RENESAS

RVC Training for new engineers

Programming Exercise (3) Part 01

C Language pointer and address

Renesas Design Vietnam Co., Ltd.
Software Department

Hiroyuki Nakagawa

Rev. 2.33

Confidential

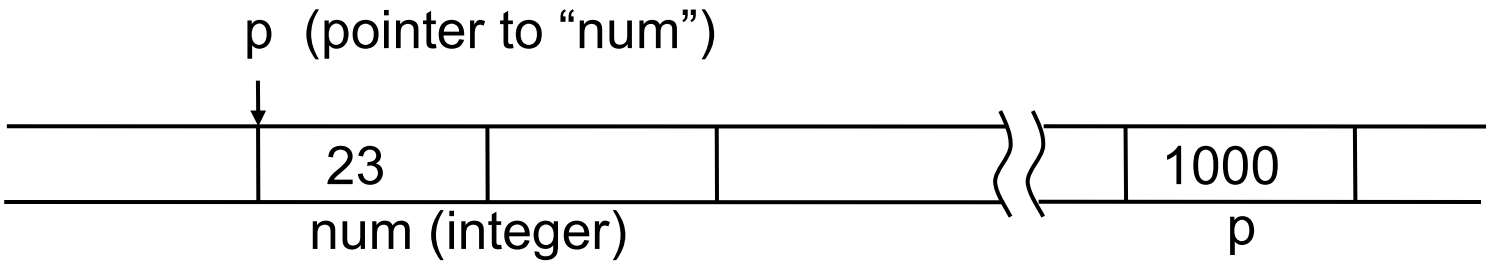
Pointer

A pointer is a variable which contains the address of another variable.

Sample program 1

```
int num;    /* integer */
int *p;     /* p is the pointer of integer */
num = 23;
p = &num;   /* set  num's address to p */
printf("%d  %d  %d  %d ", num, &num, *p, p);
```

Example: num's address is 1000



Sample result

```
23  1000  23  1000
```

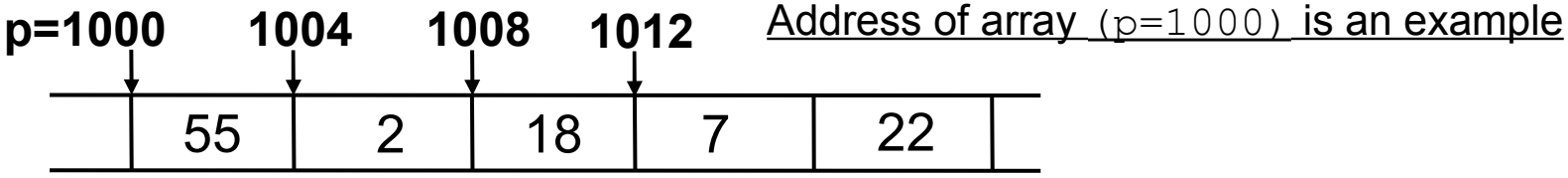
Programming Exercise 3

Pointer and array

Elements of an array can be handled by pointer.

Sample program 2-1

```
int array[5] = {55, 2, 18, 7, 22};/* integer array */
int *p; /* pointer p */
p = &array[0];  /* set  array's top address to p */
printf("%d    %d ",  *p,  *(p+4));
```



Address of array `(p=1000)` is an example

An array is a "pointer to the 0th element of the array".
`*p` is the same thing as `array[0]`, `*(p+1)` is same thing as `array[1]`,
`*(p+2)` is same thing as `array[2]` and so on.
Integer pointer is incremented by 4bytes. (in 32bit environment)

Sample result

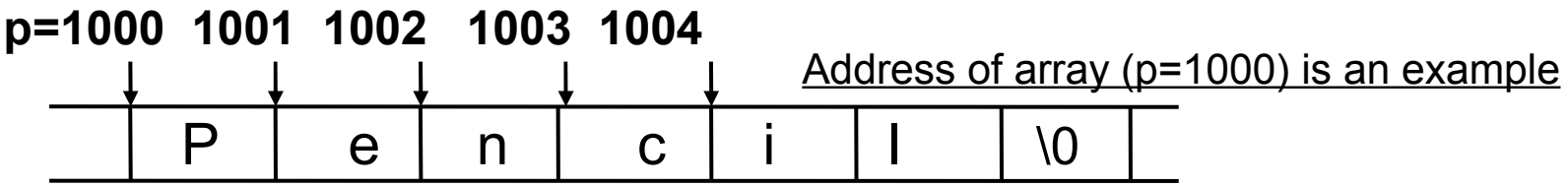
```
55   22
```

Programming Exercise 3

Character array (String)

Sample program 2-2

```
char array[] = "Pencil"; /* character array(=string)*/
char *p; /* pointer p */
p = &array[0]; /* set  array's top address to p  */
printf("%c    %c  \n",  *p,  *(p+4));
printf("%s\n", array);
```



An array is a "pointer to the 0th element of the array".
Character pointer is incremented by 1byte.

Sample result

```
P  i
Pencil
```

Programming Exercise 3

Character pointer and character array

A character pointer is similar to character array, but they are not same.

Sample program 3-1

```
char *str1;
char *str2 = "World";
str1 = str2;
printf( "%S\n", str1 );
```

Ok.
str1 is character pointer and "World" is an address of string.

Sample program 3-2 (Error)

```
char str1[6];
char *str2 = "World";
str1 = str2;
printf( "%S\n", str1 );
```

Compile error.
str1 is character array, and it cannot be input an address of string.

Correct cording is `strcpy(str1, str2);`

Sample result World

Programming Exercise 3

Exercise 1-1

Following program outputs the string "world".

Fill [a] in the program and complete the program.

Program

```
void main()
{
   char data01[ ] = "Hello
world";
   char *pData;
    _______[a]_____ ;
   printf( "%s\n", pData );
}
```

Result

```
world
```

Programming Exercise 3

Exercise 1-2

Following program calculate the distance of element of array.

(1) Fill the result of this program [b] and confirm by HEW simulater.

(2) Change "long" array to "character" array and compare results.

Program

```
void main()
{
    long  data02[16];
    long  *p1 = &data02[10];
    long  *p2 = &data02[5];
    printf( "%d   %d \n", p1, p2) );
    printf( "%d\n", (p1 - p2) );
}
```

When you change long to char, how the result change?

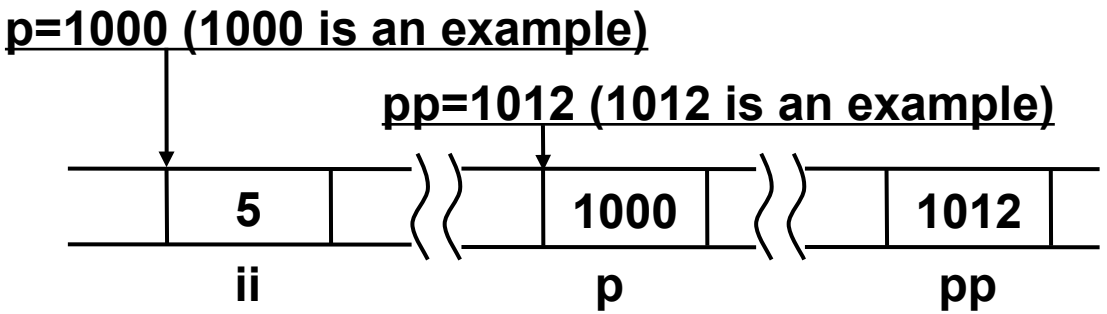
Result

```
[a]        [b]
[c]
```

Programming Exercise 3

Pointer's pointer

A pointer value is stored in somewhere in the memory and it has an address. A pointer's address is defined as pointer's pointer.



Sample program 4

```
int ii;
int **pp; /* pointer pointer */
int *p;   /* pointer */
ii = 5;
p = &ii;
pp = &p;
printf("%d\n", **pp);
```

An address of ii is set to p.

An address of p is set to pp.

**pp is focused to value of ii.

Sample result

```
5
```

Programming Exercise 3

Exercise 1-3

Fill [a] of program Ex2-1

program Ex2-1

```
int ii,jj,kk;
int **pp;  /* pointer pointer */
int *p[3];  /* pointer array */
ii = 5;
jj = 7;
kk = 9;
p[0] = &ii;
p[1] = &jj;
p[2] = &kk;
pp = p;
**(pp+2) = **pp * **(p+1);
printf("%d\n", kk);
```

Result

Programming Exercise 3

Pointer for function arguments

Sample program 5-1(Bad sample)

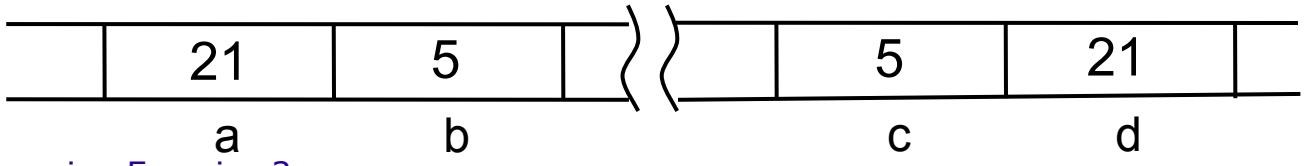
```
void intswap(int c, int d){
 int tmp;
 tmp = c;
 c = d;
 d = tmp;
}
int main(void) {
 int a = 21;
 int b = 5;
 printf("Before:%d  %d",  a,  b);
 intswap( a , b );
 printf("Afte r:%d  %d",  a,  b);
}
```

Sample result

```
Before: 21  5
After :  21  5
```

I want to change value in a and b by "intswap".(a: 21->5 , b: 5 ->21)
But left program (intswap) cannot change two values. Why?

Because arguments are always passed by value in C function calls.
This means that "local copies" of the values are passed to the subroutines.
Any change made to the arguments internally in the subroutine are made only to the local copies of the arguments.



Programming Exercise 3

Pointer for function arguments

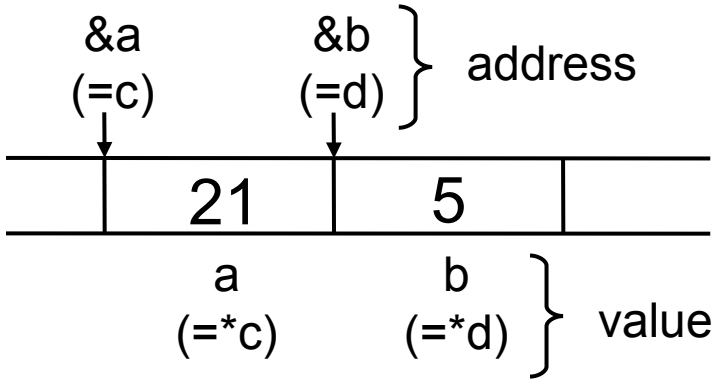
Sample program 5-2

```
void intswap(    int* c, int* d     ) {
 int tmp;
 tmp = *c;
 *c = *d;
 *d = tmp;
}
int main(void) {
 int a = 21;
 int b = 5;
 printf("Before:%d  %d",  a,  b);
 intswap(        &a, &b          );
 printf("After :%d  %d",  a,  b);
}
```

Sample result

```
Before:21  5
After :  5  21
```

In order to get back value from subroutine, argument must be passed as a value's address, and show same address in subroutine with main routine.



Programming Exercise 3

Exercise 1-4

program Ex1-4

```
int main(void) {
    int a = 21;
    int b = 5;
    printf("%d  %d",  a,  b);
    magic01(     [ a ]      );
    printf("%d  %d",  a,  b);
}
```

Result

```
21  5
26  16
```

*Fill **[a]** and make the function magic01.*

Note,
Subroutine `magic01()` get 2 parameters from `main()`.
When returns, 2 parameters are changed to their sum and sub.
21 + 5 = 26
21 – 5 = 16

Pointer for function arguments

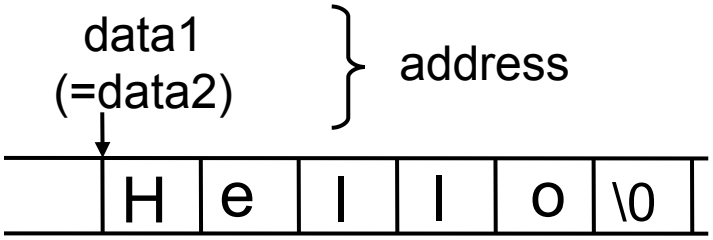
Sample program 5-3

```
void sub01(char *data2 ) {


    strcpy(data2, "Hello");
}

int main(void) {
   char data1[16];
   sub01( data1 );
   printf("%s\n", data1);
}
```

This sample function use the character pointer for argument.

String is declared in main routine, and it has 16byte area.

"sub01" is called with string address and data"Hello" is stored in the address.



Sample result

```
Hello
```

Programming Exercise 3

Exercise 2-1

Following program changes small letter string to capital letter string.

Example:
Input :hello
Output :HELLO

(1) Elemental exercise
Fill [a] [b] [c] [d] and complete the program. When you fill [C] , don't use array expression.
Function StoC had already prepared which changes one small letter to capital letter.

(2) Additional exercise
Function StoC can process 'a' to 'z'. But it doesn't check any other invalid input. Add data check cording and change for better cording.

```
char StoC( [a]  )  /* Change one char */
{
  return (c - 32); /* Only alphabet */
}
void changeString(char *data)
{
  int i;
  for (i = 0;   [b]    != '\0'; i++) {
      printf("%c",  StoC(   [c]   ));
  }
  printf("\n");
}
void main()
{
  char data01[32];
  scanf("%s",data01);
  changeString(     [d]     );
}
```

Programming Exercise 3

Ascii code table

Dec	Hex	Sym	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	NUL	32	20		64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	TAB	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	¥	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	DEL

Meaning of invisible code

Name	Dec	Hex	Description
NUL	0	00	null
SOH	1	01	start of heading
STX	2	02	start of text
ETX	3	03	end of text
EOT	4	04	end of transmission
ENQ	5	05	enquiry
ACK	6	06	acknowledge
BEL	7	07	bell
BS	8	08	backspace
TAB	9	09	horizontal tab
LF	10	0A	line feed
VT	11	0B	vertical tab
FF	12	0C	form feed
CR	13	0D	carriage return
SO	14	0E	shift out
SI	15	0F	shift in
DLE	16	10	data link escape
DC1	17	11	device control 1
DC2	18	12	device control 2
DC3	19	13	device control 3
DC4	20	14	device control 4
NAK	21	15	negative acknowledge
SYN	22	16	synchronous idle
ETB	23	17	end of trans. block
CAN	24	18	cancel
EM	25	19	end of medium
SUB	26	1A	substitute
ESC	27	1B	escape
FS	28	1C	file separator
GS	29	1D	group separator
RS	30	1E	record separator
US	31	1F	unit separator
DEL	127	7F	Delete

Programming Exercise 3

Exercise 2-2

Make `my_strlen()` function with using character pointer.
This function get character pointer(=string) and return the string length.

Function format : `int my_strlen (char*);`

How to use `strlen()`

```
void main()
{
    char *strdata = "This is a test";
    int leng;
    leng = my_strlen(strdata);
    printf ("%d\n", leng);
}
```

Result

```
14
```

Programming Exercise 3

Exercise 2-3

Make `my_strstr()` function with using character pointer.

Function format :

```
char* my_strstr(char* s1,
char* s2);
```

This function search string `s2` in the string `s1`, and when it is found, return the address of find point.
When it is not found, return NULL;

How to use `strstr()`

```
void main()
{
  char *my_strstr(char*, char*);
  int leng;
  char *s1 = "abcdefghijk";
  char *s2 = "cdef";
  char *s3;
  s3 = my_strstr(s1, s2);
  printf ("%s\n", s3);
}
```

Result

```
cdefghijk
```

Programming Exercise 3

Structure pointer

Sample program 6

```
struct exam
{
    char name[32];
    int  science;
    int  english ;
};
void put_result(  struct exam *pEx   )
{
    printf("%s  %d  %d\n",
   pEx->name, pEx->science, pEx->english );
}
void main(void)
{
    struct exam ex1;
    strcpy(ex1. name, "Nguyen");
    ex1.science = 96;
    ex1.english = 88;
    put_result( &ex1 );
}
```

Sample result

```
Nguyen  96  88
```

`pEx` is structre exam's pointer

How to use structure member by structure pointer.

Declare structure exam as `ex1`

Set data to char array member

Set data to integer member

Call sub. `&ex1` is `ex1`'s address

Programming Exercise 3

Function pointer

Sample program 7

```
int f_sum(int c, int d) {
    int r;
    r = c + d;
    return (r);
}

int main(void) {
   int (*p) (int, int);
   int a = 2;
   int b = 5;
   int ans;
   p = f_sum;
   ans = (*p)(a, b);
   printf("%d",  ans);
}
```

Declare function pointer p

Set "f_sum" function pointer to p

Call function by function pointer.
It is same meaning as "ans = f_sum(a, b);"
You can write "ans = p(a, b);"

Sample result

```
7
```

Function also has an address. Function address is called "Entry point".

Programming Exercise 3

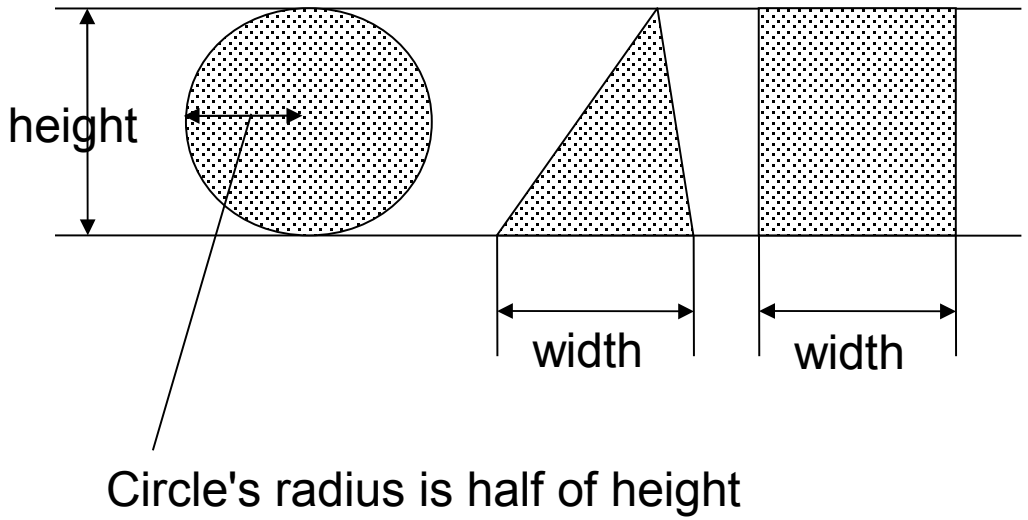
Function pointer array

Sample program 8 show you how to use a function pointer array.

This program get height and width of figure, and calculate area size of figure, when figure is circle, triangle, rectangle.

Sample result

```
Input height and width:6  4
28
12
24
```



Programming Exercise 3

Function pointer array

Sample program 8

```
/* Define prototype */
int circle( int , int);
int triangle( int, int);
int rectangle( int, int);

void getAreaSize(void)
{
   int (*p[3]) (int , int ) = {circle , triangle , rectangle};
   int num1, num2;
   int i;

   printf( "%s","Input height and width" );
   scanf( "%d %d", &num1, &num2 );

   /* Access to the function pointer array */
   for(i=0; i<3; ++i)
   {
      printf( "%d\n", (*p[i])(num1, num2) );
   }
}
```

```
int circle(int height, int nouse)
{
   return ( (height / 2) * (height / 2) * 3 );
}
int  triangle(int height, int width)
{
   return (height * width / 2);
}
int rectangle(int height, int width)
{
   return (height * width);
}
```

Declare function pointer array and initialize with 3 subroutine name

Call subroutine with function pointer

Programming Exercise 3

Exercise 3

Following program is similar to sample program 8.

This program get structure pointer, height and width of figure, and calculate area size of figure.

The structure stores figure kind, height and width.

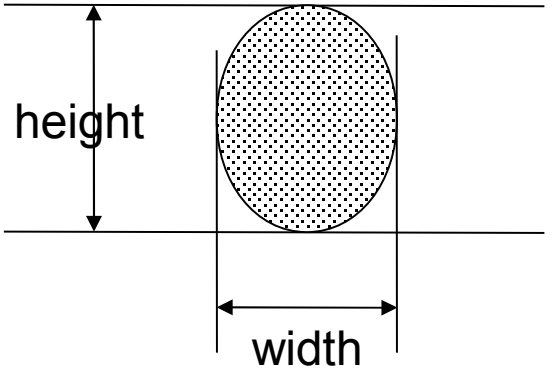
(1) Fill blank of this program and complete it.

And test the coding by HEW simulator.

(2) Add function for get area size of oval to the Exercise 3 program.

Note1: The figure kind number of oval is 3.

Note2: Area size of oval is (height/2 * width/2 * 3).



(3) Make new program that calculate

circumference of specified figure.

Note1: Use structure pointer for parameter.

Note2: Call subroutine by function pointer.

Exercise 3 program

```
/* Define prototype */
int circle (int , int);
int triangle( int , int);
int rectangle( int , int);
int (*pFunc[3]) (int, int) = {circle, triangle, rectangle};

struct fig {
    int kind;   /* Circle=0, Triangle=1, Rectangle=2 */
    int height;  /* Height of figure */
    int width;   /* Width of figure  */
} ;

int getAreaSize2(          [a]          )
{
    int s;
    s =   (     [b]     )(   [c]   ,   [d]   ) );
    return s;
}

int circle (int height, int nouse)
{
   return ( (height/2) * (height/2)  * 3 );
}
```

```
int  triangle (int height, int width)
{
   return ( height * width / 2 );
}
int rectangle (int height, int width)
{
   return (height * width );
}

void main()
{
    struct fig f1;
    int k,h,w;
    printf("Input figure kind:");
    scanf("%d", &k);
    printf("Input  height  width:");
    scanf("%d %d", &h, &w);
    f1.kind = k;
    f1.height = h;
    f1.width = w;
    printf("%d\n", getAreaSize2(   [e]   ));
}
```

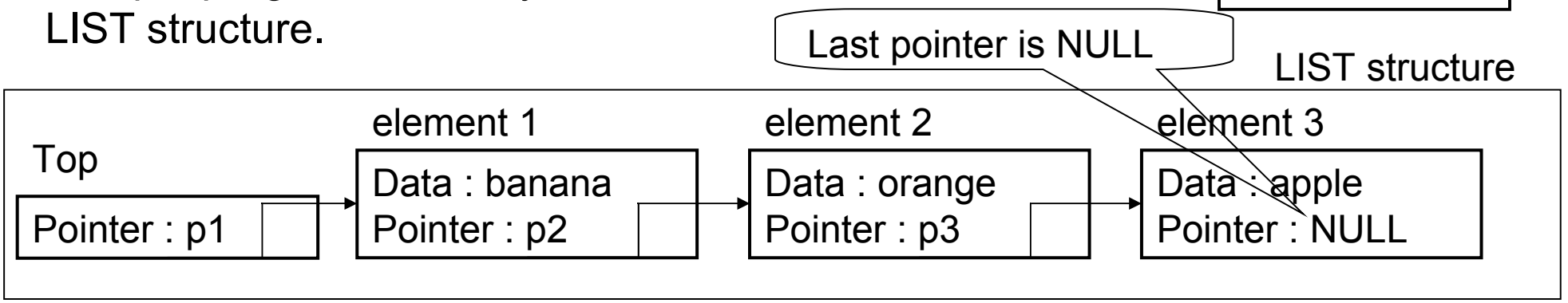
Use structure pointer

LIST

LIST is a set of jointed data elements.
One element has data and the pointer to next element.
Sample program 9 show you how to use LIST structure.

Array data

0	apple
1	orange
2	banana



Sample program 9 action

(1) Creates the LIST from array data
(2) Print the LIST
(3) Delete the LIST.
Sample data are fruit names.
There are 3 fruits in the LIST.

One element of LIST

```
struct leaf {
   char data[16];     /* string data */
   struct leaf *next; /* pointer to next*/
};
```

Programming Exercise 3

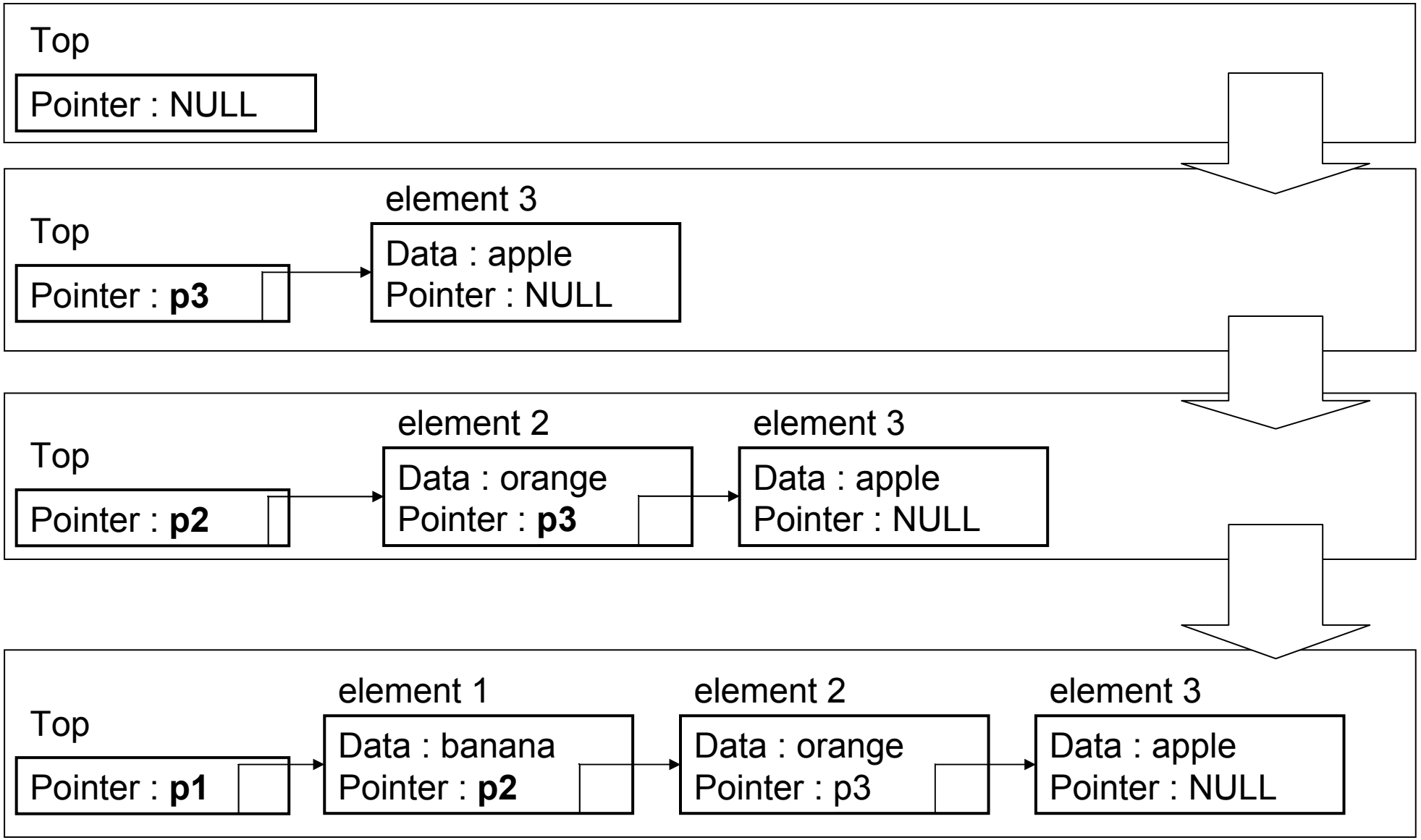
Sample program 9

```
struct leaf {
   char data[16];     /* string data */
   struct leaf *next; /* pointer to the next leaf */
};
struct leaf *addLeaf(char*, struct leaf *);
void printLeaves(struct leaf *p);
void freeLeaves(struct leaf *p);

int main(void)
{
    struct leaf *top;   /* top of the leaves */
    char *fruits[3] = {"apple", "orange", "banana" } ;
    top = NULL;
    top = addLeaf(fruits[0], top);
    top = addLeaf(fruits[1], top);
    top = addLeaf(fruits[2], top);
    printLeaves(top);   /* print elements*/
    freeLeaves(top);   /* delete all elements */
}

void printLeaves(struct leaf *pL)
{
    while (pL != NULL) {
        printf("%s\n",  pL->data);
        pL = pL->next;
    }
}
```

Programming Exercise 3

```
struct leaf *addLeaf(char *data, struct leaf *top)
{
    struct leaf *pL;

    pL = (struct leaf*) malloc(sizeof(struct leaf));
    strcpy(pL->data, data);

    /* change pointer to the top */
    pL->next = top; /* Old top is new leaf's next    */
    top = pL;/* New top is new leaf's address */
    return top;
}

void freeLeaves(struct leaf *pL)
{
    struct leaf *pLnext;

    while (pL != NULL) {
        pLnext = pL->next;
        free(pL);
        pL = pLnext;
    }
}
```

List create step (addLeaf)



Programming Exercise 3

Confirmation

Regarding the sample program 9, add following function.

(1) After create initial 3 data (prepared in program 9),

insert element 4 after the top. (element 4 data is "cherry")

Exercise 4-1

Regarding the sample program 9, add following function.

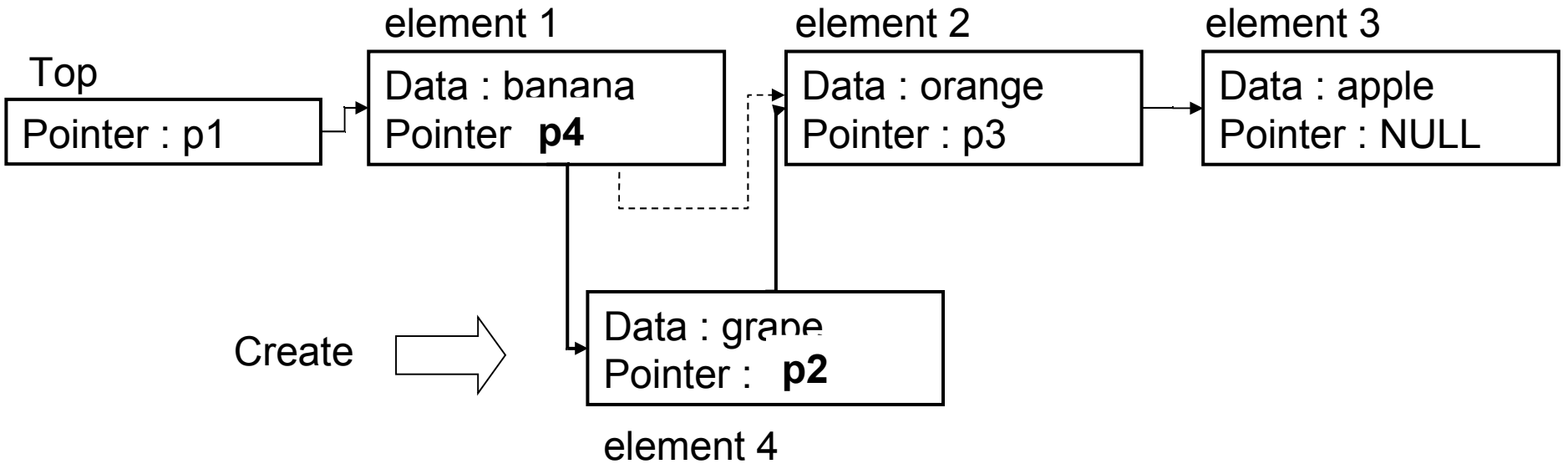
(1) After creating 3 initial data (prepared in program 9),

insert element 4 after element 1. (element 4 data is "grape")

[Expand exercise]
(2) Get “insert number” and “fruit name” by "scanf", and insert the element to the LIST.

Add new element to the LIST

LIST structure is more easy to add element than array.



How to add new element 4

(1) Create element4 data.
(2) Copy element1's next pointer to element4's next pointer.
(3) Set element4 's address to the element1's next pointer.

Programming Exercise 3

Exercise 4-2

Regarding the sample program 9, add following function.

(1) After create initial 3 data (prepared in program 9),

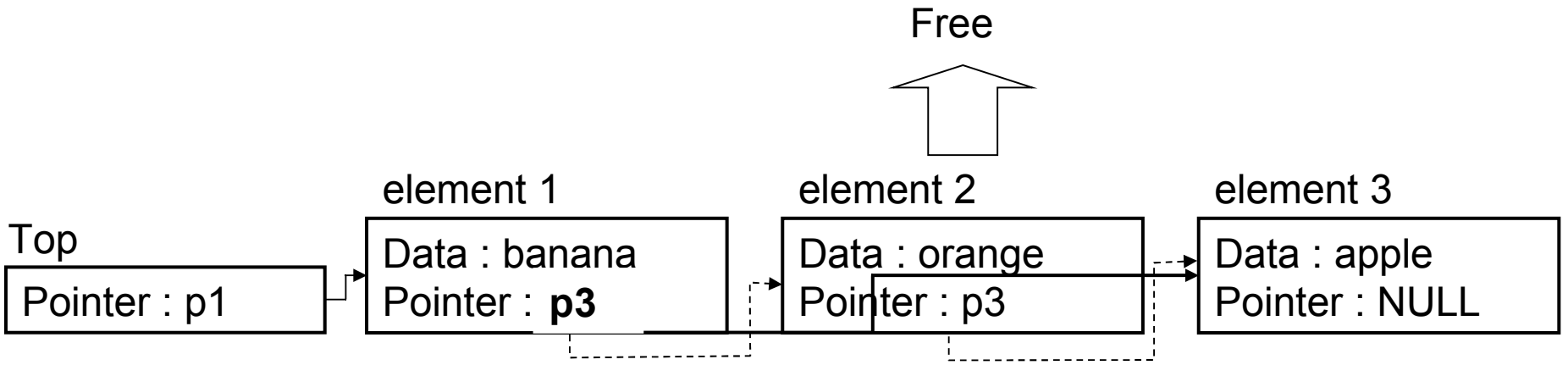
remove element 2.

[Expand exercise]

(2) Get “remove number” by "scanf", and remove the element from the LIST.

Remove element from the LIST

LIST structure is more easy to remove element than array.



How to remove element 2

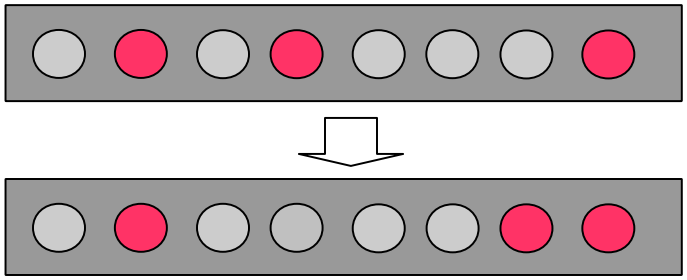
(1) Copy element2's next pointer to element1's next pointer.

(2) Free memory the data of element 2.

Programming Exercise 3

Definition of direct address (Optional)

There is a board with 8 LED.
0x003E0 is address of LED register.
Each bit of register control one LED.
I want to light on LED as follows.

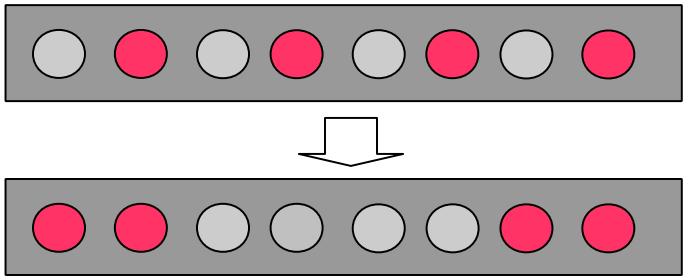


```
#define PORT_LED   *(volatile unsigned char *)0x003E0

void main(void)
{
    PORT_LED = 0x51;
    PORT_LED = 0x43;
}
```

What is volatile means?

Exercise 5

There is a board with 8 LED.
0x003E0 is address of LED register.
Each bit of register control one LED.
I want to light on LED as follows.



```
#define PORT_LED   (volatile unsigned char *)   0x003E0

void main(void)
{
        [a]             ;
        [b]             ;
}
```

Programming Exercise 3

Exercise 6

Make your original program and confirm that function pointer can be use in function parameter.(Find any effective case.)
Say in other words,
(1) Make the main routine and any subroutines.
(2) Declare function pointer in the main routine.
(3) Call subroutine1 which has function pointer argument.
(4) Call subroutine2 by function pointer.

Q&A

Renesas Design Vietnam Co., Ltd.