**6장 Looping**

■ Keywords: for, while, do while

■ Operators:

< > >=

<= != == +=

\*= -= /= %=

■ Functions:

fabs()

■ C의 3가지 loop structures— while , for , do while

■ relational operators를 사용하여 loop를 제어하는 expressions 작성

■ Arrays, loop 만큼 자주 사용됨

■ return values을 주는 function 작성

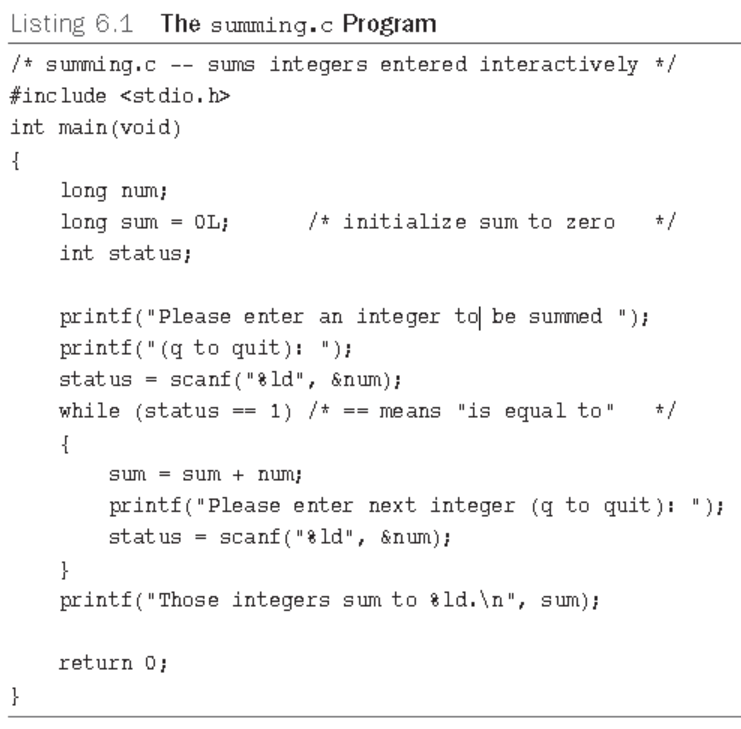
# 3가지 종류의 program flow:

■ a sequence of statements의 실행

■ 어떤 조건이 만족될 때까지 a sequence of statements을 반복 (looping)

■ alternative sequences 사이에 분기를 결정하는 조건문(branching)

**6.1 while Loop 복습**



**6.1.1 Program Comments**

while문의 조건: status == 1

* == operator는 *equality operator*

*initialize sum to 0*

*prompt user*

*read input*

*while the input is an integer,*

*add the input to sum,*

*prompt user,*

*then read next input*

*after input completes, print sum*

* *pseudocode*라고 함 – 가장 대표적인 예제 프로그램 모형으로 학생들이 작성능력을 가져야 함
* 컴퓨터 언어와 유사한 형태로 프로그램을 영어 문장으로 표현하는 것

**6.1.2 C-Style Reading Loop**

status = scanf("%ld", &num);

while (status == 1)

{

/\* loop actions \*/

status = scanf("%ld", &num);

}

while (scanf("%ld", &num) == 1)

{

/\* loop actions \*/

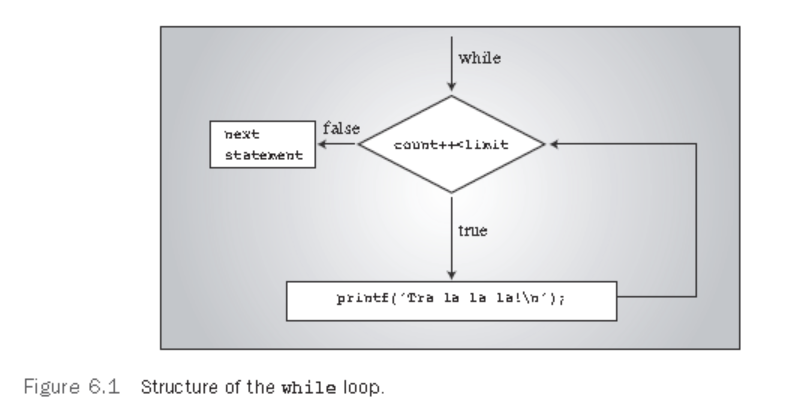
}

**6.2 while Statement**

while ( *expression* )

*statement*

* 각cycle을 *iteration* 이라 한다



**6.2.1 while Loop의 종료**

index = 1;

while (index < 5)

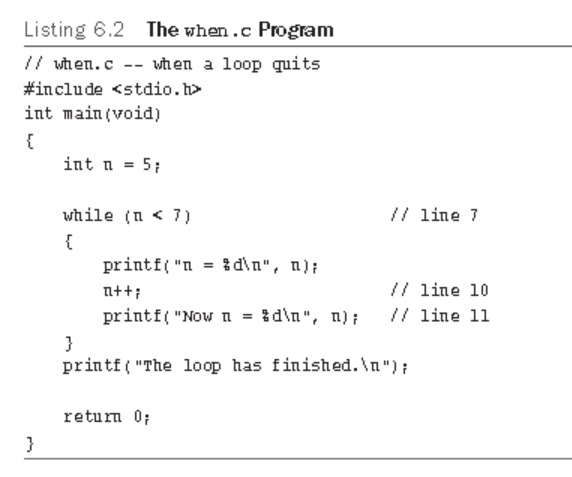
printf("Good morning!\n");

index = 1;

while (--index < 5)

printf("Good morning!\n");

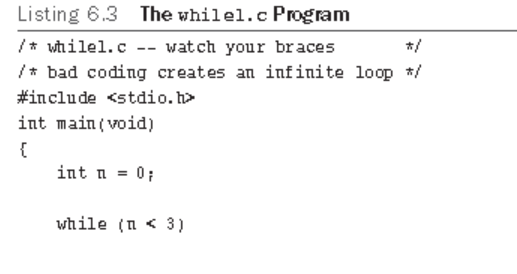
**6.2.2 언제 Loop가 종료되나?**

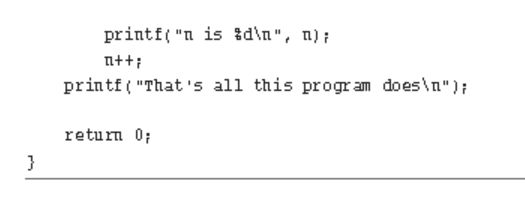


**6.2.3 while : Entry-Condition Loop**

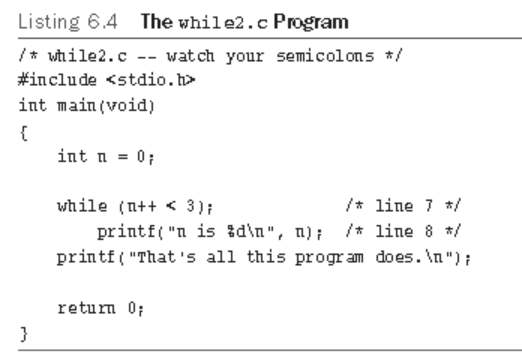
* while loop는 entry condition을 사용하는 *conditional* loop라 한다

**6.2.4 Syntax Points**





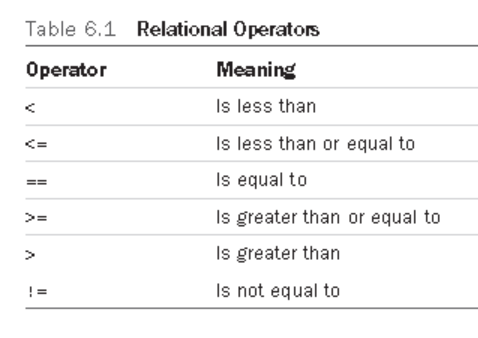
* example of an *infinite loop*



* test condition 다음에 *null statement* 실행 -> 무한 loop,
* while (scanf("%d", &num) == 1)
* ; /\* skip integer input \*/

**6.3 크기 비교: Relational Operators와 Expressions 사용**

* *relational expressions, relational operators* .



while (number < 6)

{

printf("Your number is too small.\n");

scanf("%d", &number);

}

while (ch != '$')

{

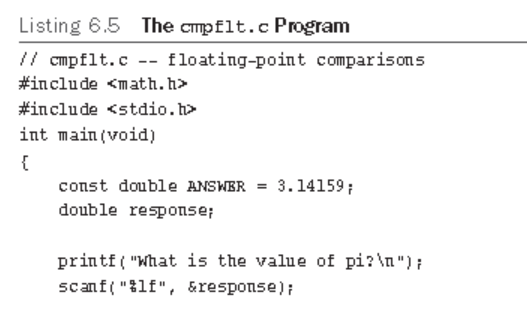
count++;

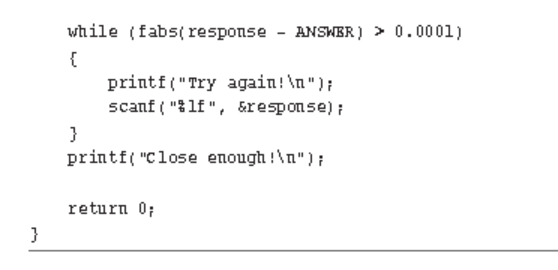
scanf("%c", &ch);

}

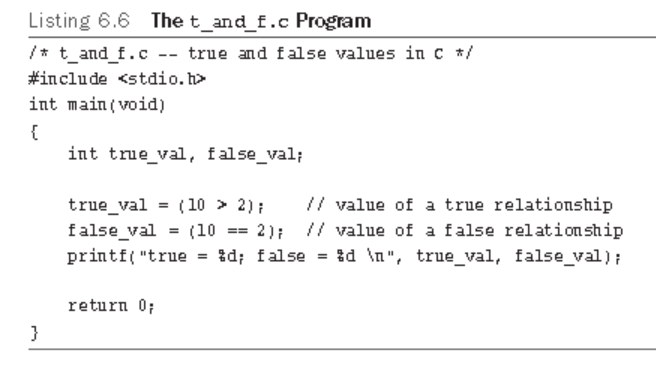
while (scanf("%f", &num) == 1)

sum = sum + num;

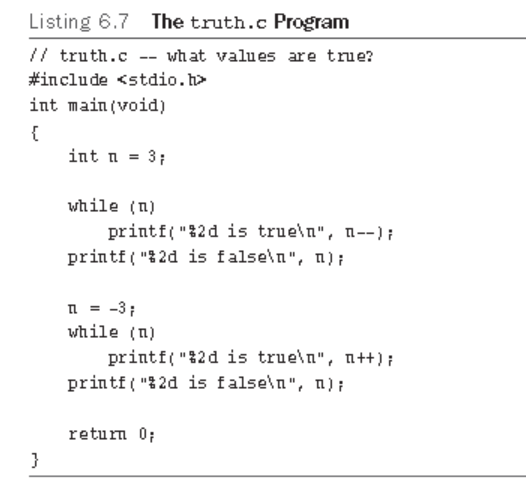




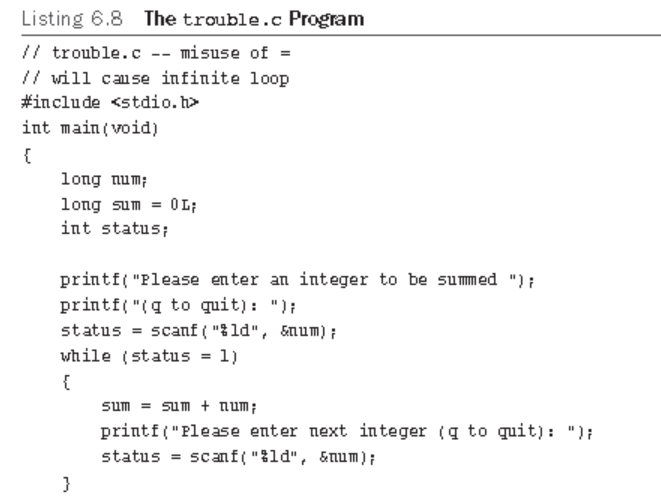
**6.3.1 어느 것이 Truth인가?**



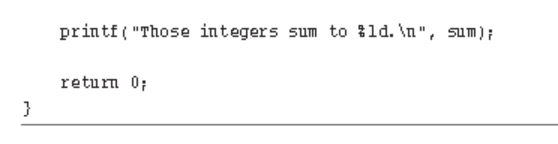
**6.3.2 while 조건식의 True?**

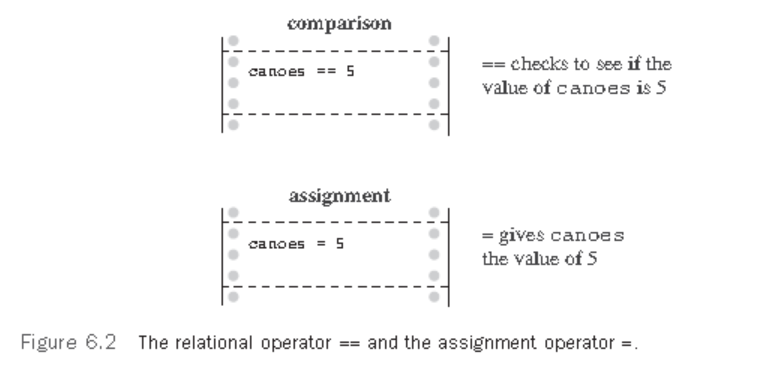


**6.3.3 while 조건식의 Truth**



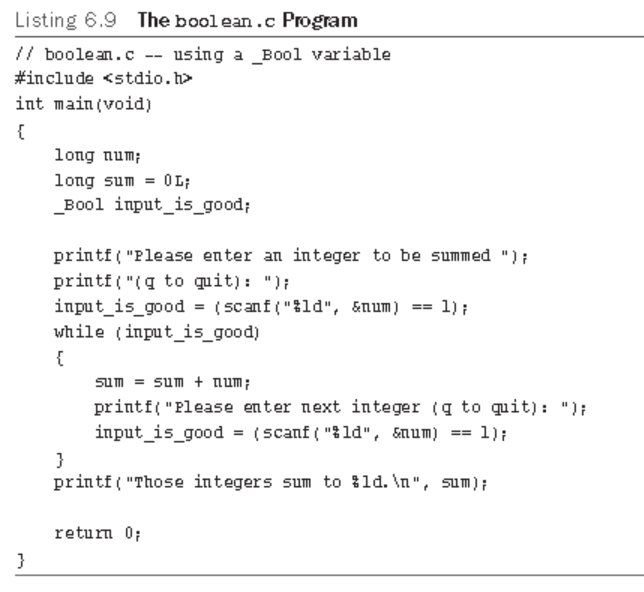
* while (status == 1)





**6.3.4 새로운 Bool Type 사용**

* C99는 Bool type을 사용. The type is named after George Boole,



* C++는 bool , true , false를 keywords로 사용.

**6.3.5 Precedence(우선순위) of Relational Operators**

x > y + 2

은 다음과 같다: x > (y + 2)

x = y > 2

은 다음과 같다: x = (y > 2)

x\_bigger = x > y;

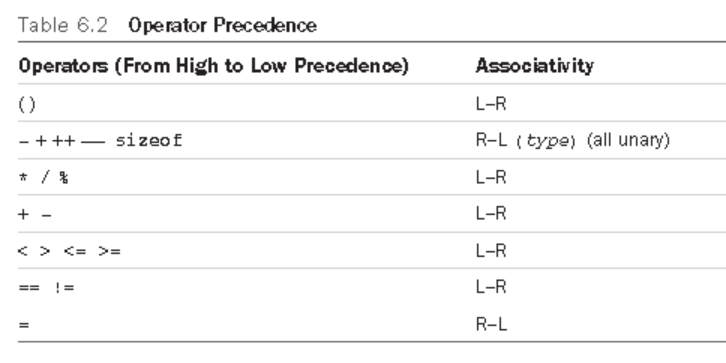
은 다음과 같다: x\_bigger = (x > y);

높은 precedence group: < <= > >=

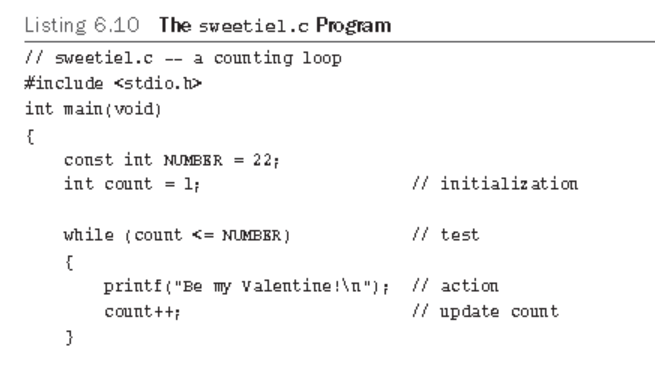
낮은 precedence group: == !=

ex != wye == zee

은 다음과 같다: (ex != wye) == zee



**6.4 무한 Loops와 Counting Loops**



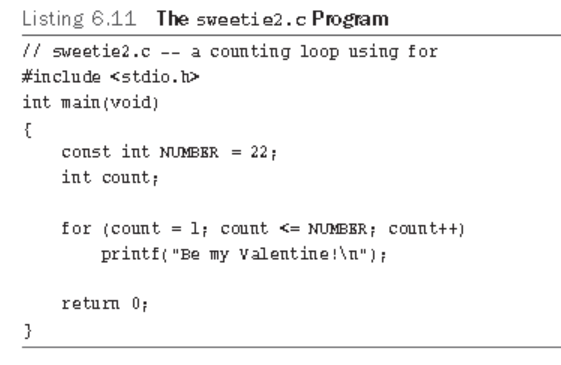


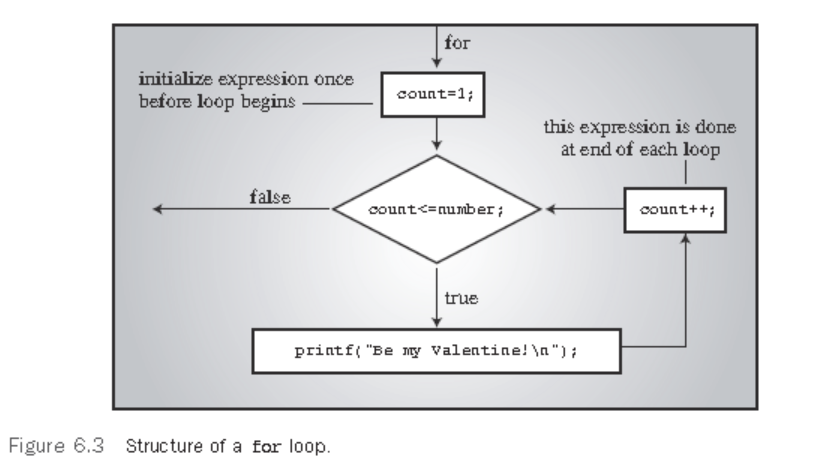
1. counter는 초기화됨.

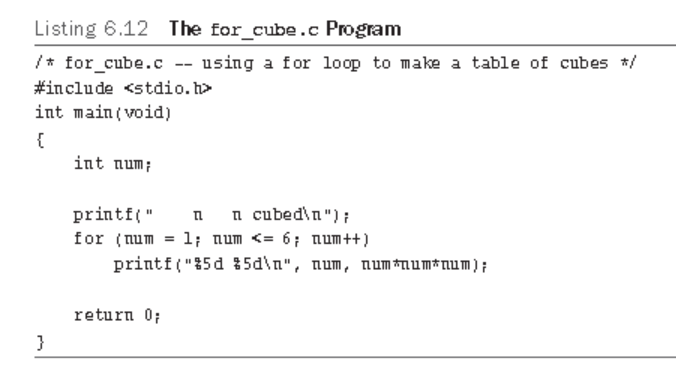
2. counter는 limiting value와 비교됨.

3. counter는 loop가 반복될 때마다 증가됨.

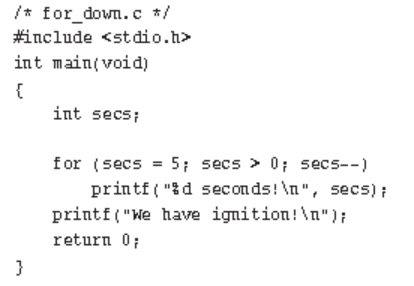
**6.5 for Loop**

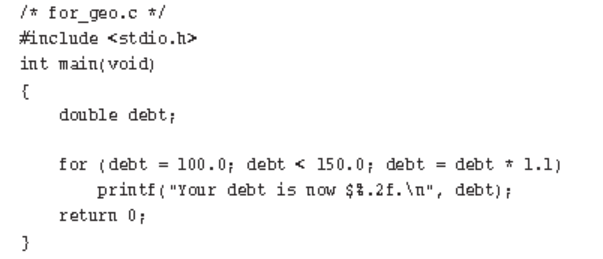






**6.5.1 for 문의 편리한 사용: flexibility**





**6.6 간편한 Assignment Operators: += , -= , \*= , /= , %=**

scores += 20 은 다음과 같다: scores = scores + 20 .

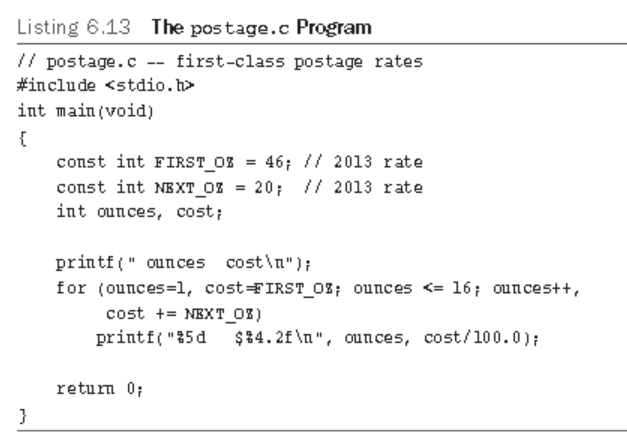
dimes -= 2 은 다음과 같다: dimes = dimes - 2 .

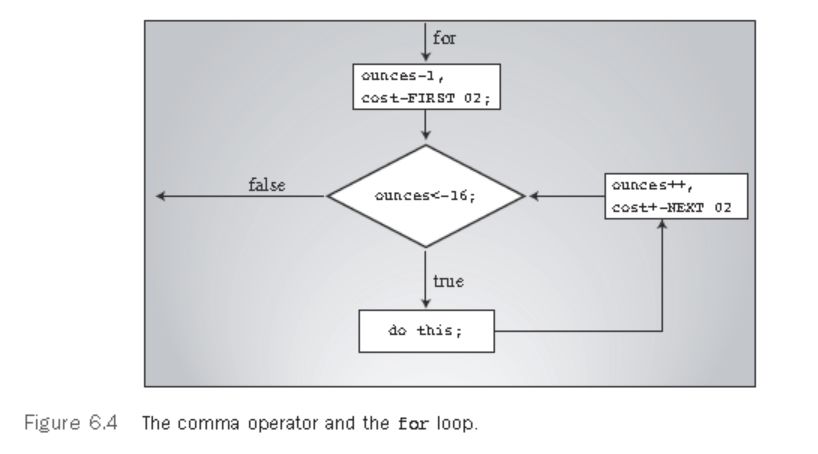
bunnies \*= 2 은 다음과 같다: bunnies = bunnies \* 2 .

time /= 2.73 은 다음과 같다: time = time / 2.73 .

reduce %= 3 은 다음과 같다: reduce = reduce % 3 .

**6.7 Comma Operator**





* comma는 sequence point이다

ounces++, cost = ounces \* FIRST\_OZ

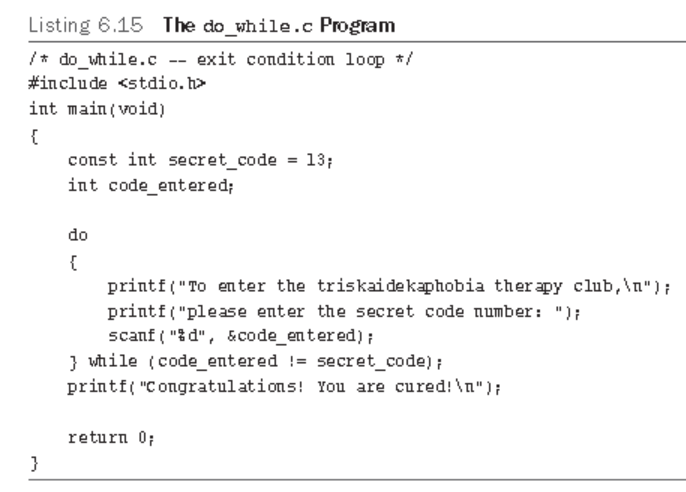
* comma는 a sequence point이다: 좌측 expression의 side effects가 먼저 일어남, 다음에 우측 expression이 실행됨
* comma expression의 값은 우측 expression이 값이 최종 값

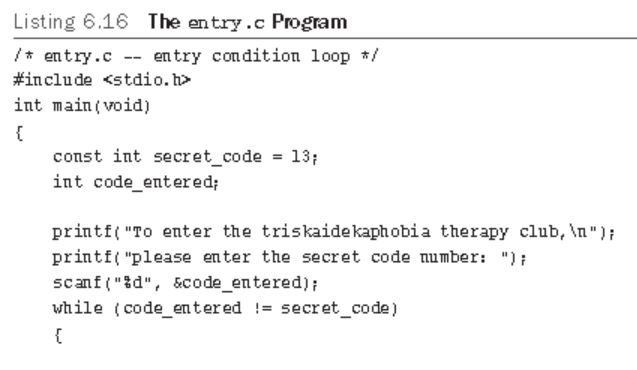
x = (y = 3, (z = ++y + 2) + 5);

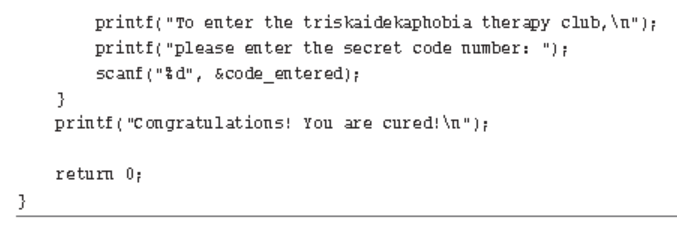
* 1) y = 3, 2) y = 4, 3) z = 4+2 = 6, 4) 6+5 = 11, 5) x = 11

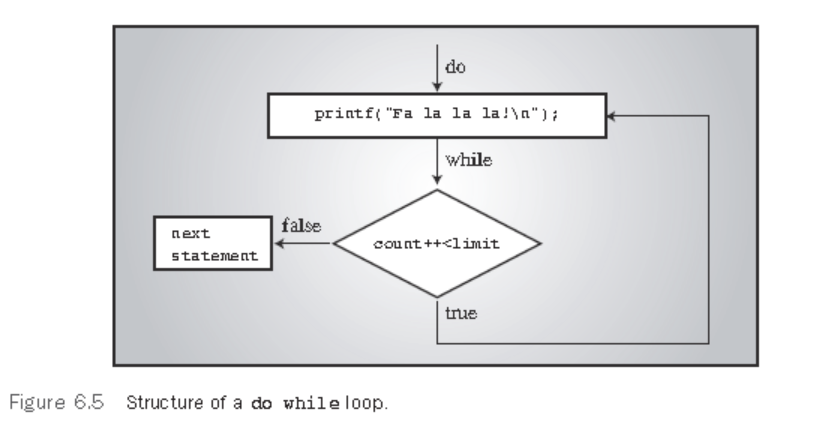
**6.8 Exit-Condition Loop: do while**

* while loop와 for loop는 entry-condition loops이다
* test condition이 loop iteration 전에 checked
* *exit-condition* loop: test condition이 loop iteration 다음에 실행









*do*

*{*

*prompt for password*

*read user input*

*} while (input not equal to password);*

*do*

*{*

*ask user if he or she wants to continue*

*some clever stuff*

*} while (answer is yes);*

**6.9 어떤 유형의 Loop를 사용할 것인가?**

* 먼저 entry-condition loop 또는 exit-condition loop를 결정.
* 둘째, loop test가 loop 시작전에 오는 것이 프로그램을 이해하기 쉽다
* 끝으로 loop test가 만족되지 않으면 loop 자체가 skip된다

for ( ; *test* ; )

is the same as

while ( *test* )

*initialize;*

while ( *test* )

{

*body* ;

*update* ;

}

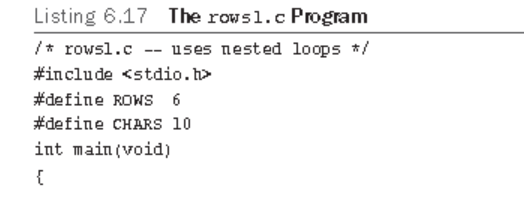
is the same as

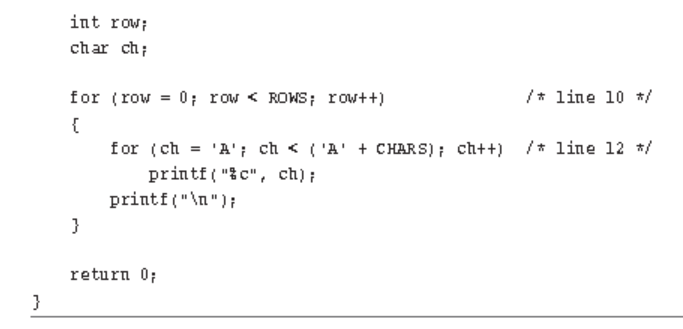
for ( *initialize* ; *test* ; *update* )

*body* ;

**6.10 Nested Loops**

* *nested loop* 는 loop 내에 또다른 loop가 있는 것

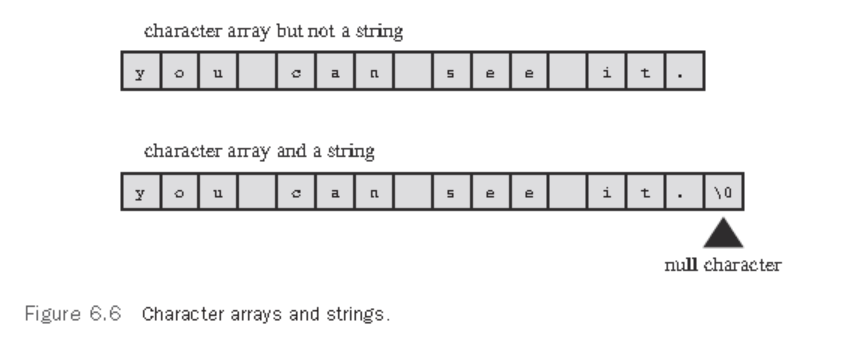




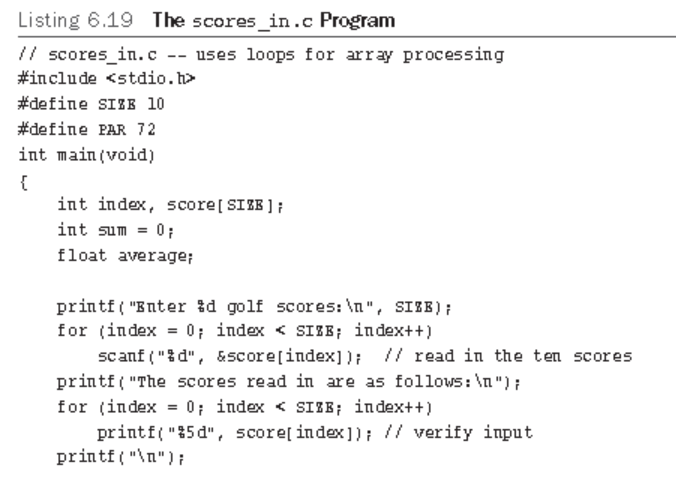
**6.11 Arrays 소개**

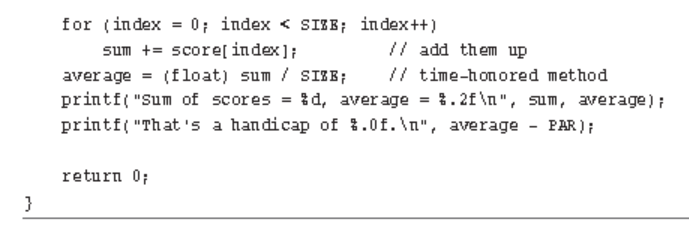
* *array* 은 같은 type의 값이 수열(series)처럼 만드는 것

float debts[20];



**6.11.1 Array 으로 Loop 사용 – 가장 많이 코딩 작성 연습해야 한다**





**6.12 Function Return Value을 사용하는 Loop 예제 – 전형적인 coding style임**

