Theory of Multi-Paradigm Programming Languages

Lecture 2-2

Chapter 2 Procedural Languages

Overview

Prof. Moonkun Lee, Ph.D.

Division of Computer Science and Engineering
Jeonbuk National University, Republic of Korea

Contents

- **제2장** 절차중심언어
  - **2.1 정의**
  - **2.2 동기**
  - **2.3 속성**
  - **2.4 함수/절차**

2.1 정의

1	function gcd (x, y: in integer) return integer	f: (int, int) -> int
2	is	선언부:
3	a, t, b: integer;	지역변수 선언
4	begin	실행부: 시작
5	a := x;	배정문
6	b := y;	배정문
7	loop	반복문: 시작
8	exit when b = 0; (* gcd(a, 0)=a *)	반복문 종료 조건
9	t := b;	배정문
10	b := a mod b; (* gcd(a, b)=gcd(b, a mode b) *)	배정문: 나머지 연산에 의한
11	a := t;	배정문
12	end loop;	반복문: 끝
13	return a;	반환문
14	end gcd;	실행부: 끝

<프로그램 2-1> Pascal: gcd 예제

[L#]	코드	Loop			
		1	2	3	4
1	function gcd (x, y: in integer) return integer	(**48**,**18**)			
2	is				
3	a, t, b: integer;	[0/0/0]			
4	begin				
5	a := x;	[**48**/0/0]			
6	b := y;	[48/0/**18**]			
7	loop				
8	exit when b = 0;	(b=18)/=0	(b=12)/=0	(b=6)/=0	(b=0)=0
9	t := b;	[48/**18**/18]	[18/**12**/12]	[12/**6**/6]	
10	b := a mod b;	[48/18/**12**]	[18/12/**6**]	[12/6/**0**]	
11	a := t;	[**18**/18/12]	[**12**/12/6]	[**6**/6/0]	
12	end loop;				
13	return a;				**6**
14	end gcd;				

<프로그램 2-2> Pascal: gcd(48,18) 실행

1	int gcd (int x, y)	f: (int, int) -> int
2	{	함수: 시작
3	int a, t, b;	선언부:
4		지역변수 선언
5	a = x;	실행부:
6	b = y;	배정문
7	while (b != 0) /* gcd(a, 0)=a */	배정문
8	{	반복문: 시작
9	t = b;	배정문
10	b = a mod b; /* gcd(a, b)=gcd(b, a mode b) */	배정문: 나머지 연산에 의한
11	a = t;	배정문
12	}	반복문: 끝
13	return a;	반환문
14	}	함수: 끝

<프로그램 2-3> C: gcd

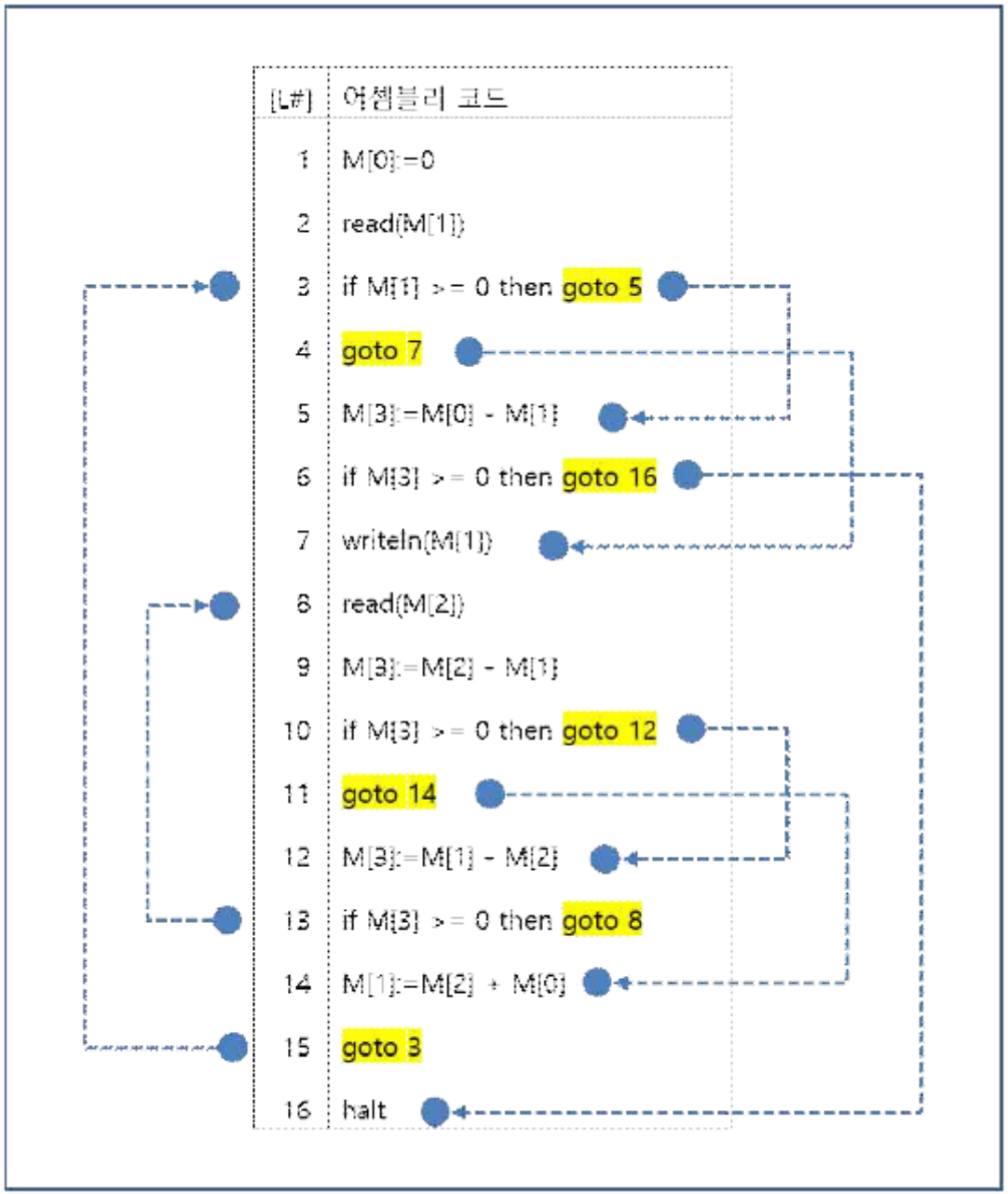
[L#]	코드	Loop			
		1	2	3	4
1	int gcd (int x, y)	(**48**,**18**)			
2	{				
3	int a, t, b;				
4		[0/0/0]			
5	a = x;				
6	b = y;	[**48**/0/0]			
7	while (b != 0)	[48/0/**18**]			
8	{	(b=18)/=0	(b=12)/=0	(b=6)/=0	(b=0)=0
9	t = b;				
10	b = a mod b;	[48/**18**/18]	[18/**12**/12]	[12/**6**/6]	
11	a = t;	[48/18/**12**]	[18/12/**6**]	[12/6/**0**]	
12	}	[**18**/18/12]	[**12**/12/6]	[**6**/6/0]	
13	return a;				**6**
14	}				

<프로그램 2-4> C: gcd(48,18) 실행

[L#]	코드	Loop			
		1	2	3	4
1	int gcd (int x, y)	(**48**,**18**)			
2	{				
3	int a, t, b;	[0/0/0]			
4					
5	a = x;	[**48**/0/0]			
6	b = y;	[48/0/**18**]			
7	while (b != 0)	(b=18) /=0	(b=12) /=0	(b=6) /=0	(b=0)=0
8	{				
9	t = b;	[48/**18**/18]	[18/**12**/12]	[12/**6**/6]	
10	b = a mod b;	[48/18/**12**]	[18/12/**6**]	[12/6/**0**]	
11	a = t;	[**18**/18/12]	[**12**/12/6]	[**6**/6/0]	
12	}				
13	return a;				**6**
14	}				

<프로그램 2-5> C: gcd(48,18) 실행 - 상태변이

2.2 동기

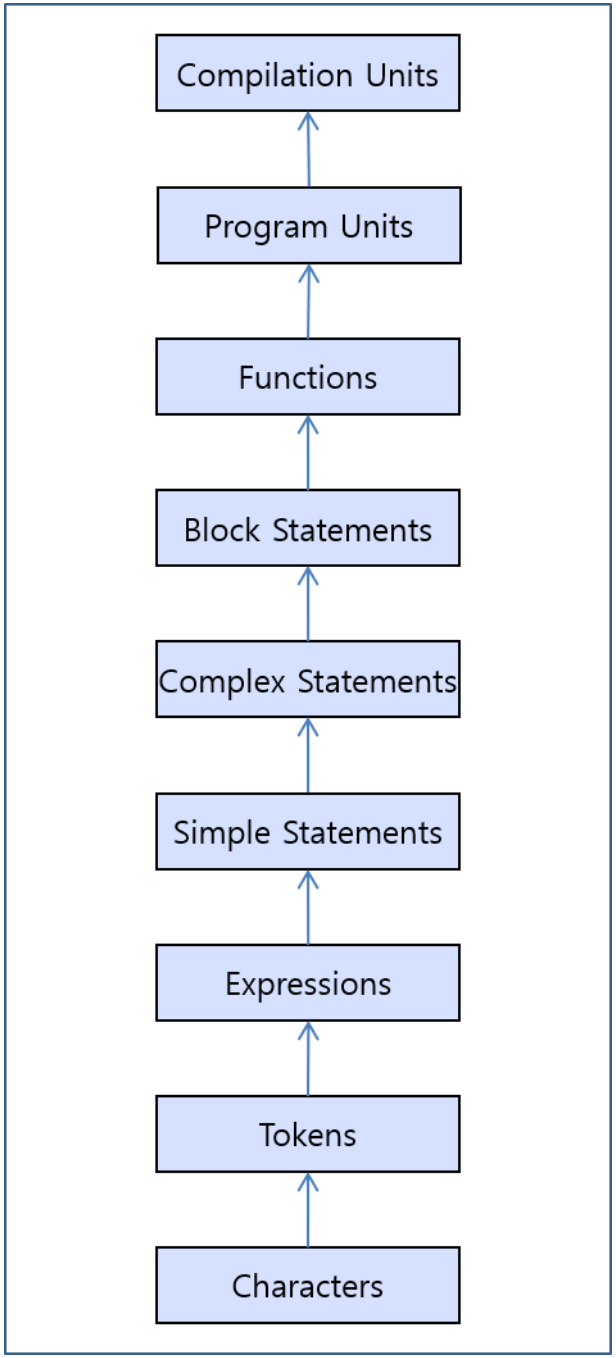


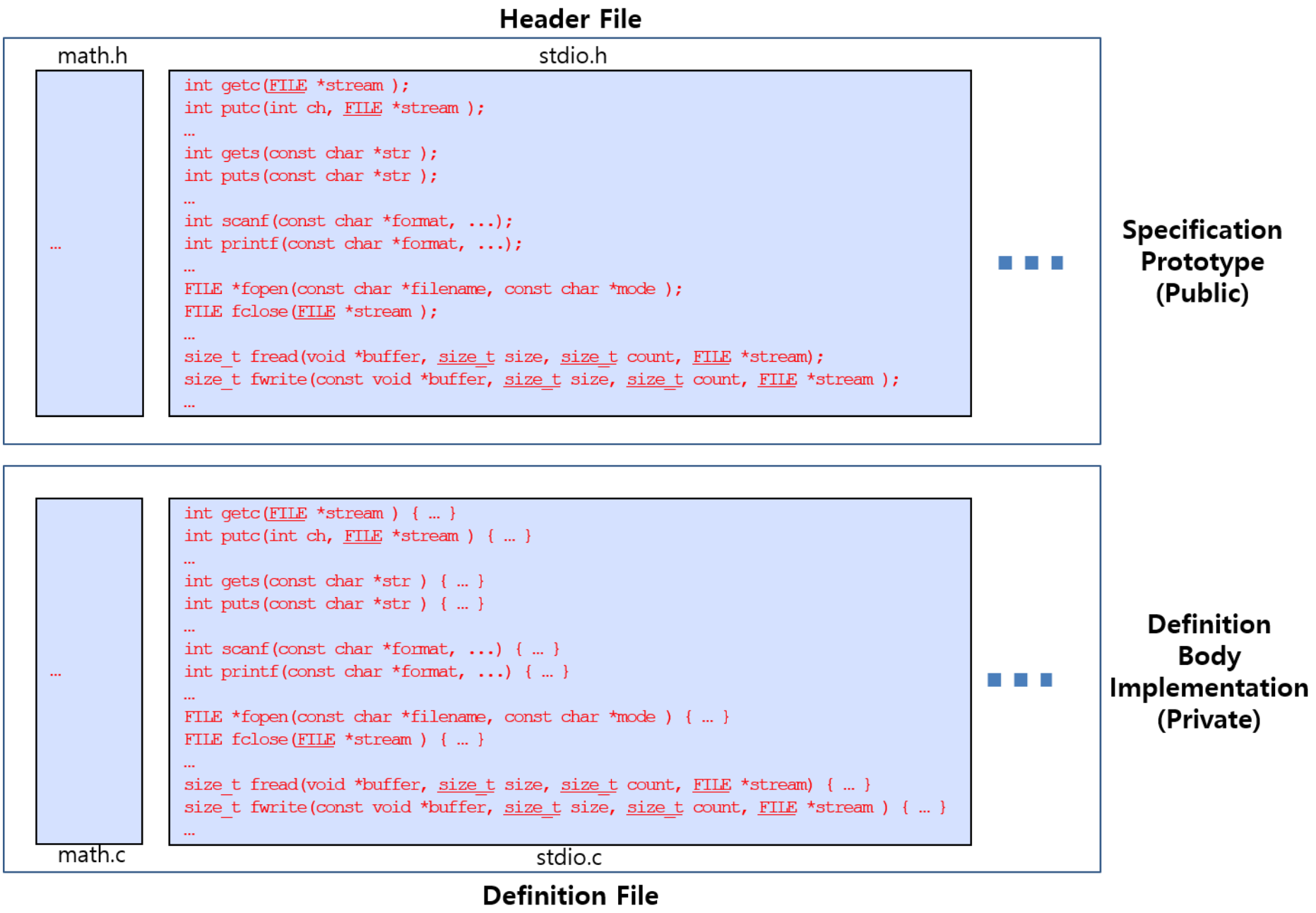
<그림 2-1> 어셈블리 프로그램 예제: 스파게티 코드

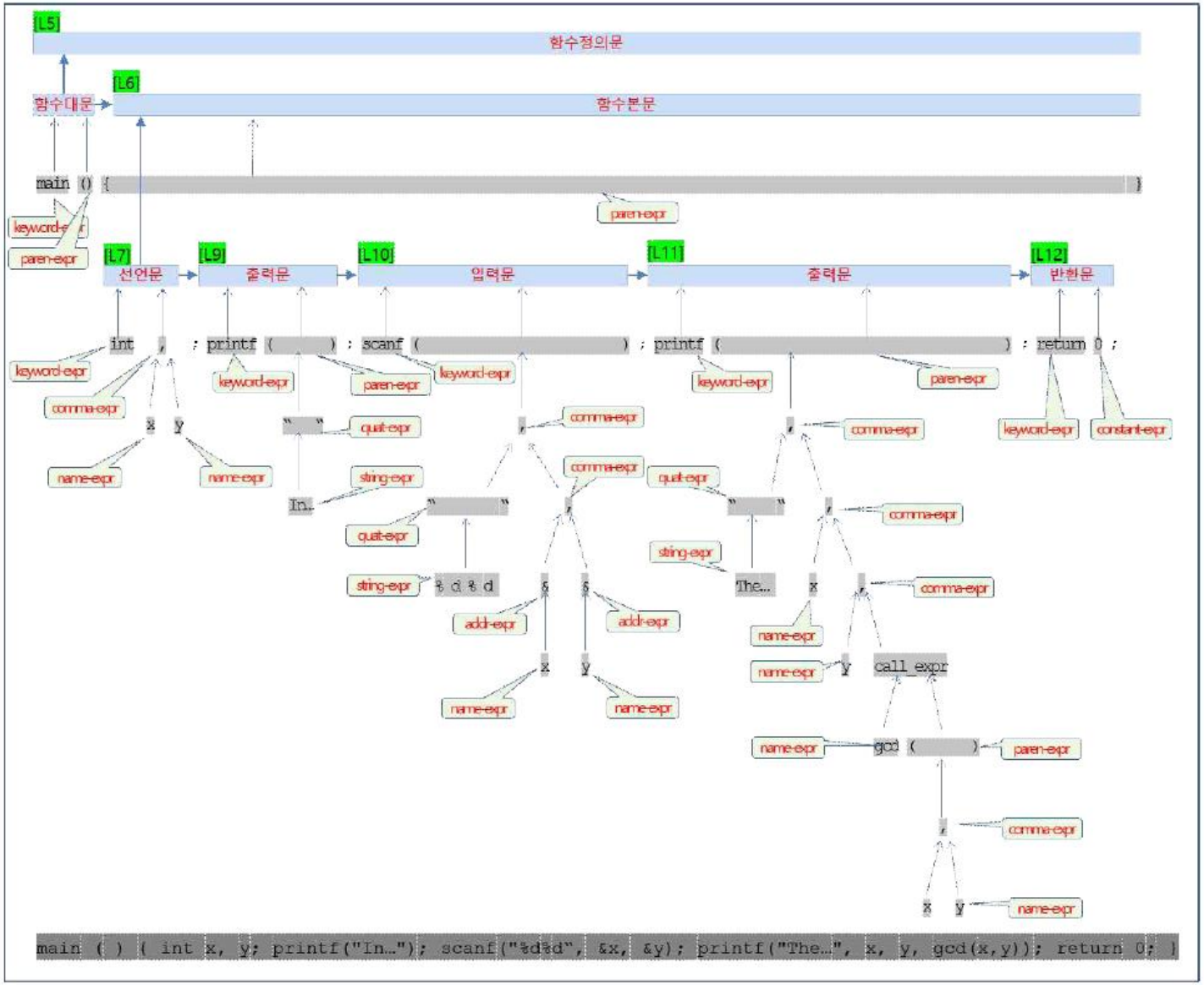
2.3 속성

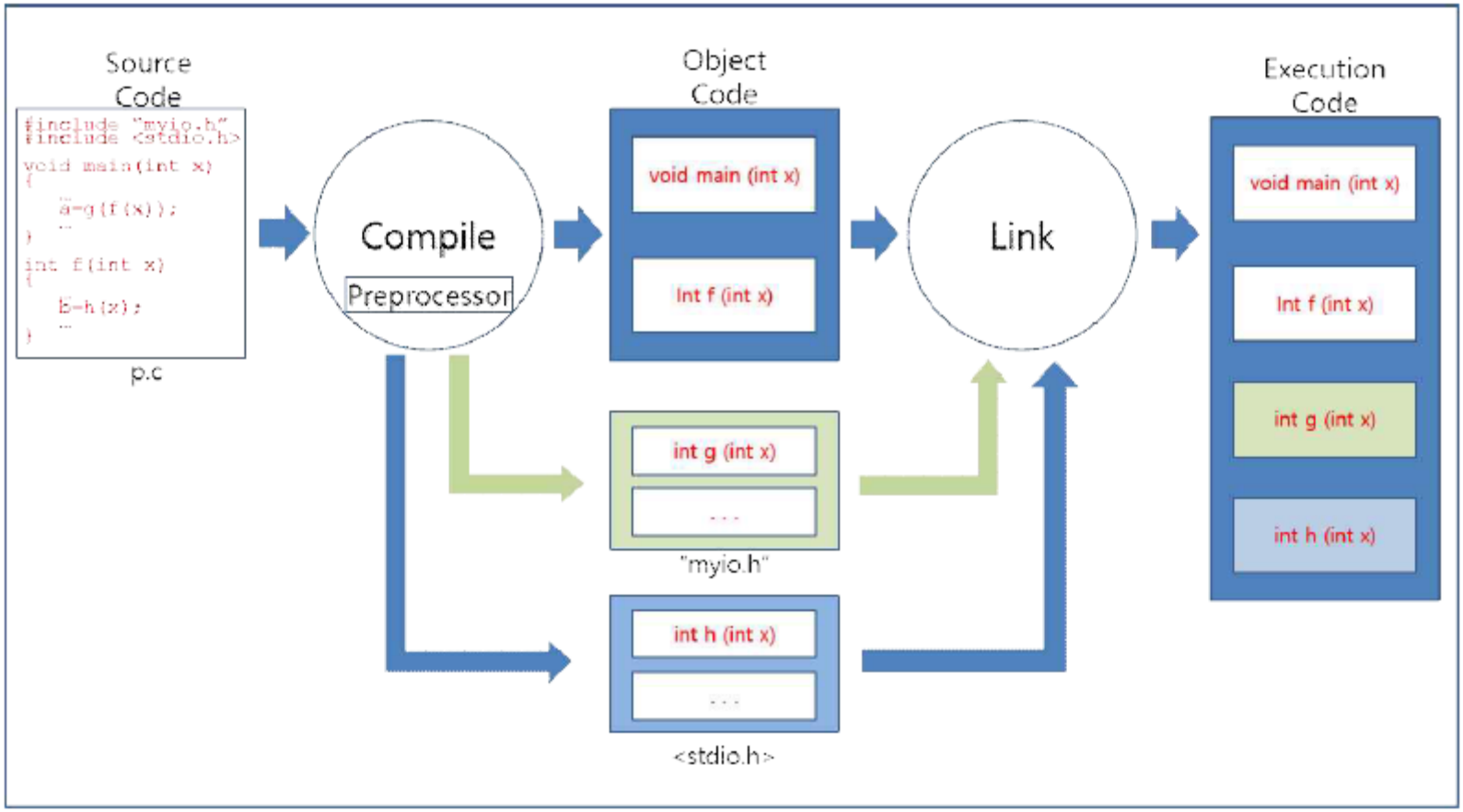
```c
#include <stdio.h>    /* Standard Input/Output Library */

int gcd(int a, int b); /* Function GCD Open Declaration */

main()                 /* Main Function: I/O driver */
{
   int x, y;

   printf("Input two integers: \n");
   scanf("%d%d",&x,&y);
   printf("The gcd of %d and %d is %d \n", x, y, gcd(x,y));
   return 0;
}

int gcd(int a, int b) /* Function GCD Definition */
{
   if (b == 0)
      return a;
   else
      return gcd(b, a % b); /* GCD Tail recursion */
}
```

<프로그램 2-6> C: gcd, main Function

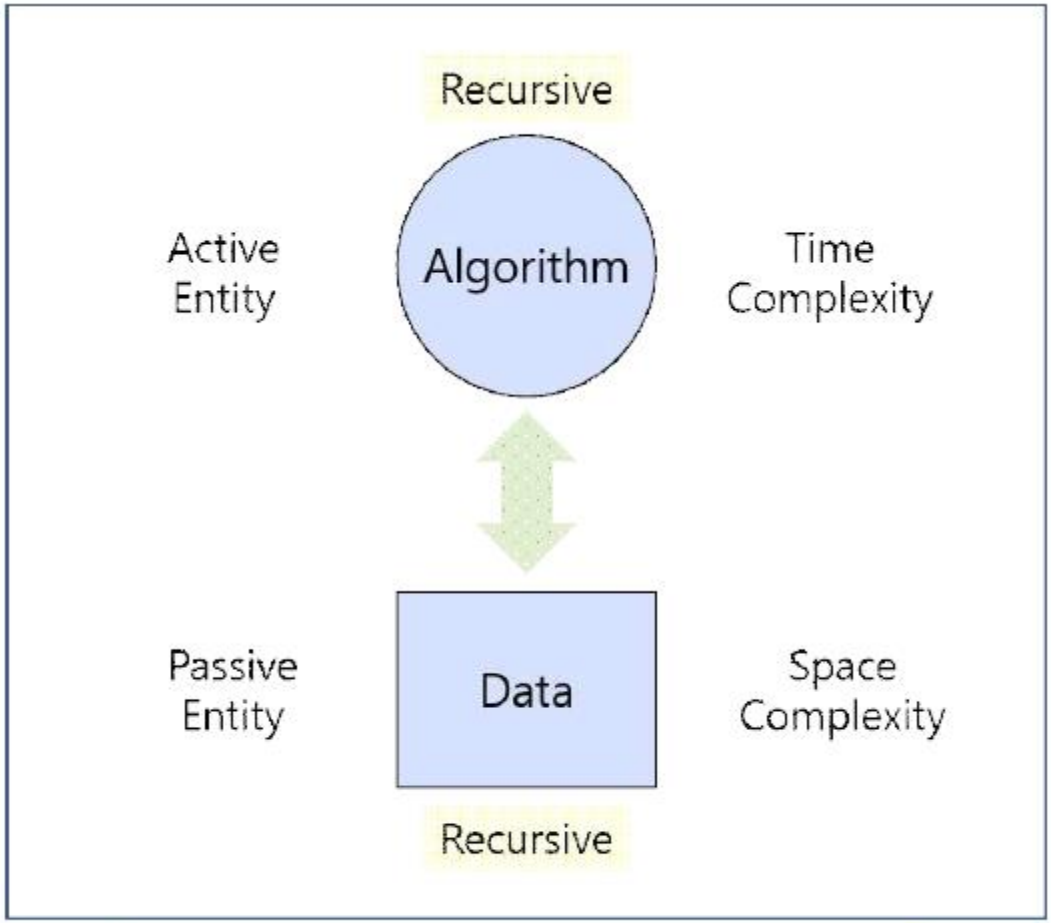

Compilation Units
Program Units
Functions
Block Statements
Complex Statements
Simple Statements
Expressions
Tokens
Characters


Header File
math.h
stdio.h
...
int getc(FILE *stream);
int putc(int ch, FILE *stream);
...
int gets(const char *str);
int puts(const char *str);
...
int scanf(const char *format, ...);
int printf(const char *format, ...);
...
FILE *fopen(const char *filename, const char *mode);
FILE fclose(FILE *stream);
...
size_t fread(void *buffer, size_t size, size_t count, FILE *stream);
size_t fwrite(const void *buffer, size_t size, size_t count, FILE *stream);
...
Specification
Prototype
(Public)
...
int getc(FILE *stream) { ... }
int putc(int ch, FILE *stream) { ... }
...
int gets(const char *str) { ... }
int puts(const char *str) { ... }
...
int scanf(const char *format, ...) { ... }
int printf(const char *format, ...) { ... }
...
FILE *fopen(const char *filename, const char *mode) { ... }
FILE fclose(FILE *stream) { ... }
...
size_t fread(void *buffer, size_t size, size_t count, FILE *stream) { ... }
size_t fwrite(const void *buffer, size_t size, size_t count, FILE *stream) { ... }
...
Definition
Body
Implementation
(Private)
math.c
stdio.c
Definition File


[L5]
함수정의문
[L6]
함수대문
함수본문
main () {
}
paren-expr
keyword-expr
paren-expr
[L7]
선언문
[L9]
출력문
[L10]
입력문
[L11]
출력문
[L12]
반환문
int , ; printf ( ) ; scanf ( ) ; printf ( ) ; return 0 ;
keyword-expr
comma-expr
keyword-expr
paren-expr
keyword-expr
keyword-expr
paren-expr
keyword-expr
constant-expr
x y
name-expr
name-expr
" "
quat-expr
,
comma-expr
,
comma-expr
In..
string-expr
quat-expr
" "
,
comma-expr
quat-expr
" "
,
comma-expr
string-expr
string-expr
%d%d
&
&
The...
x
,
comma-expr
add-expr
add-expr
name-expr
x
y
name-expr
name-expr
y
call_expr
name-expr
gcd ( )
paren-expr
,
comma-expr
x y
name-expr
main ( ) { int x, y; printf("In…"); scanf("%d%d", &x, &y); printf("The…", x, y, gcd(x,y)); return 0; }



<그림 2-14> Compilation Process



<그림 2-15> Algorithm-Data Relationship

```c
#include <stdio.h>

#define IN  1 /* inside  a word */
#define OUT 0 /* outside a word */

/* count lines, words, and characters in input */94)
main()
{
   int c, nl, nw, nc, state;

   state = OUT;
   nl = nw = nc = 0;
   while ((c=getchar()) != EOF)
   {
      ++nc;
      if (c == '\n')
         ++nl;
      if (c == ' ' || c == '\n' || c == '\t')
         state = OUT;
      else if (state == OUT)
     {
         state = IN;
         ++nw;
      }
   }
   printf ("%d %d %d \n", nl, nw, nc);
}
```

<프로그램 2-17> C: 문자 Counting 예제 (1)

```c
#include <stdio.h>

enum STATE { IN, OUT} ;

/* count lines, words, and characters in input */
main()
{
   int c, nl, nw, nc;
   STATE state;

   state = OUT;
   nl = nw = nc = 0;
   while ((c=getchar()) != EOF)
   {
      ++nc;
      if (c == '\n')
         ++nl;
      if (c == ' ' || c == '\n' || c == '\t')
         state = OUT;
      else if (state == OUT)
     {
         state = IN;
         ++nw;
      }
   }
   printf ("%d %d %d \n", nl, nw, nc);
}
```

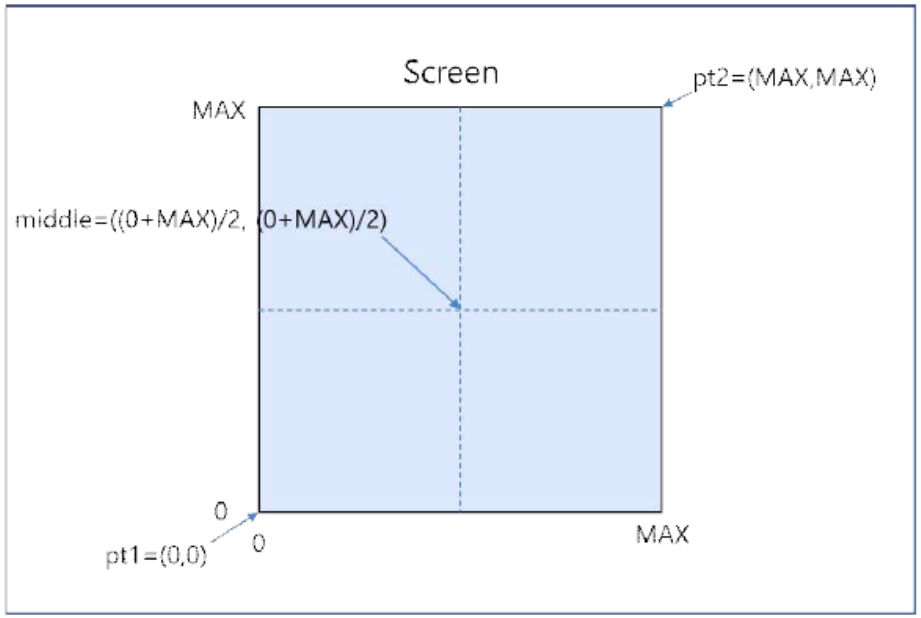
<프로그램 2-18> C: 문자 Counting 예제 (2)

```c
#include <stdio.h>

/* count digits, white space, others */
main()
{
   int c, i, nwhite, nother;
   int ndigits[ 10] ;

   nwhite = nother = 0;
   for (i=0; i<10; ++i)
      ndigit[ i] = 0;

   while ((c=getchar()) != EOF)
   {
      if (c >= '0' && c <= '9')
         ++ndigit[ c-'0'] ;
      if (c == ' ' || c == '\n' || c == '\t')
         ++nwhite;
      else
         ++nother;
   }
   printf("digits = ");
   for (i=0; i<10; ++i)
      printf(" %d", ndigit[ i] );
   printf (", white space =%d, other = %d \n", nwhite, nother);
}
```

<프로그램 2-19> C: 숫자 Counting 예제

```c
struct point
{
   int x;
   int y;
};

struct rect
{
   struct point pt1;
   struct point pt2;
};

struct point makepoint (int x, int y)
{
   struct point temp;

   temp.x = x;
   temp.y = y;
   return temp;
}

main()
{
   struct rect  screen;
   struct point middle;

   screen.pt1 = makepoint(0,0);
   screen.pt2 = makepoint(MAX, MAX);
   middle = makepoint((screen.pt1.x + screen.pt2.x)/2,
                      screen.pt1.y + screen.pt2.y)/2);
}
```

<프로그램 2-20> C: Screen Record 예제 (1)



<그림 2-18> C: Screen Record 예제

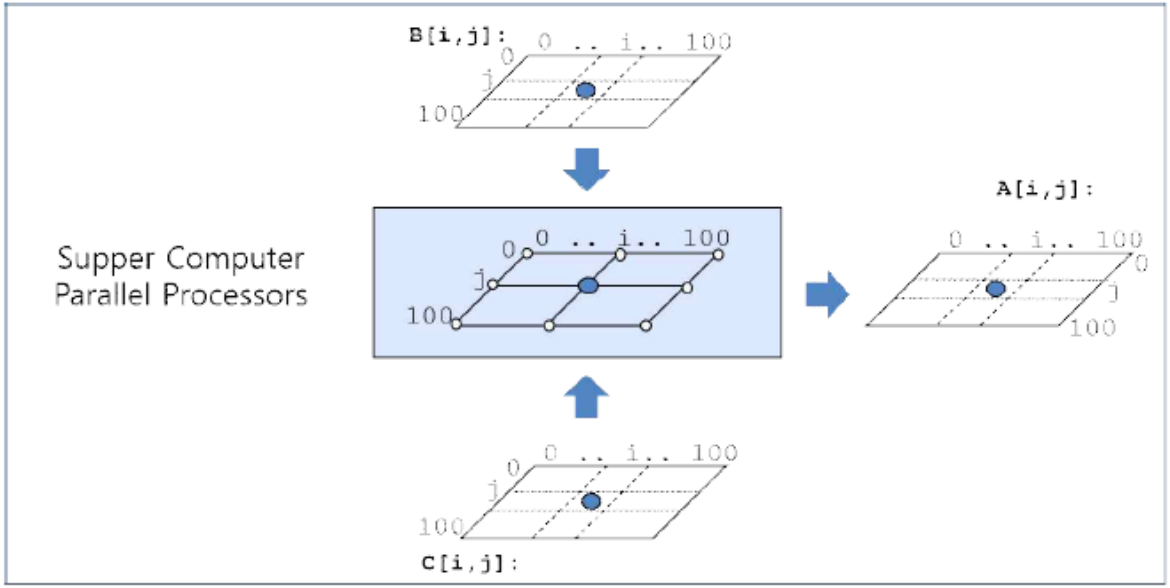
```c
struct point
{
   int x;
   int y;
};

struct rect
{
   struct point pt1;
   struct point pt2;
};

typedef struct point Point;
typedef struct rect  Rect;

Point makepoint (int x, int y)
{
   struct point temp;

   temp.x = x;
   temp.y = y;
   return temp;
}

main()
{
   Rect  screen;
   Point middle;

   screen.pt1 = makepoint(0,0);
   screen.pt2 = makepoint(MAX, MAX);
   middle = makepoint((screen.pt1.x + screen.pt2.x)/2,
                      screen.pt1.y + screen.pt2.y)/2);
}
```

<프로그램 2-21> C: Screen Record 예제 (2)

일반 연산(C)	행렬 연산(C*)

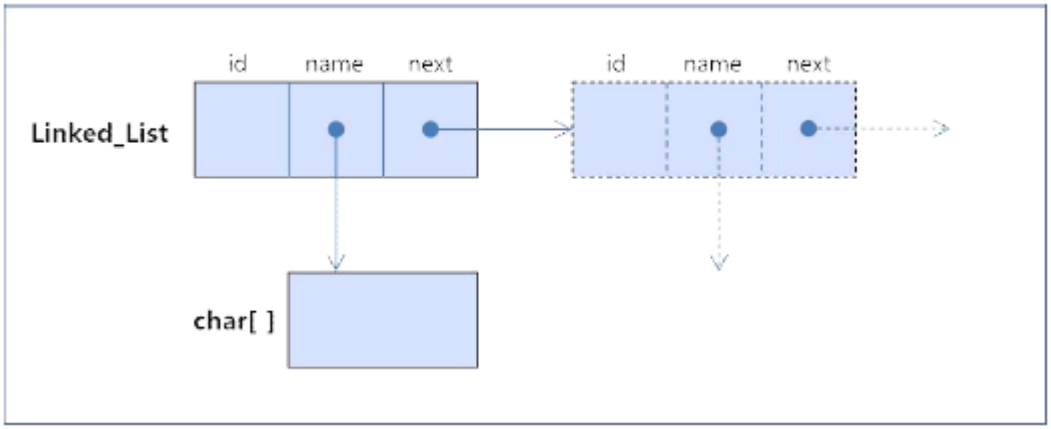
void matrix_addition()
{
int A[100][100] ;
int B[100][100] ;
int C[100][100] ;
int i, j;
 	
/* B,C:initialzation */
 	
for (i=0; i<100; i++)
for (j=0; j<100; j++)
A[i][j] = B[i][j] + C[i][j] ;
}
	void matrix_addition()
	{
	int A[100][100] ;
	int B[100][100] ;
	int C[100][100] ;
	
	/* B,C:initialzation */
	
	A = B + C;
	}

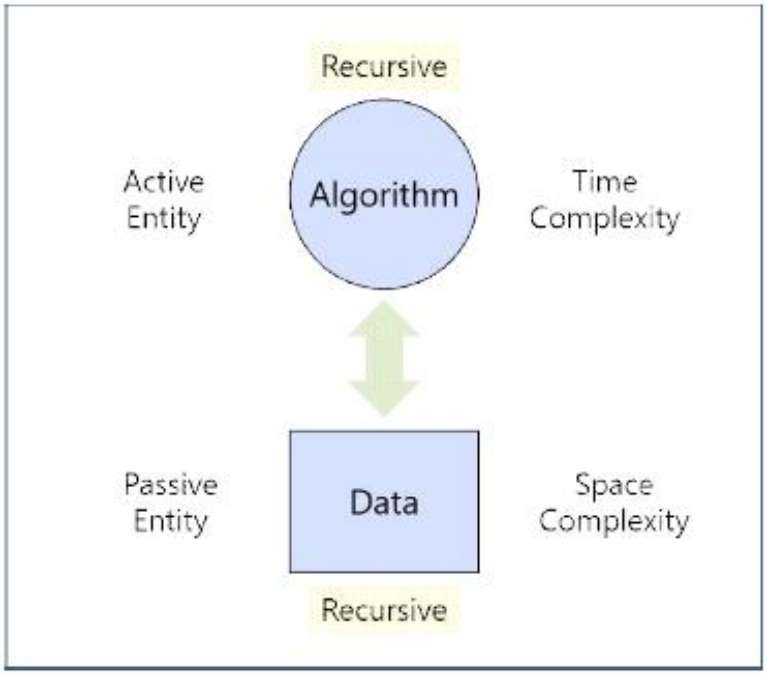
<프로그램

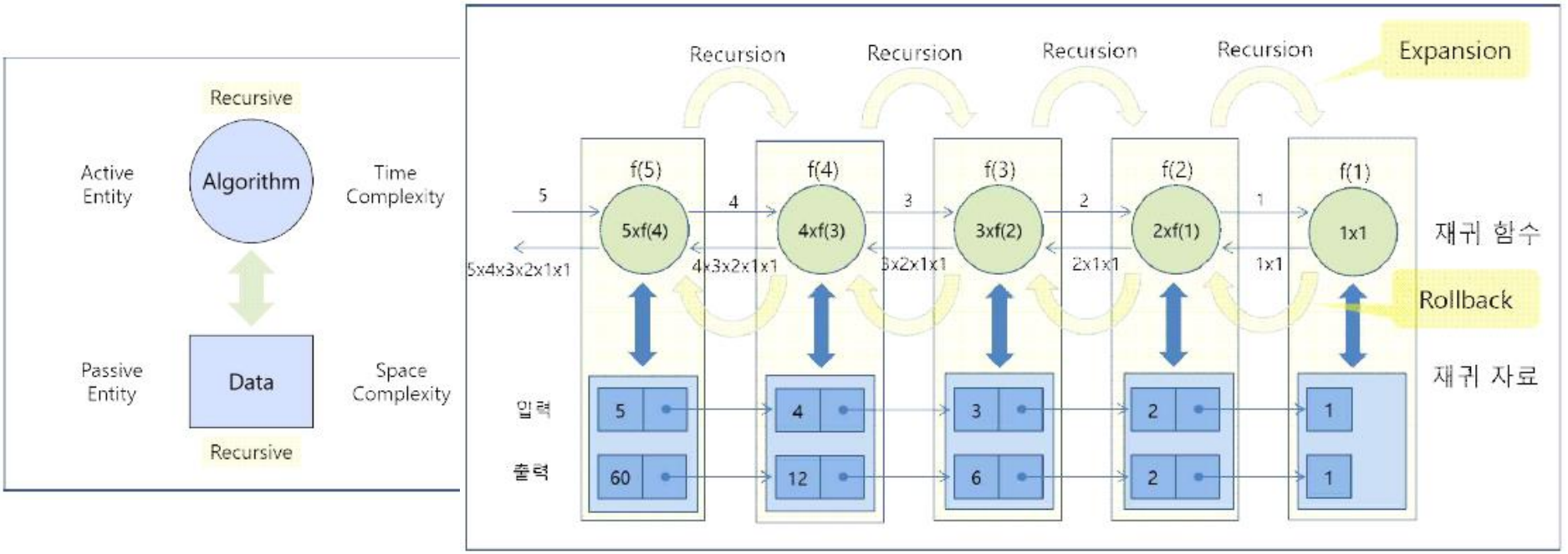


<그림 2-19> Matrix Addition Operation in Super Computer

```
struct linked_list
{
   int    id;
   char * name;
   struct linked_list * next;
};

typedef struct linked_list   LL;
typedef struct linked_list * LLP;

LL   t;
LL * tp;
LLP  tp;
```







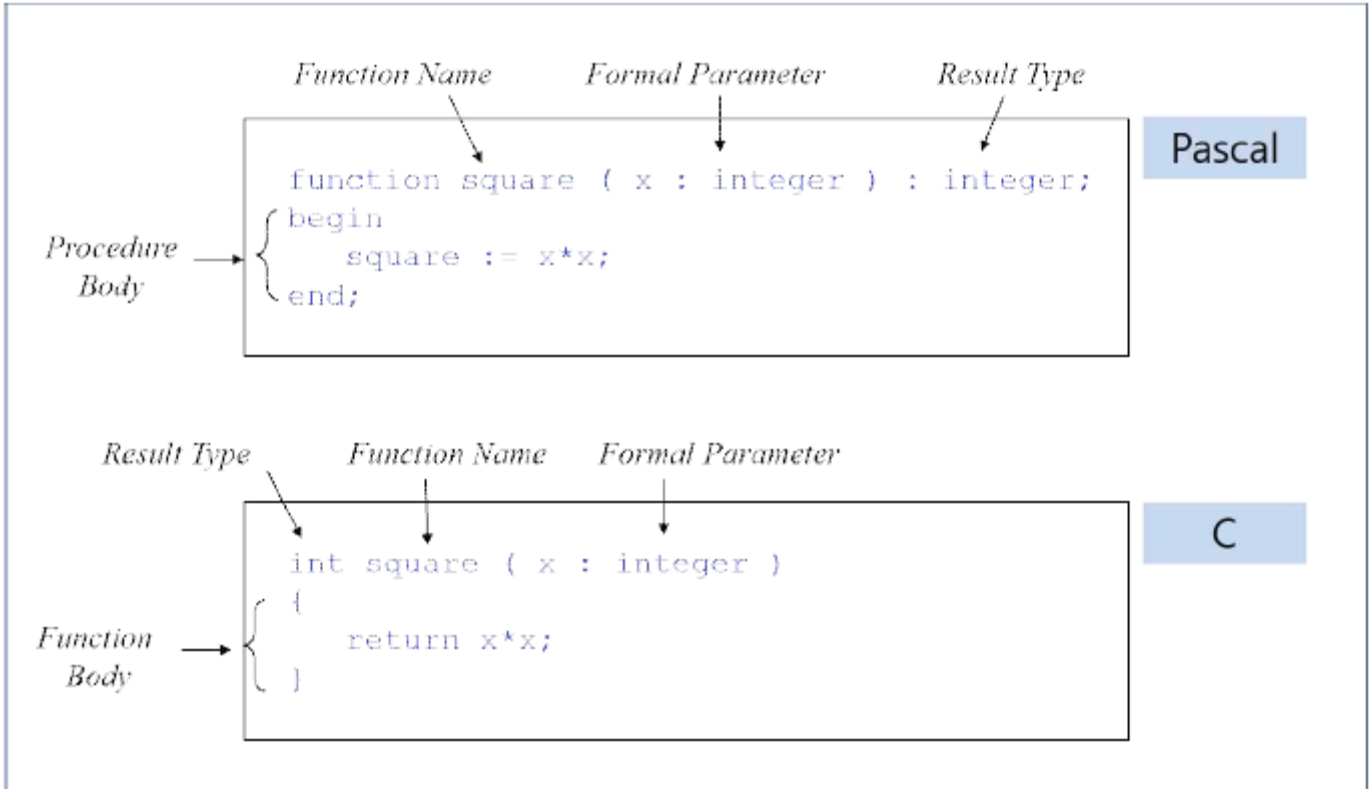
<그림 2-22> Recursion 관계도: factorial 예제

```c
#include <stdio.h>
#include <ctype.h>
#include <string.h>

#define MAXWORD 100

/* the tree node */
struct tnode
{
   char * word;
   int    count;
   struct tnode * left
   struct tnode * right
} ;

struct tnode *talloc(void);
void         treeprint(struct tnode *);
int          getword(char *, int);

/* addtree: add a node with w, at or below p */
struct tnode *addtree(struct tnode *p, char *w)
{
   int cond;

   if (p == NULL)
   {
      p = talloc();
      p->word  = strdup(w);
      p->count = 1;
      p->left = p->right = NULL;
   }
   else
      if ((cond = strcmp(w, p->word)) == 0)
        p->count++;
      else
         if (cond < 0)
            p->left  = addtree(p->left,  w);
         else
            p->right = addtree(p->right, w);
   return p;
}

/* word frequency count */
main()
{
   struct tnode *root;
   char          word[ MAXWORD] ;

   root = NULL;
   while (getword(word, MAXWORD) != EOF)
      if (isalpha(word[ 0] ))
         root = addtree(root,word[ 0] );
   treeprint(root);
}
```

2.4 함수/절차

[L#]	원 코드	프로시저 정의	프로시저 호출
1	begin	procedure getch	begin
2	while eoln do	(var ch: character)	getch(ch)
3	readln;	begin	end;
4	read(ch)	while eoln do	
5	end;	readln;	
6		read(ch)	
7		end;	

<프로그램 2-24> Pascal Procedure


Function Name
Formal Parameter
Result Type
function square (x : integer) : integer;
begin
square := x*x;
end;
Procedure Body
Pascal
Result Type
Function Name
Formal Parameter
int square (x : integer)
{
return x*x;
}
Function Body
C

	L#	Pascal	C
함	1	function square (x : integer) : integer;	int square (int x);
수	2	begin	{
선	3	square := x*x;	return x*x;
언	4	end;	}
함	5	var a, b : integer;	int a, b;
수	6	...	...
호	7	a := square(2);	a = square(2);
출	8	b := square(a);	b = square(a);

<프로그램 2-25> Call-By-Value 예제: square 프로시저/함수 (Pascal, C)

프로시저 선언	1	procedure swap(x, y : integer)
	2	var z: integer;
	3	begin
	4	z:=x;
	5	x:=y;
	6	y:=z;
	7	end;
프로시저 호출	8	var a, b : integer;
	9	...
	10	a:= 1;
	11	b:= 2;
	12	swap(a,b);
	13	write(a,b);

<프로그램 2-26> Call-By-Value 예제: swap - Pascal

프로시저 선언	1	procedure swap(var x : integer; var y : integer)
	2	var z: integer;
	3	begin
	4	z:=x;
	5	x:=y;
	6	y:=z;
	7	end;
프로시저 호출	8	var a, b : integer;
	9	...
	10	a:= 1;
	11	b:= 2;
	12	swap(a,b);
	13	write(a,b);

<프로그램 2-28> Call-By-Reference 예제: swap - Pascal

함수
선언
	1
	2
	3
	4
	5
	6
	7
	8
		void swap(int *px, int *py)
		{
		  int z;
		
		  z = *px;
		  *px = *py;
		  *py = z;
		}

함수
호출
	9
	10
	11
	12
	13
	14
		int a, b;
		...
		a = 1;
		b = 2;
		swap(&a,&b);
		printf("a=%d; b=%d", a,b);

<프로그램 2-29> Call-By-Reference 예제: swap - C

프로시저 선언	1	program
	2	...
	3	i: integer;
	4	...
	5	procedure foo(x,y);
	6	begin
	7	i := y
	8	end;
호출	9	begin
	10	i := 2;
	11	j := 3;
	12	foo(i,j);
	13	end;

<프로그램 2-30> Call-By-Value-Result 예제: foo - Pascal

프로시저 선언	1	procedure swap(x, y : integer)
	2	var z: integer;
	3	begin
	4	z:=x;
	5	x:=y;
	6	y:=z;
	7	end;
프로시저 호출	8	var i : integer;
	9	var a: array 1..10 of integer;
	10	...
	11	i := 3;
	12	a[i] := 4;
	13	swap(i,a[i]);
	14	write(i,a[i]);

<프로그램 2-31> Call-By-Name 예제: swap - Pascal

i:=3; a[i] =4;	코드	변수값					
					i	a[3]	a[4]
		ix	a[i]		3	4	?

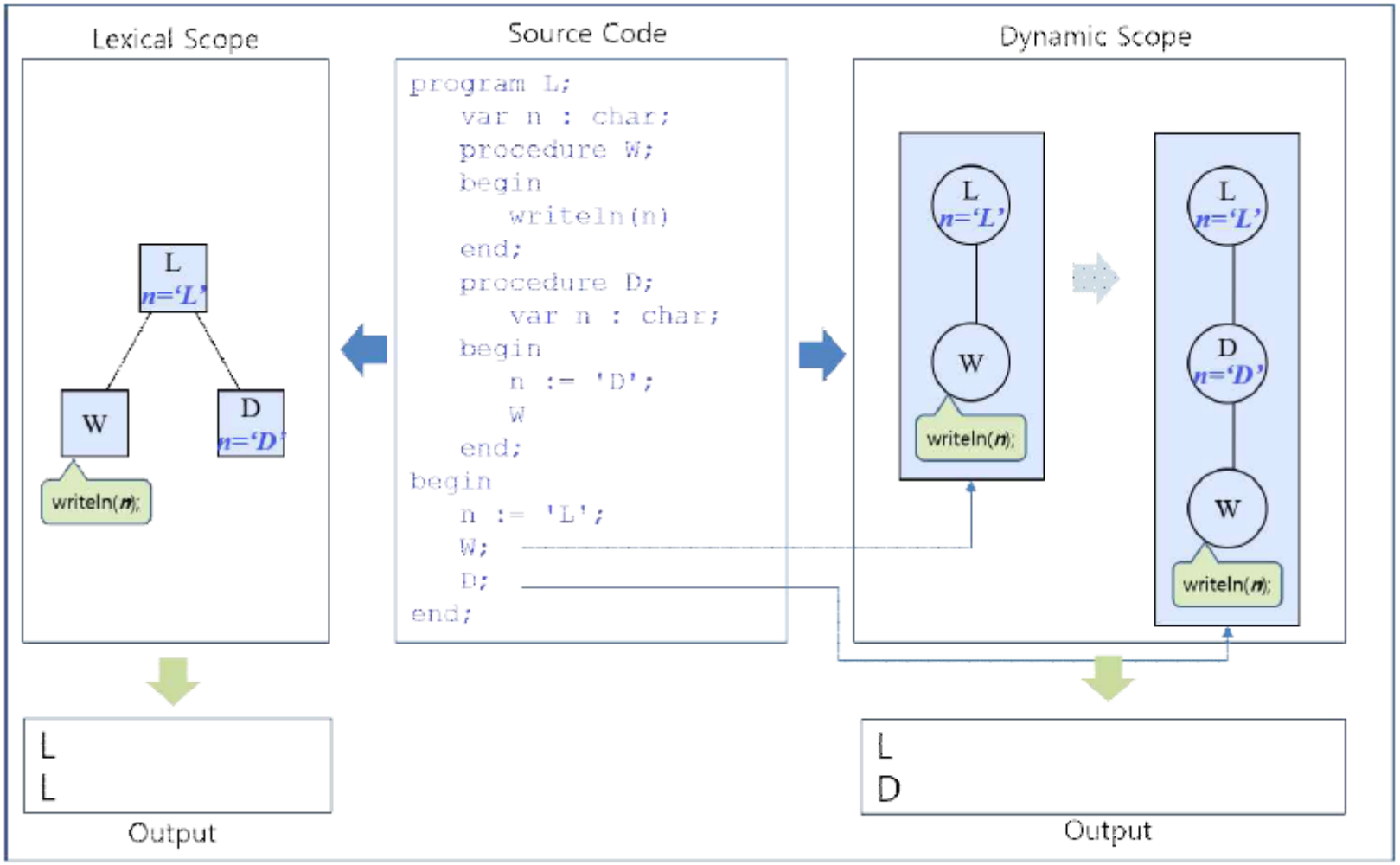
swap(i, a[i]);	procedure swap ((x←i),(y←a[i]))
	  var temp: integer;
	begin
	  temp := i;
	  i := a[i] ;
	  a[i] := temp;
	end swap;
		x	y				

				temp			
				3			
					4		
							3
write(i,a[i]);					**4**	**4**	3

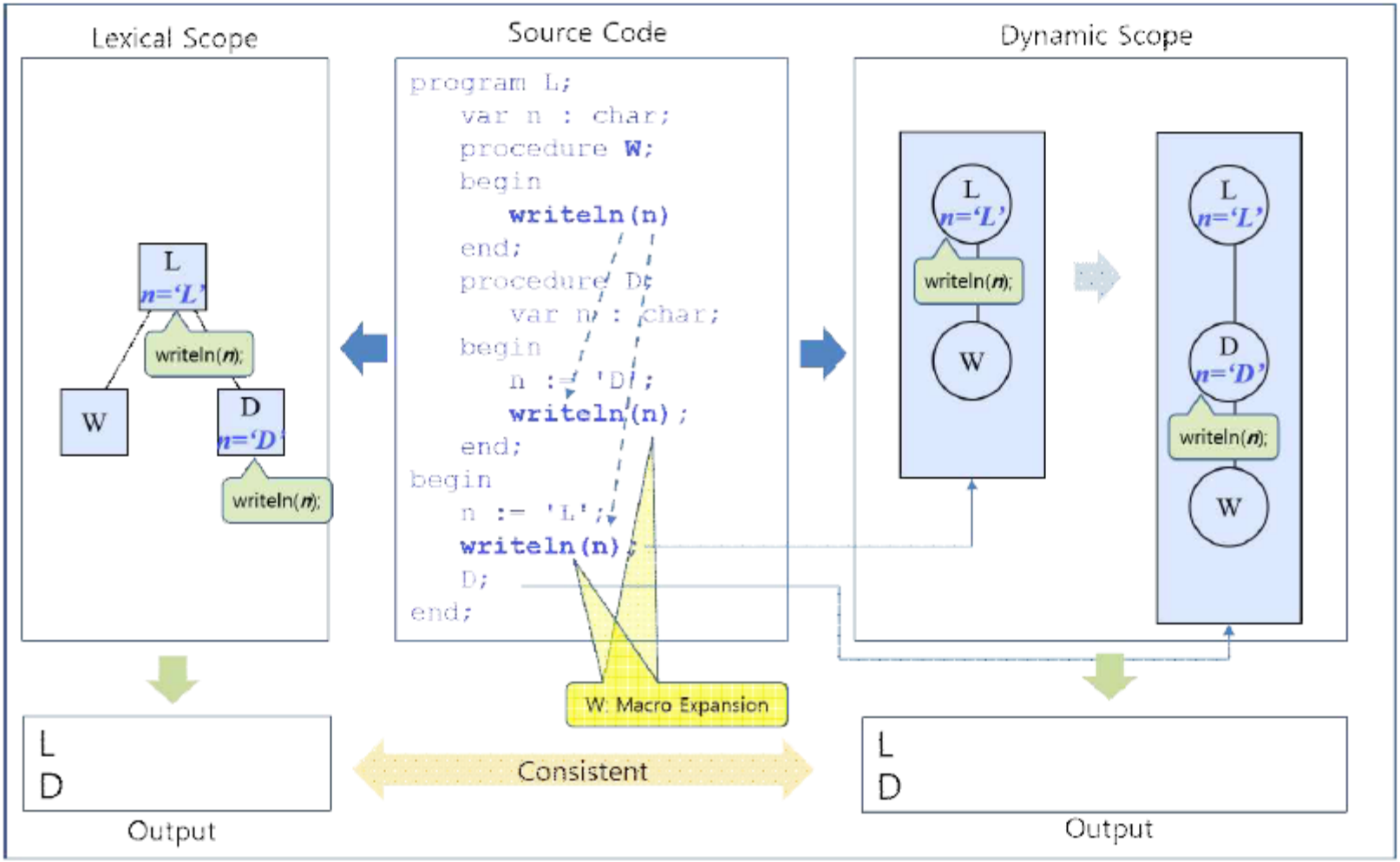
<프로그램 2-32> Call-By-Name 예제: 대입 코드

```
 1  program L;
 2     var n : char;
 3     procedure W;
 4     begin
 5        writeln(n)
 6     end;
 7     procedure D;
 8        var n : char;
 9     begin
10        n := 'D';
11        W
12     end;
13  begin
14     n := 'L';
15     W;
16     D;
17  end;
```

<프로그램 2-33> Pascal: Scope 예제



<그림 2-24> Scope의 차이점: Pascal



<그림 2-24-1> Scope Rule - Consistency

Name Conflict

Renaming

원문	Macro Expansion	Renaming

procedure P(x);
begin
  int i;
  ...
  i := i+n;
  x := x+n;
  ...
end;
	procedure P(A[i]);
	begin
	  int i;
	  ...
	  i := i+n;
	  A[i] := A[i]+n;
	  ...
	end;
		procedure P(A[i]);
		begin
		  int j;
		  ...
		  j := j+n;
		  A[i] := A[i]+n;
		  ...
		end;

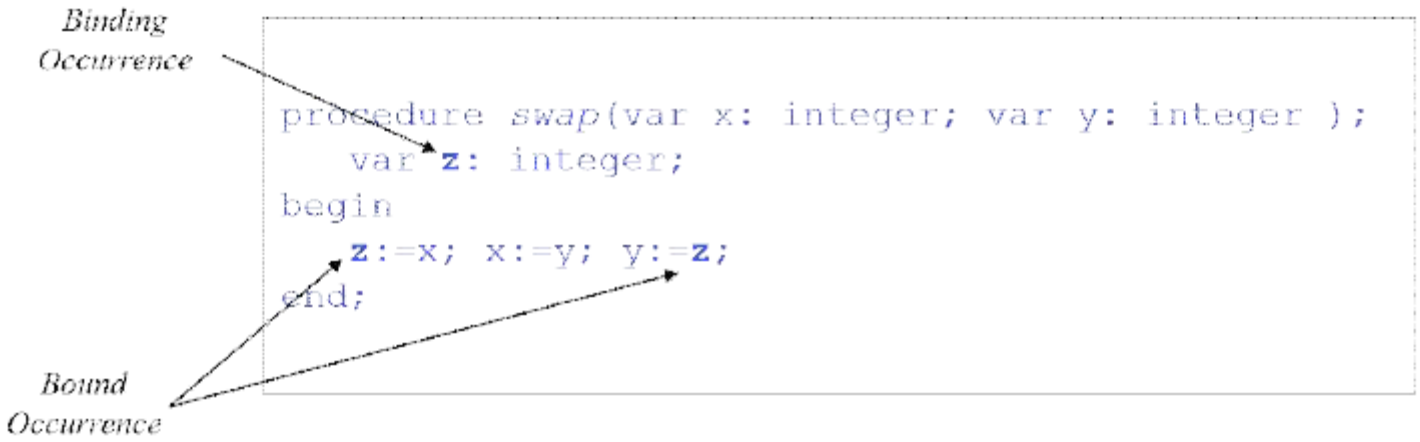
<그림 2-25> Name Conflict: Renaming

원문	Macro Expansion

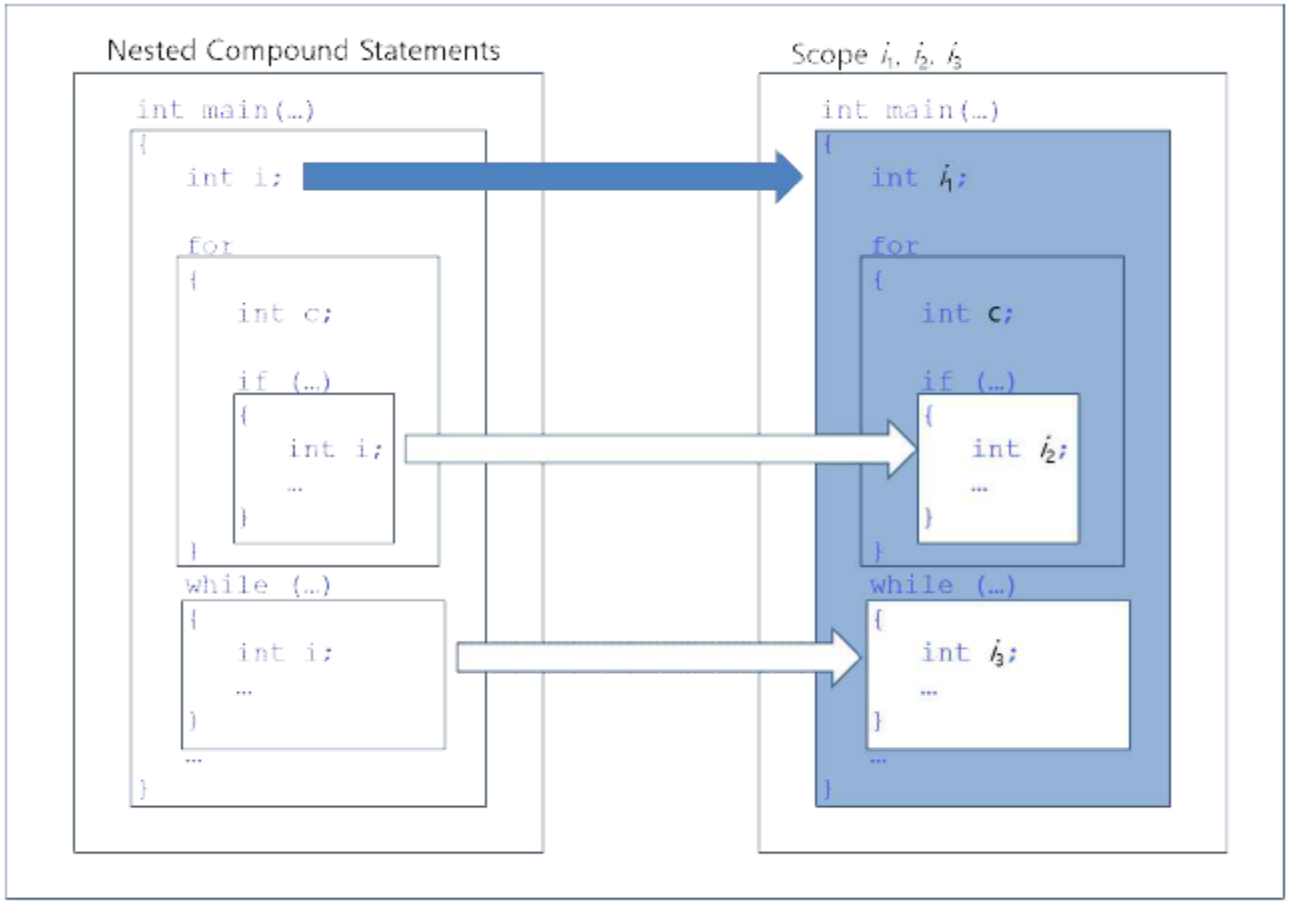
begin
  integer n;
  ...
  n := n-1;
  P(A[i]);
  ...
end;
	begin
	  integer m;
	  ...
	  m := m-1;
	  int j;
	  ...
	  j := j+n;
	  A[i] := A[i]+n;
	  ...
	end;

Macro Expansion

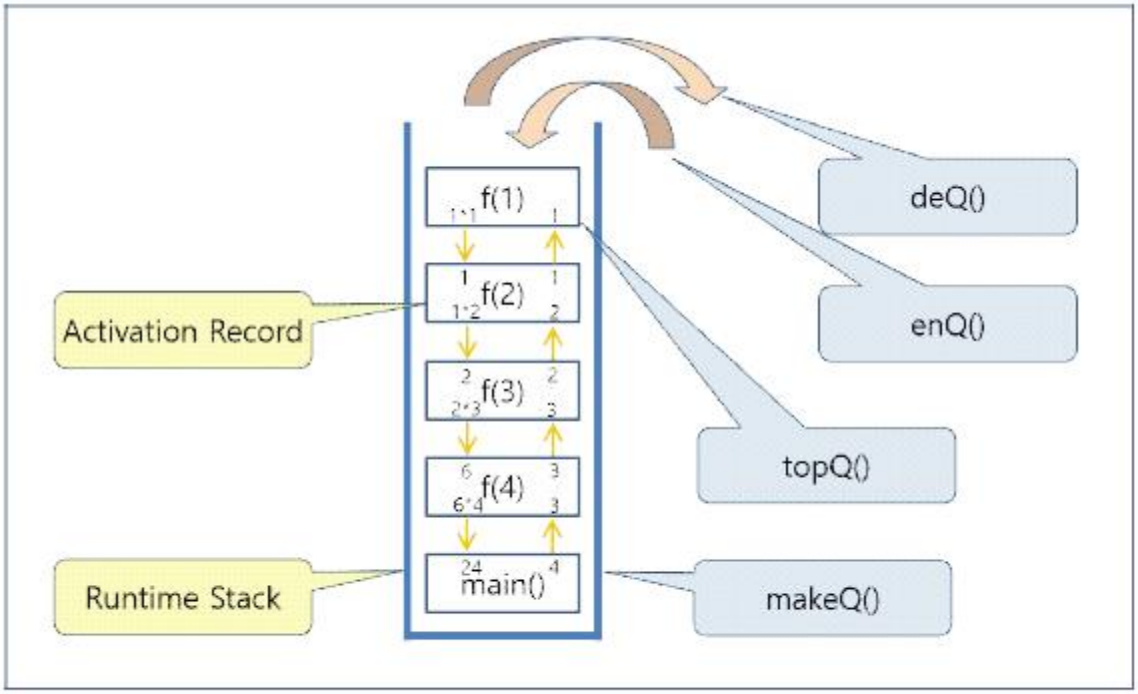
<그림 2-26> Macro Expansion - Renaming



<그림 2-27> Binding & Bound Occurrences

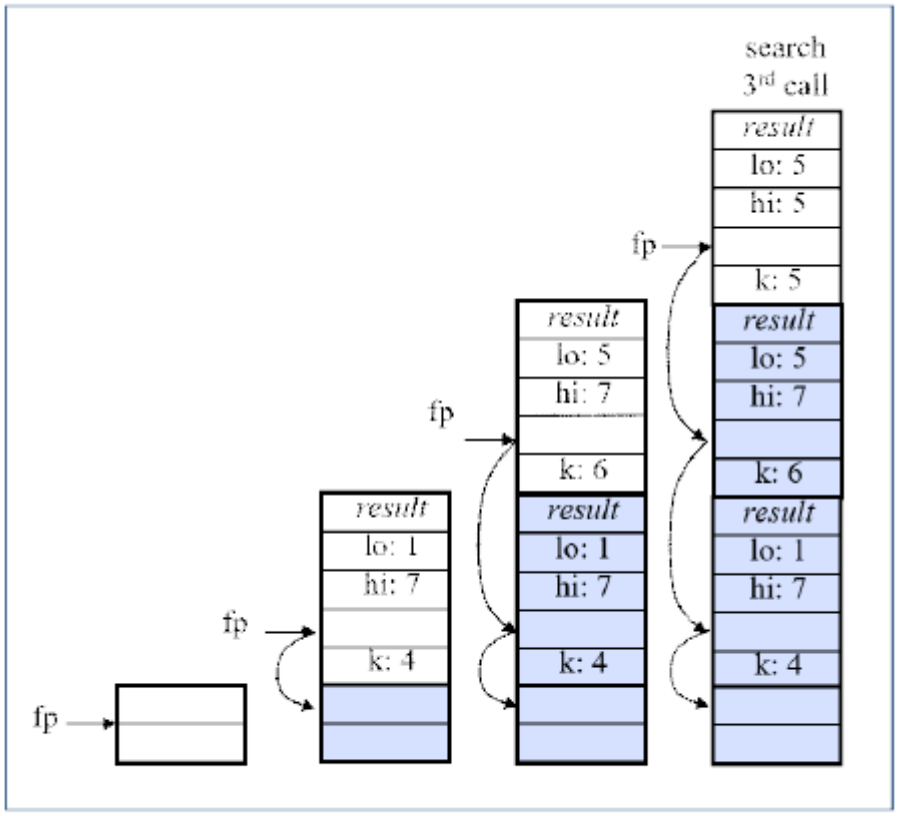


<그림 2-28> Scope Visibility

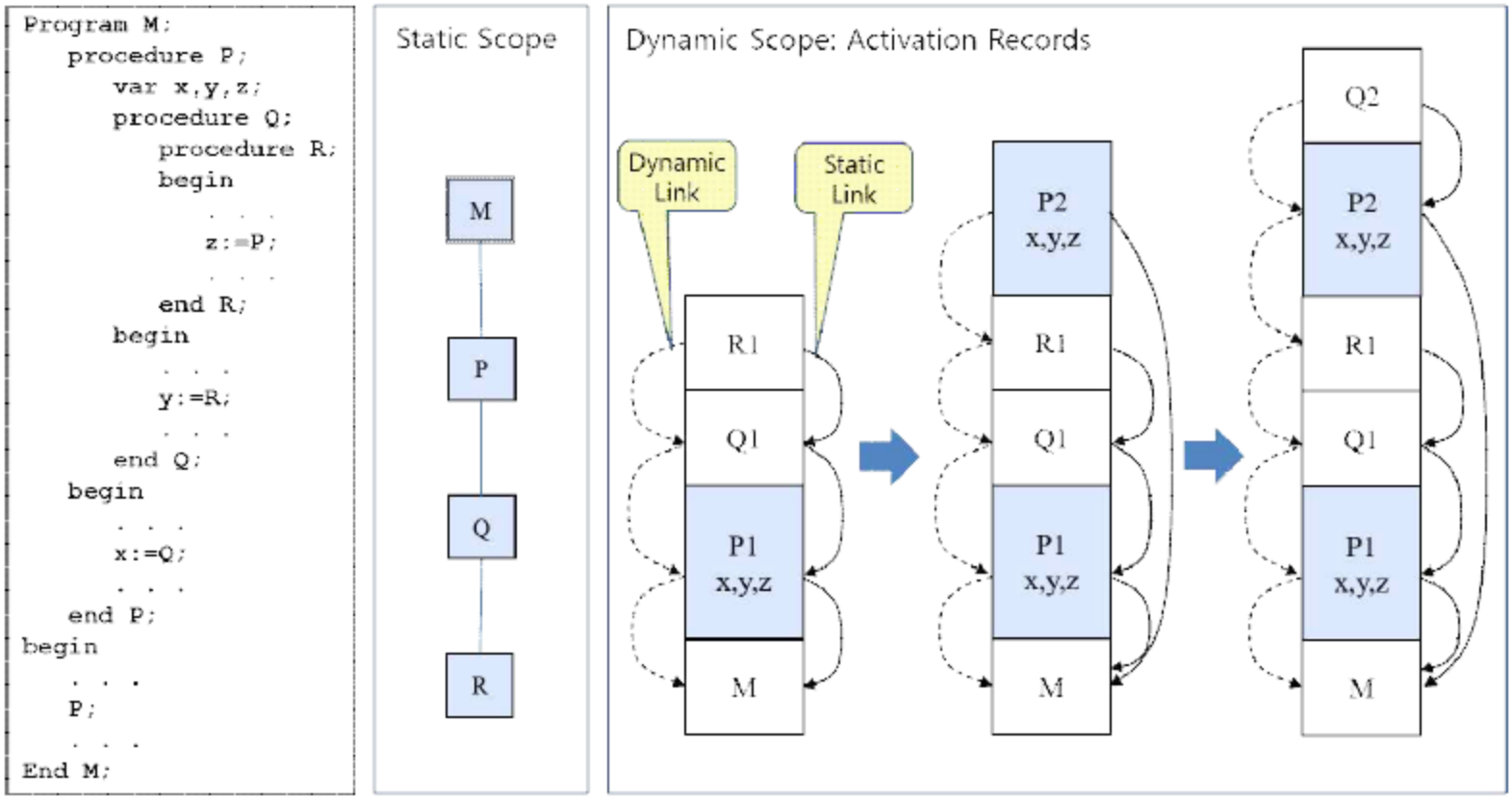


<그림 2-30> Runtime Stack: `factial(4)` 예제

```c
#include <stdio.h>
int yes =1, no=0;
#define N 7
int X[] = {0, 11, 22, 33, 44, 55, 66, 77};
int T;
int search(int lo, int hi)
{
   int k;
   if (lo>hi) return no;
   k=(lo+hi)/2;
   if (T==X[k]) return yes;
   else if (T < X[k]) return search(lo, k-1);
   else if (T > X[k]) return search(k+1, hi);
}

int main(void)
{
   scanf("%d", &T);
   if (search(1, N))
      printf("found\n");
   else
      printf("not found\n");
   return 0;
}
```



<그림 2-31> Activation Records



<그림 2-32> Activation Records : Pascal Example

2.5 결어