The statement repeated for loop a specified number of times is composed as <initial expression>, <condition>, <or decrease expression>, <text> as follows:

```
for(< The initial formula >;< Conditional  
expression >;< Increased or decreased expression >) {  
    sentence 1;  
}
```

for문 기본 형식

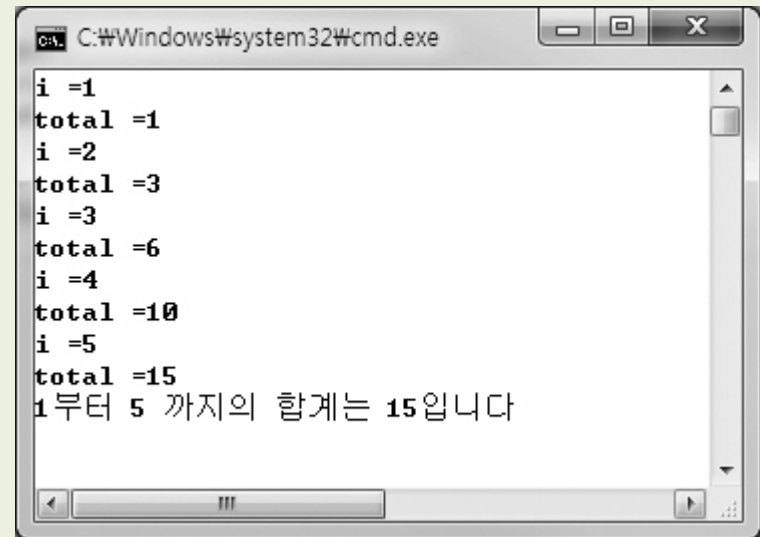
03 Loop



- ① First, initialize the control variables in the <initial expression>
- ② Checks whether a condition is true or false in the <condition>.
- ③ And the <condition> is true, do the <sentence> in blocks,
- ④ <Increased or decreased expression> increases or decreases the controlled variable.
- ② Check whether the <or decrease expression> in <condition> is true, the value is changed by the applied.
- ⑤ If <condition> is false, applying the values of the control parameters for exiting the loop without performing the <sentence> This <next sentence> is performed.

Example 3-7. Obtaining the sum of from 1 to 5 using a for statement (03_07.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     int total=0; // 반드시 초기화해야 한다.
06     int i;
07     for(i=1;i<=5;i++) {
08         cout<<"i ="<<i<<endl;
09         total+=i; // total=total+i;
10         cout<<"total ="<<total<<endl;
11     }
12     cout<<"1부터 " << i-1 <<" 까지의 합계는 "
13     << total <<"입니다"<<endl;
14 }
```



```
C:\Windows\system32\cmd.exe
i =1
total =1
i =2
total =3
i =3
total =6
i =4
total =10
i =5
total =15
1부터 5 까지의 합계는 15입니다
```

03 Loop

■ For multi-statement

- Including the use of the for statement in the for statement called for a multi-statement.

```
int out, in;

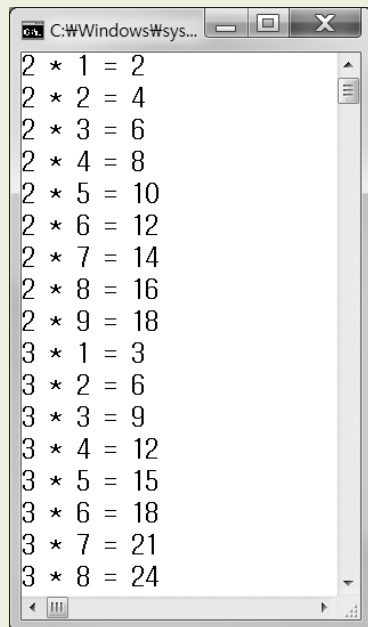
cout<< "out:시침 \t >> \t in:분침";
cout<< "바깥쪽 제어변수 \t >> \t 안쪽 제어변수";

for (out = 1; out <= 3; out++){ ----- ❶
    // 문장1;
    for (in = 1; in <= 5; in++){ ----- ❷
        cout<< out << "\t\t >> \t " << in; // 문장2;
    }
    // 문장3;
}
```

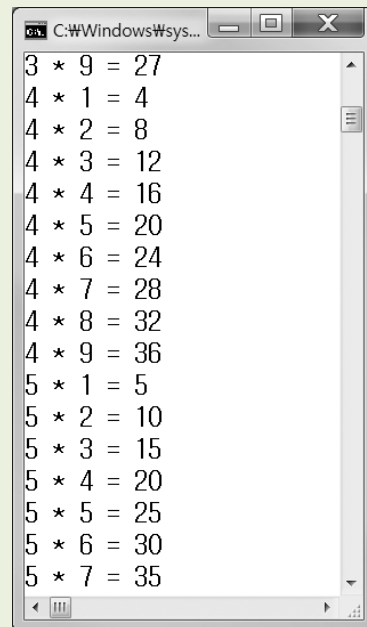
- The sentence 1 and sentence 3 is repeated as much as the number of iterations of a for statement (❶) outside. But two sentences are repeated as many times as (the number of iterations for the outer loop) * (inside contact for repetitions).

Example 3-10. Use it to output multiple times tables for specific statements (03_10.cpp)

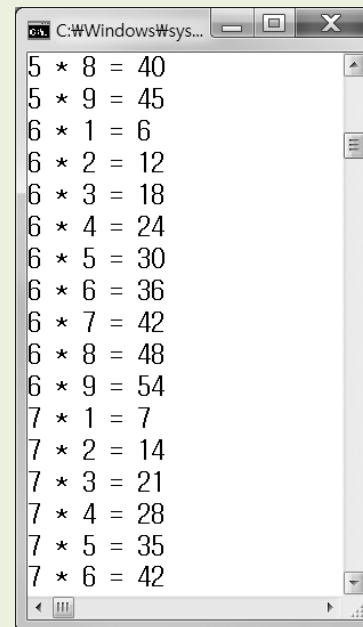
```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     for(int dan=2; dan<=9; dan++) // 바깥 for문
06         for(int j=1; j<10; j++) // 안쪽 for문
07             cout << dan << " * " << j << " = " << dan*j << "\n";
08 }
```



```
C:\Windows\sys...
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
```



```
C:\Windows\sys...
3 * 9 = 27
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
```



```
C:\Windows\sys...
5 * 8 = 40
5 * 9 = 45
6 * 1 = 6
6 * 2 = 12
6 * 3 = 18
6 * 4 = 24
6 * 5 = 30
6 * 6 = 36
6 * 7 = 42
6 * 8 = 48
6 * 9 = 54
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
```

...

03 Loop

■ While statement

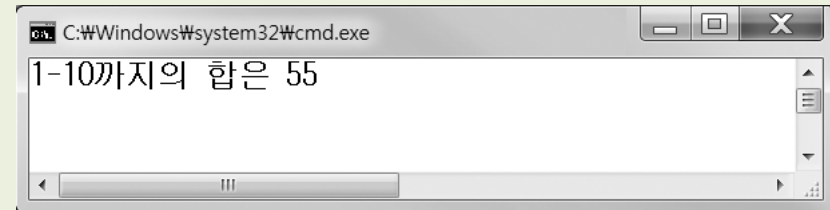
- while loop repeatedly performs the statement while a condition is met.
- Compared loops for <initial expression> and the <increase or decrease expression> is not only the form <condition>.

```
i = 1; .....❶  
while (i <= 10) {...❷  
    total += i;.....❸  
    i++; .....❹  
}
```

- If you are satisfied with the condition (❷) statement performs a (❸, ❹), and correct the harm check back condition (❷) after executing the sentence conditions and repeat the process for performing sentences (❸, ❹), and if the condition is false without carrying out the sentence comes out of the while statement.

Example 3-11. Obtaining the sum of from 1 to 10 using a while loop (03_11.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     int total=0;
06     int i=1;        // for문의 초기식에 해당
07     while(i<=10) { // for문의 조건식에 해당
08         total+=i;
09         i++;        // for문의 증감식에 해당
10     }
11     cout<<"1-10까지의 합은 "<<total<<"\n";
12 }
```



03 Loop

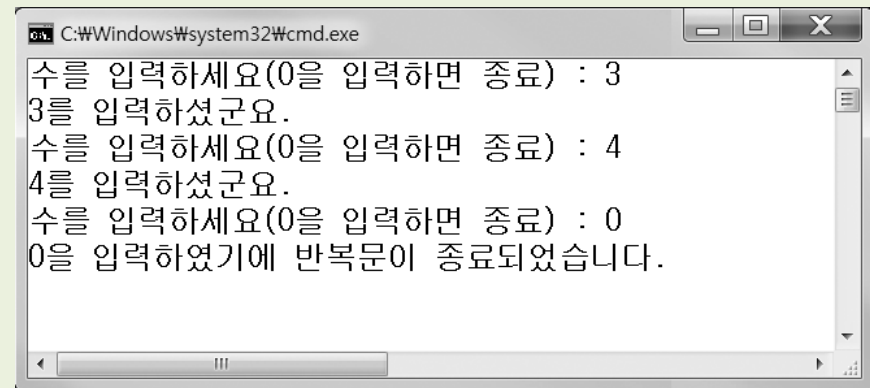
■ do~while statement

- do ~ while statement is similar with while statement but differs in that the specifier has been added that do while statement. In other words, one run once and then repeated the sentence over and over again if you examine the condition. If the condition is true and the false, exit the while statement.

do ~ while the default format statement	For use while statements do ~
<pre>do { sentence } while(Conditional expression);</pre>	<pre>do{ cout<<"수를 입력하세요(0을 입력하면 종료) : "; cin>>num; cout<< num <<"를 입력하셨습니다.Wn"; } while(num!=0);</pre>

Example 3-13. do ~ to create a little more concise program while the door (03_13.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     int num;
06     do{
07         cout<<"수를 입력하세요(0을 입력하면 종료) : ";
08         cin>>num;
09         cout<< num <<"를 입력하셨습니다.\n";
10     } while(num!=0);
11     cout<< num << "을 입력하였기에 반복문이 종료되었습니다.\n";
12 }
```

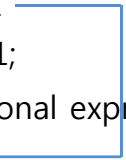


04 Auxiliary control statements

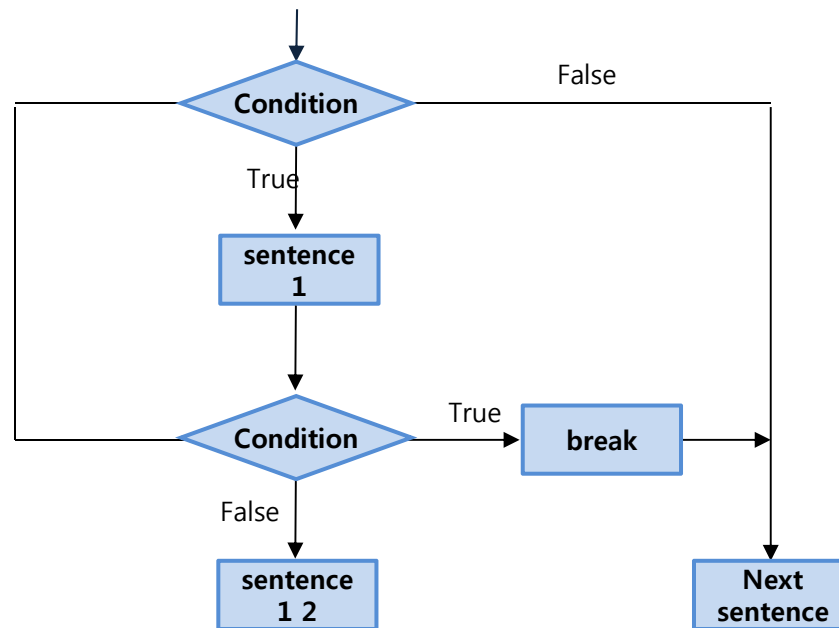
- The auxiliary control statement break statement continue statement can be used in combination with a loop to control the flow of a program in more detail.

■ Break statement

- The break statement can be used to skip without taking part in the program in order to escape a switch statement, for statement, while statement, statement do ~ while control.

The break statement default format	The break statement Example
<pre>for(The initial expression; Condition; Increased or decreased expression) { sentence1; if(Conditional expression) break; sentence2; } sentence3;</pre> 	<pre>for(i=1; i<=10; i++) { if(i%2==0) // i가 2로 나누어서 떨어지면 break // for문을 벗어남 total+=i; }</pre>

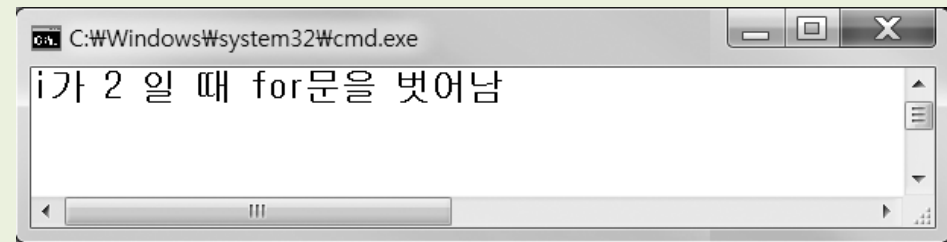
04 Auxiliary control statements



[Picture 3-8] break control flowchart

Example 3-14. statement for escape in the middle repeatedly (03_14.cpp)

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     int total=0;
06     int i;
07     for(i=1; i<=10; i++) {
08         if(i%2==0) // I가 2로 나누어서 떨어지면
09             break // for문을 벗어남
10         total+=i;
11     }
12     cout<<"i가" << i <<" 일 때 for문을 벗어남\n"
13 }
```



05 Endless loop

- An infinite loop endlessly refers to a state that is carried out without the execution of the program be terminated.

■ Using the for endless loop

- The following is the simplest form of the for statement. for statements with a semicolon (;) only two times when technology is no grammatical problems. But because it did not describe the conditions out of the loop is infinite loop.

```
for( ; ; ) {  
    ...  
    ...  
}
```

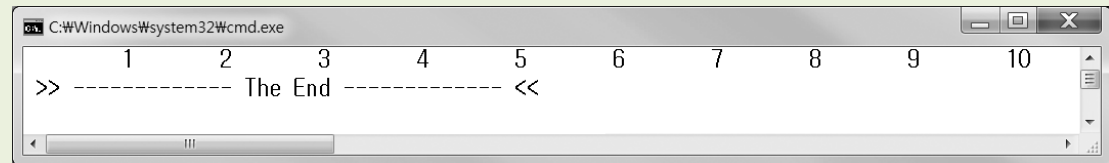
■ Infinite loop using the while statement

- while if the result of the conditional statement is true, continue to repeat the lies and beyond if the loop. An endless loop by the while statement is true for the technique generally represented by the conditional expression.

```
while( true ) {  
    ...  
    ...  
}
```

Example 3-17. The break statement (03_17.cpp) to escape from the endless loop

```
01 #include <iostream>
02 using namespace std;
03 void main()
04 {
05     int i=0;
06     for( ; ; ){
07         cout << "Wt" << ++i;
08         if(i%10==0)
09             break;          // 무한루프 탈출
10     }
11     cout << "Wn >> ----- The End ----- << Wn";
12 }
```



Homework

- Chapter 3 Exercise: 4, 5, 9, 10, 12, 14, 15, 17, 18, 21, 24, 27, 29