Fundamentals of Computer System
- C control statements: Branching and Jumps

민기복

Ki-Bok Min, PhD

서울대학교 에너지자원공학과 조교수

Assistant Professor, Energy Resources Engineering

SEOUL NATIONAL UNIVERSITY

Mid-term exam

SEOUL NATIONAL UNIVERSITY

- 채점 중

Lecture feedback (1)

SEOUL NATIONAL UNIVERSITY

- 총 67명이 대답함.
- 교수 강의의 좋은 점
  - 열정적이다(17)
  - 자세한 설명 (14)
  - 영어 강의자료(2)
  - 인상이 좋으시다(1)
  - ....

Lecture feedback (2)

SEOUL NATIONAL UNIVERSITY

- 강의 개선 제안
  - 진도가 빠르고 어렵다 (7)
  - 사람이 너무 많고, 실습을 했으면 (6)
  - 수업자료 및 숙제를 한글로 (9)
  - 강의 노트를 수업 전에 (3)
- 개선방안
  - 수업시간에 나온 예제를 집이나 전산실에서 따로 실습해 주세요
  - 내년에 추가강좌 개설 요청
  - 영어자료 및 숙제는 계속하되 한글용어 혼용
  - 강의 노트는 전날 밤 10시에 업로드 하겠음.

Two weeks ago
Chapter 6. C primer Plus

SEOUL NATIONAL UNIVERSITY

- C control statements: Looping
  - for
  - while

  } Entry-condition loop

  - do while — Exit-condition loop
- What is true?
- Nested loop
- Introduction to **array**
- Using a function return value

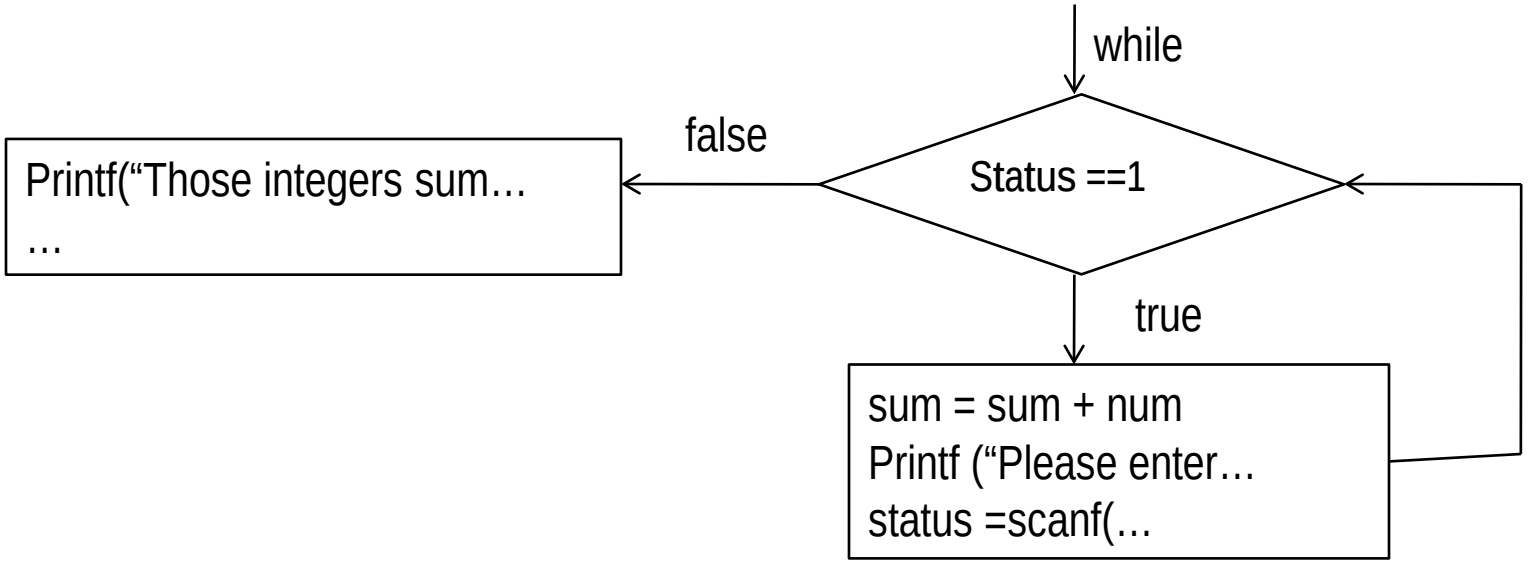
while statement
general form and structure

SEOUL NATIONAL UNIVERSITY

- General form

  while (*expression*)

  statement ----------> One statement without {} or a block with {}



while loop
what is truth? – All non-zero values

SEOUL NATIONAL UNIVERSITY

- Recall that an expression in C always has a value.

```c
/* t_and_f.c -- true and false values in C */
#include <stdio.h>
int main(void)
{
    int true_val, false_val;

    true_val = (10 > 2);    /* value of a true relationship  */
    false_val = (10 == 2);  /* value of a false relationship */
    printf("true = %d; false = %d \n", true_val, false_val);

    return 0;
}
```

expression	Value
-4+6	2
c = 3 + 8	11
5 > 3	1
6 + (c = 3 + 8)	17
q = 5 * 2	10

- An infinite while loop

```
While (1)
{ … }
```

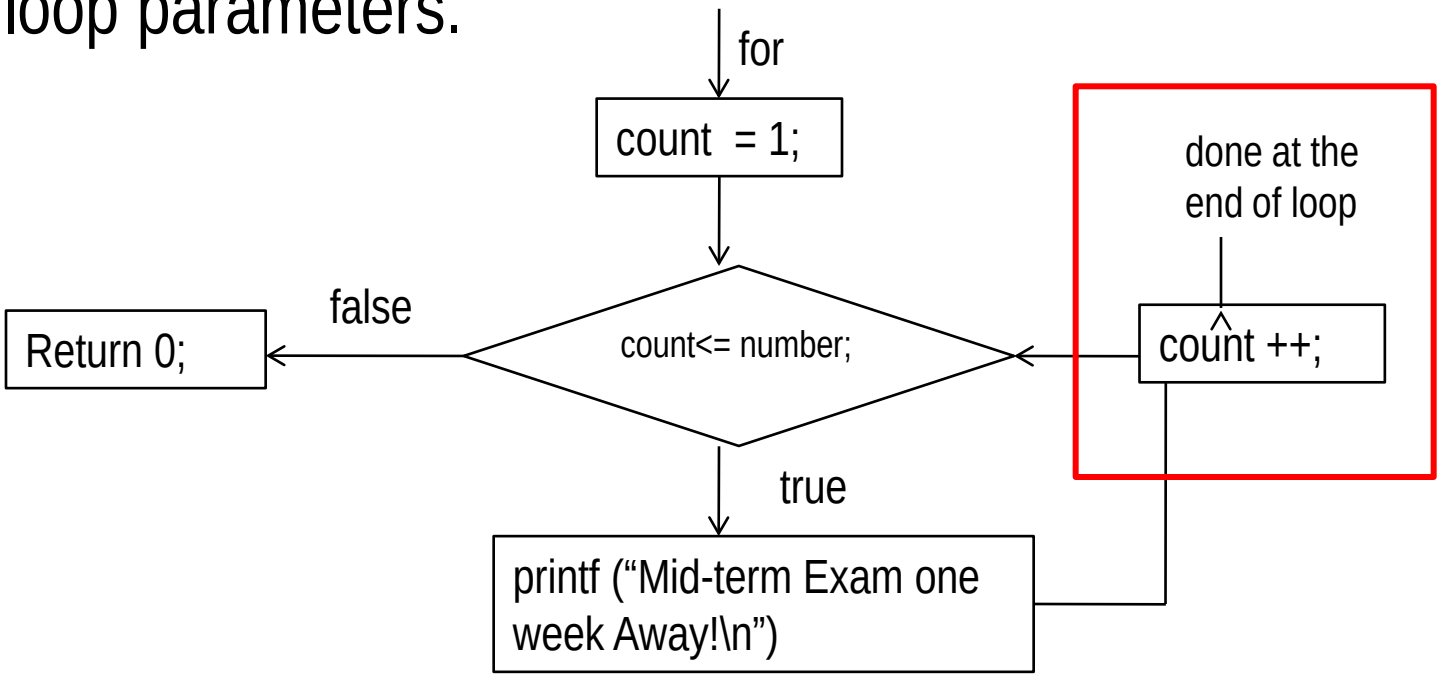
for loop
Form of for loop

SEOUL NATIONAL UNIVERSITY

for (initializing; test; upgrade)

statement ----------> One statement with ; or a block with {}

- The first line of for loop tells us immediately all the information about the loop parameters.



Which loop?
while versus for

SEOUL NATIONAL UNIVERSITY

```
for ( ; test ; )
```
↔
```
while (test)
```

```
initialize;
while(test)
{
        body;
        update;
}
```
↔
```
for (initialize; test; update)
        body;
```

- Initializing & updating → for
- Other than this → while

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

For(count = 1; count<=100;count++)

do while
An exit-condition loop

SEOUL NATIONAL UNIVERSITY

- while loop & for loop : *entry-condition* loop (탈출조건루프)
  - Test is checked before each iteration
  - The statement in the loop may not execute
- do while loop: *exit-condition* loop (진입조건루프)
  - The statements are executed at least once

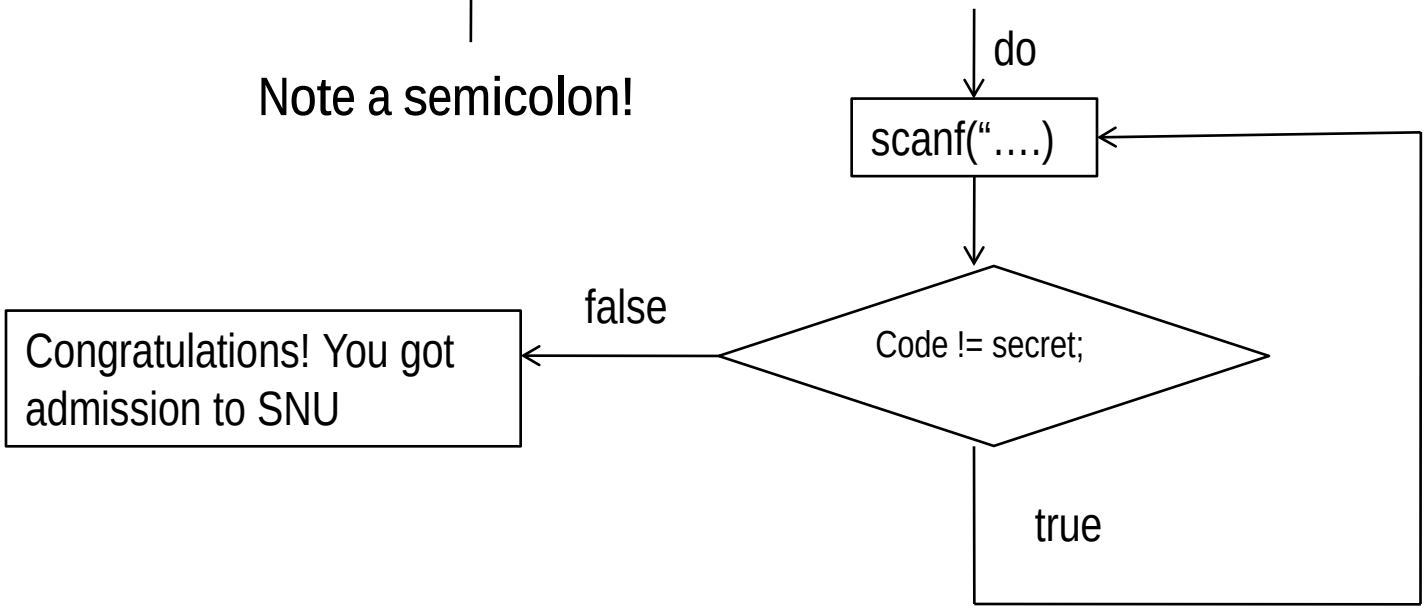
do while
Form of do while loop

do

statement ----------> One statement with ; or a block with {}

while (expression);

↑ Note a semicolon!



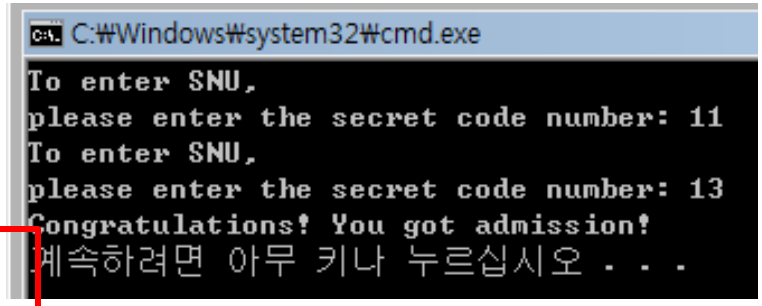
do while
An exit-condition loop

SEOUL NATIONAL UNIVERSITY

```c
/* do_while.c -- exit condition loop */
#include <stdio.h>
int main(void)
{
    const int secret_code = 13;
    int code_entered;

    do
    {
        printf("To enter SNU,₩n");
        printf("please enter the secret code number: ");
        scanf("%d", &code_entered);
    } while (code_entered != secret_code);
    printf("Congratulations! You got admission!₩n");

    return 0;
}
```



while loop - a little longer

```c
printf("To enter SNU,₩n");
printf("please enter the secret code number: ");
scanf("%d", &code_entered);
while (code_entered != secret_code)
{
    printf("To enter SNU,₩n");
    printf("please enter the secret code number: ");
    scanf("%d", &code_entered);
}
```

Nested loop (중첩루프)

SEOUL NATIONAL UNIVERSITY

- Nested loop: one loop inside another loop
- Useful for many cases; e.g.) data in rows and columns

```c
/* rows1.c -- uses nested loops */
#include <stdio.h>
#define ROWS  6
#define CHARS 10
int main(void)
{
    int row;
    char ch;

    for (row = 0; row < ROWS; row++)            /* line 10 */
    {
        for (ch = 'A'; ch < ('A' + CHARS); ch++)  /* line 12 */
            printf("%c", ch);
        printf("\n");
    }

    return 0;
}
```

outer loop

inner loop

Run 10 times for each iteration of outer loop

More assignment operators

+= -= *= /= %=

SEOUL NATIONAL UNIVERSITY

scores += 20	↔	scores = score + 20
dimes -= 2	↔	dimes = dimes +2
bunnies *= 2	↔	bunnies = bunnies *2
time /= 2.73	↔	time = time / 2.73
reduce %= 3	↔	reduce = reduce % 3
x *= 3 * y + 12	↔	x = x * (3 * y + 12)

These assignment operators has low priorities as =

Introducing arrays
- very brief introduction

SEOUL NATIONAL UNIVERSITY

- Arrays (배열): important! & Useful!
- Array: a series of values of the same type stored sequentially. The whole arrays bears a single name.
- 배열: 동일한 데이터 형을 가진 여러 값들이 연속적으로 저장되어 있는 것. 배열 전체가 하나의 이름 사용.
- int score[10];

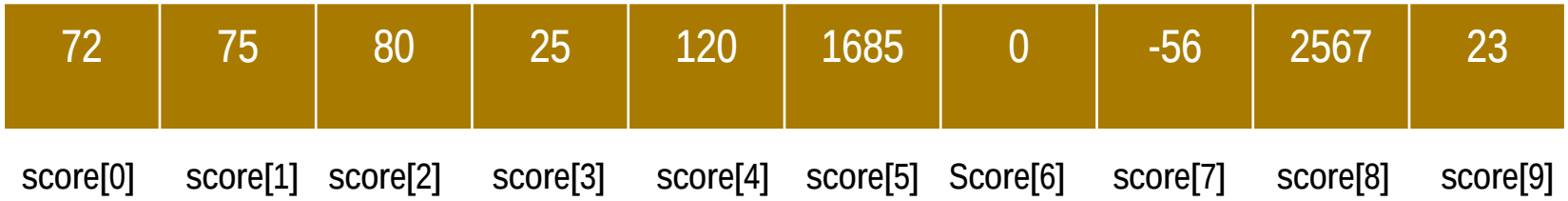
Index, subscript(첨자) or offset

- score is an array with 10 elements. Each of element can hold a type int value

Introducing arrays
- very brief introduction

SEOUL NATIONAL UNIVERSITY

int score[10]



- Numbering starts from 0 (not 1!!!).
- Each element can be assigned a int value.

```
score[4] = 120;          score[9]=23;
```

- 배열원소를 같은 데이터형의 일반 변수를 사용하는 것과 동일한 방식으로 사용가능

```
scanf("%d", &score[4]);
```

- An array can be of any data type

Loop using a function return value

SEOUL NATIONAL UNIVERSITY

```c
// power.c -- 어떤 수의 정수 거듭제곱을 구한다
#include <stdio.h>
double power(double n, int p); // ANSI 프로토타입
int main(void)
{
    double x, xpow;
    int exp;

    printf("어떤 수와, 원하는 거듭제곱수를 양의 정수로");
    printf(" 입력하시오.₩n끝내려면 q를");
    printf(" 입력하시오.₩n");
    while (scanf("%lf%d", &x, &exp) == 2)
    {
        xpow = power(x,exp);    // 함수 호출
        printf("%.3g의 %d제곱은 %.5g입니다.₩n", x, exp, xpow);
        printf("두 수를 입력하시오. 끝내려면 q를 입력하시오.₩n");
    }
    printf("거듭제곱 구하기가 재미 있었나요? -- 안녕!₩n");

    return 0;
}

double power(double n, int p)  // 함수 정의
{
    double pow = 1;
    int i;

    for (i = 1; i <= p; i++)
        pow *= n;

    return pow;                 // pow의 값을 리턴한다
}
```

```
C:₩Windows₩system32₩cmd.exe
어떤 수와, 원하는 거듭제곱수를 양의 정수로 입력하시오.
끝내려면 q를 입력하시오.
23 3
23의 3제곱은 12167입니다.
두 수를 입력하시오. 끝내려면 q를 입력하시오.
15 2
15의 2제곱은 225입니다.
두 수를 입력하시오. 끝내려면 q를 입력하시오.
-34 1
-34의 1제곱은 -34입니다.
두 수를 입력하시오. 끝내려면 q를 입력하시오.
-34 2
-34의 2제곱은 1156입니다.
두 수를 입력하시오. 끝내려면 q를 입력하시오.
q
거듭제곱 구하기가 재미 있었나요? -- 안녕!
계속하려면 아무 키나 누르십시오 . . .
```

Lectures (1st half)

SEOUL NATIONAL UNIVERSITY

- Ch 1. Getting Ready
  - Installing compiler, hello world!
- Ch 2. Introducing C
  - Main(), printf(), comments, 개행문자, syntax/semantic errors
- Ch 3. Data and C (데이터형)
  - scanf(), Integer, char, float, overflow, matching data type
- Ch 4. Character Strings (문자열) and Formatted Input/Output
  - 문자열, format specifier/modifier
- Ch 5. Operators, Expressions and Statements
  - = - * % ++ --, type cast,
- Ch 6. C Control Statements: Looping

Lectures (2nd half)

SEOUL NATIONAL UNIVERSITY

- Ch 7. C control statements: Branching and Jumps
  - If, continue, break, switch, goto
- Ch 9. Functions (함수)
  - Defining, prototyping, calling a function, return
- Ch 10. Arrays and Pointers
  - Initializing, assigning arrays values, pointer operation….
- Ch 11. character strings and string functions
- Ch 12. File Input/Output

Today

Chapter 7. C control statements: Branching and Jumps

SEOUL NATIONAL UNIVERSITY

- C control statements: Branching (분기)
  - if, else : important! & Easy!
  - Switch ~ case
- C control statements: Jumps
  - Continue, break, goto
- Logical operators (논리연산자): && ||
- Character I/O functions (문자 입출력함수)
  - getchar() and putchar()

if statement
general form (1)

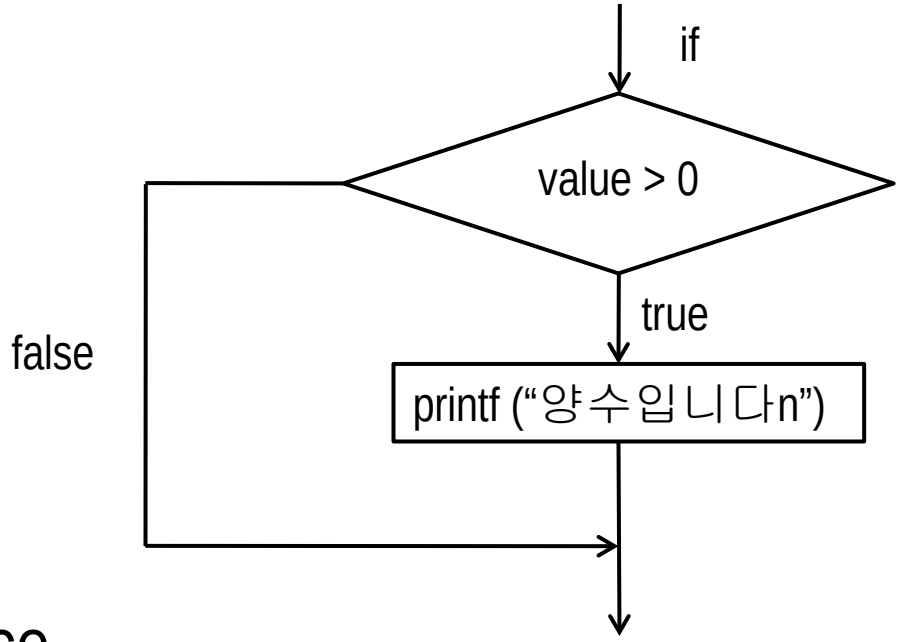
SEOUL NATIONAL UNIVERSITY

```
1  #include <stdio.h>
2  int main(void)
3  {
4     int value;
5     printf("임의의 정수를 입력하고 Enter₩n");
6     printf("정수 : ");
7     scanf("%d", &value);
8
9     if (value>0)
10       printf("양수입니다.");
11
12    return 0;
13 }
```

if statement
general form (1)

if (*expression*)

statement ⟶ a single statement *or* a single block with { }



- Branching statement (분기문)
- Test/execution is done only once
- while???

if statement
general form (1)

SEOUL NATIONAL UNIVERSITY

```
if (score > big)
    printf("우리가 이겼다!\n");        /*single statement*/


If (joe > ron)
{
    joecash++;                          /*compound statement*/
    printf("Ron, 자네가 졌어.\n");     /*compound statement*/
}
```

if statement
general form (2)

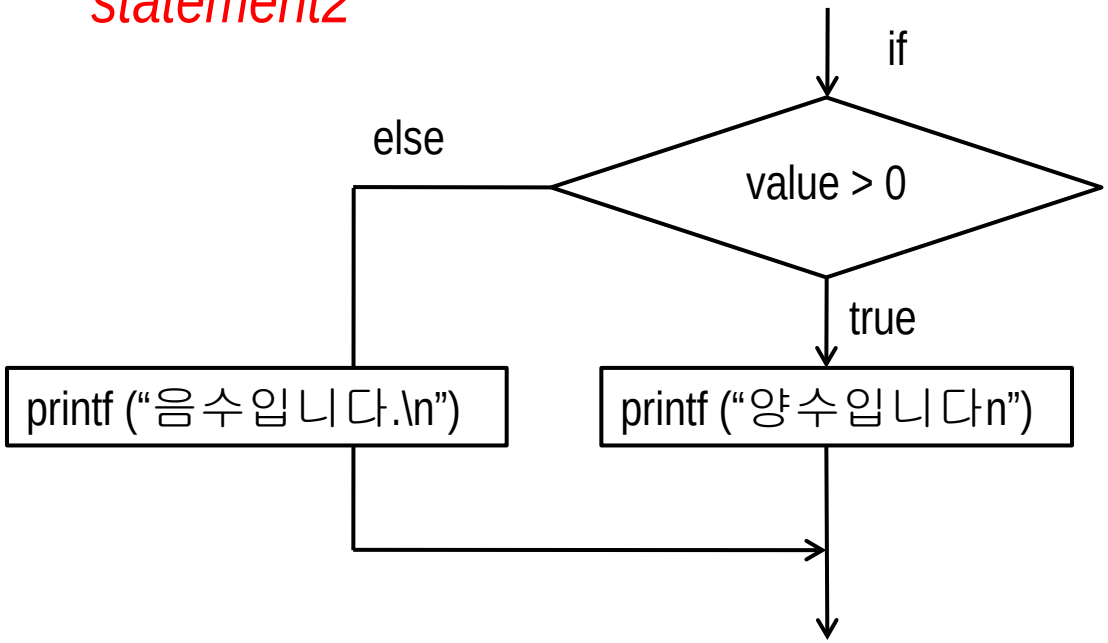
SEOUL NATIONAL UNIVERSITY

```
 1  #include <stdio.h>
 2  int main(void)
 3  {
 4    int value;
 5    printf("임의의 정수를 입력하고 Enter₩n");
 6    printf("정수 : ");
 7    scanf("%d", &value);
 8
 9    if (value>0)
10       printf("양수입니다.₩n");
11    else
12       printf("음수입니다.₩n");
13
14    return 0;
15  }
```

if statement
general form (2)

SEOUL NATIONAL UNIVERSITY

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



if statement
general form (3)

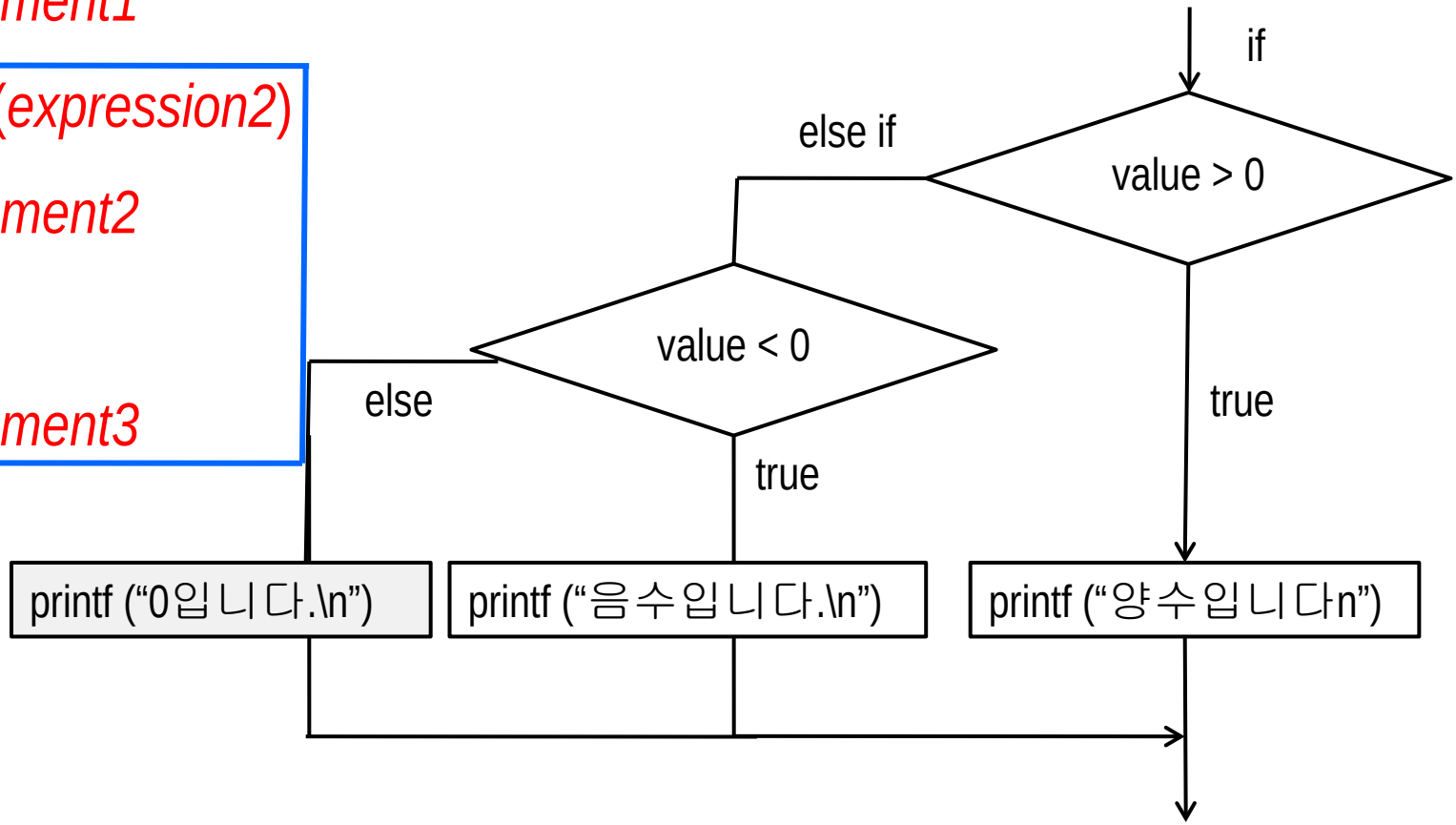
SEOUL NATIONAL UNIVERSITY

```
1  #include <stdio.h>
2  int main(void)
3  {
4     int value;
5     printf("임의의 정수를 입력하고 Enter₩n");
6     printf("정수 : ");
7     scanf("%d", &value);
8
9     if (value>0)
10       printf("양수입니다.₩n");
11    else if (value<0)
12       printf("음수입니다.₩n");
13    else
14       printf("0입니다.₩n");
15
16    return 0;
17 }
```

if statement
general form (3)

SEOUL NATIONAL UNIVERSITY

```
if (expression1)
    statement1
else if (expression2)
    statement2
else
    statement3
```



if statement
general form (3)

SEOUL NATIONAL UNIVERSITY

```
if (score < 1000)
        bonus = 0;
else if (score < 1500)
        bonus = 1;
else if (score < 2000)
        bonus = 2;
else if (score < 2500)
        bonus = 3;
else
        bonus = 4;
```

Multiple levels up to 127

if statement
three general forms

SEOUL NATIONAL UNIVERSITY

```
if (expression)
    statement
```

```
if (expression)
    statement1
else (expression2)
    statement2
```

```
if (expression1)
    statement1
else if (expression2)
    statement2
else
    statement3
```

if statement
pairing **else** with **if**

SEOUL NATIONAL UNIVERSITY

```
if (number > 6)
        if (number <12)
                printf("You are close!\n");
else
        printf("Sorry, you lose a turn!\n");
```

- When number is 5???

if statement
pairing **else** with **if**

SEOUL NATIONAL UNIVERSITY

```
if (number > 6)
        if (number <12)
                printf("You are close!\n");
        else
                printf("Sorry, you lose a turn!\n");
```

- When number is 5???
- Very important to have proper indentations (띄어쓰기)!

Character I/O functions (문자입출력함수)
getchar() & putchar() - 단일문자

SEOUL NATIONAL UNIVERSITY

- getchar()
  - returns the next character from input (입력으로부터 다음 문자를 리턴)
  - No argument
  - ch = getchar(); ←→ scanf("%c, &ch);
- putchar()
  - prints its argument.
  - putchar(ch); ←→ printf("%c", ch);

Character I/O functions (문자입출력함수)
getchar() & putchar() - 단일문자

SEOUL NATIONAL UNIVERSITY

- Getchar() and putchar()
  - faster than printf() and scanf()
  - more compact
  - no need for format specifiers
- Buffer (임시기억저장소)에 있다가 Enter를 누를때 입력 시작

Character I/O functions (문자입출력함수)
getchar() & putchar() - 단일문자

SEOUL NATIONAL UNIVERSITY

```c
/* cypher1.c -- alters input, preserving spaces */
#include <stdio.h>
#define SPACE ' '              /* that's quote-space-quote */
int main(void)
{
    char ch;

    ch = getchar();            /* read a character         */
    while (ch != '₩n')         /* while not end of line    */
    {
        if (ch == SPACE)       /* leave the space          */
            putchar(ch);       /* character unchanged      */
        else
            putchar(ch + 1);   /* change other characters  */
        ch = getchar();        /* get next character       */
    }
    putchar(ch);               /* print the newline        */

    return 0;
}
```

```
a b c d e
b c d e f
계속하려면 아무 키
```

Logical operator
&& || !

SEOUL NATIONAL UNIVERSITY

```
1  // chcount.c  -- use the logical AND operator
2  #include <stdio.h>
3  #define PERIOD '.'
4  int main(void)
5  {
6      int ch;
7      int charcount = 0;
8
9      while ((ch = getchar()) != PERIOD)
10     {
11         if (ch != '"' && ch != '₩'')
12             charcount++;
13     }
14     printf("There are %d non-quote characters.₩n", charcount);
15
16     return 0;
17 }
```

```
"student'.
There are 7 non-quote characters.
계속하려면 아무 키나 누르십시오 . . .
```

logical AND operator &&

Logical operator
&& || !

SEOUL NATIONAL UNIVERSITY

Operator	Meaning
&&	And
||	Or
!	Not

X	Y	X && Y	X || Y	X !
T	T	T	T	F
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

Logical operator
Precedence (우선순위)

SEOUL NATIONAL UNIVERSITY

A > b && b > c || b > d ←→ ((a > b) && (b > c)) || (b > d)

In the order of

> && ||

Use Parentheses () ← clear meaning

Logical operator
Order of evaluation

SEOUL NATIONAL UNIVERSITY

- C does not guarantee which part of the following expressions are evaluated first.
  - Apples = (5 + 3) * (9 + 6);
- Logical operators are exceptions. Logical expressions evaluated from left to right.
- && || ! :sequence points

  Ex) if (num ! 0 && 12/num == 2)

  printf("그 수는 6이다.\n")

  - If num has the zero value, the expression is false, and remaining part is not evaluated any further.

부작용과 시퀀스 포인트

SEOUL NATIONAL UNIVERSITY

- Side effect (부작용) : the modification of data object or file
  - States = 50;
  - For C, the main intent is to evaluating expressions and assignment is ‘side effect’ – doesn’t make sense to us.
- Sequence point : a point at which all side effects are evaluated before going to the next step.
- while (x++ < 10 && (x + y) < 20)
  - X is incremented first before the expression on the right is evaluated.

Logical operator
order of evaluation and range

SEOUL NATIONAL UNIVERSITY

```
if (range >= 90 && range <= 100 )
    printf("A\n");
```

```
if (90<= range <= 100 )
    printf("A\n");
```

Don't do this!

Semantic error!

(90<= range) <= 100 ← always true

0 or 1

Conditional operator (조건연산자) ?:

SEOUL NATIONAL UNIVERSITY

- ?: the only tertiary operator (삼항연산자)

```
x = (y < 0) ? –y : y;
```

←→

```
If (y<0)
    x = -y;
else
    x = y;
```

- General form

Expression1 ? *Expression2* : *Expression 3*

↑ true ↑ false

ex) max = (a > b) ? a : b;

Loop Aids: Continue and break

SEOUL NATIONAL UNIVERSITY

- 처리의 순서나 흐름을 조절하는 제어문
- continue: 해당루프의 나머지를 건너뛰고, 해당 루프의 다음 사이클을 시작.
- break: 어떤 처리의 순서나 흐름을 중단하고, 루프를 탈출.
- 중첩구조에서는 내부구조만 영향을 받음.

Continue and break

SEOUL NATIONAL UNIVERSITY

```
while (( ch = getchar() ) ! EOF)
{
    blabla(ch);
    if (ch == '\n')
            continue
    yakyak(ch);
}
blunder(n,m);
```

```
while (( ch = getchar() ) ! EOF)
{
    blabla(ch);
    if (ch == '\n')
            break;
    yakyak(ch);
}
blunder(n,m);
```

Multiple choices: switch ~ case

SEOUL NATIONAL UNIVERSITY

mals.c 시작 페이지

범위)

```c
/* animals.c -- uses a switch statement */
#include <stdio.h>
#include <ctype.h>
int main(void)
{
    char ch;

    printf("Give me a letter of the alphabet, and I will give ");
    printf("an animal name\nbeginning with that letter.\n");
    printf("Please type in a letter; type # to end my act.\n");
    while ((ch = getchar()) != '#')
    {
        if('\n' == ch)
            continue;
        if (islower(ch))     /* lowercase only          */
            switch (ch)
            {
                case 'a' :
                      printf("argali, a wild sheep of Asia\n");
                      break;
                case 'b' :
                      printf("babirusa, a wild pig of Malay\n");
                      break;
                case 'c' :
                      printf("coati, racoonlike mammal\n");
                      break;
                case 'd' :
                      printf("desman, aquatic, molelike critter\n"
                      break;
                case 'e' :
                      printf("echidna, the spiny anteater\n");
                      break;
                case 'f' :
                      printf("fisher, brownish marten\n");
                      break;
                default :
                      printf("That's a stumper!\n");
            }                /* end of switch           */
        else
            printf("I recognize only lowercase letters.\n");
        while (getchar( ) != '\n')
              continue;      /* skip rest of input line */
        printf("Please type another letter or a #.\n");
    }                        /* while loop end          */
    printf("Bye!\n");

    return 0;
}
```

C:\Windows\system32\cmd.exe

```
Give me a letter of the alphabet, and I will give an animal name
beginning with that letter.
Please type in a letter; type # to end my act.
a
argali, a wild sheep of Asia
Please type another letter or a #.
b
babirusa, a wild pig of Malay
Please type another letter or a #.
c
coati, racoonlike mammal
Please type another letter or a #.
d
desman, aquatic, molelike critter
Please type another letter or a #.
#
Bye!
계속하려면 아무 키나 누르십시오 . . .
```

Multiple choices: switch ~ case

SEOUL NATIONAL UNIVERSITY

```
switch (integer expression)
{
    case constant1:
            statement1;

    case constant2:
            statement2;

    default:
            statements;
}
```

- expression & case label : **integer** (including **char**)
- No case label matches expression: default

Multiple choices: switch ~ case

SEOUL NATIONAL UNIVERSITY

Switch~case with break

```
switch(number)
{
case: statement 1;
      break;
case: statement 2;
      break;
case: statement 3;
      break;
default: statement 4;
}
statement 5;
```

```
switch(number)
{
case: statement 1;
case: statement 2;
case: statement 3;
default: statement 4;
}
statement 5;
```

Switch~case without break

Multiple choices: switch ~ case

SEOUL NATIONAL UNIVERSITY

- Ex)

choice = 1, 2, 3, 4, 5?

```
Switch (choice)
{
    case 1:
    case 2: printf("파이팅!\n"); break
    case 3: printf("잘했다!\n")
    case 4: printf("멋지다!\n"); break
    default: printf("잘 지내라.\n");
}
```

goto
조심해서 쓸 것

SEOUL NATIONAL UNIVERSITY

변수이름 규칙과 동일

goto *label*;

..

..

..

label: statement

```
while (funct > 0)
{
    for (I = 1; I <=100; i++)
    {
                for (j = 1; j <=50; j++)
                {
                ...
                if(xxx)
                        goto help;
                ...
                }
    ...
    }
..
}...
help: xxx
```

Today

Chapter 7. C control statements: Branching and Jumps

SEOUL NATIONAL UNIVERSITY

- C control statements: Branching (분기)
  - if, else : important! & Easy!
  - Switch~case
- C control statements: Jumps
  - Continue, break, goto
- Logical operators (논리연산자): && ||
- Character I/O functions (문자 입출력함수)
  - getchar() and putchar()

Next week
Chapter 9. Functions (함수)

SEOUL NATIONAL UNIVERSITY

- Functions and how to define them
- Arguments and return values (전달인자와 리턴값)
- Function types (함수의 데이터형)
- ANSI C prototype
- Recursion (재귀)
- Pointer variables as arguments