프로그래밍 나두 할 수 있다!
I CAN DO PROGRAMMING

•••

네번째 모임
~Fourth Meeting~

네번째 모임 주제

- 워밍업 문제
  - Precedence
  - Third meeting recap
- String and array
  - Strlen
  - Sizeof
- ASCII 코드를 활용해 소문자 대문자 바꿔봅시다
- Argc/ argv
- Cryptography
- Pointer
  - *
  - &

Operator	Description	Associativity
() [] . -> ++ --	Parentheses or function call Brackets or array subscript Dot or Member selection operator Arrow operator Postfix increment/decrement	left to right
++ -- + - ! ~ (type) * & sizeof	Prefix increment/decrement Unary plus and minus not operator and bitwise complement type cast Indirection or dereference operator Address of operator Determine size in bytes	right to left
* / %	Multiplication, division and modulus	left to right
+ -	Addition and subtraction	left to right
<< >>	Bitwise left shift and right shift	left to right
< <= > >=	relational less than/less than equal to relational greater than/greater than or equal to	left to right
== !=	Relational equal to or not equal to	left to right
&&	Bitwise AND	left to right
^	Bitwise exclusive OR	left to right
|	Bitwise inclusive OR	left to right
&&	Logical AND	left to right
||	Logical OR	left to right
?:	Ternary operator	right to left
= += -= *= /= %= &= ^= |= <<= >>=	Assignment operator Addition/subtraction assignment Multiplication/division assignment Modulus and bitwise assignment Bitwise exclusive/inclusive OR assignment	right to left
,	comma operator	left to right

```c
#include <stdio.h>

int main(void){
    int x = 12;
    int y = 5;

    x *= y + 1;

    printf("%d", x);
}
```

Q. 결과값이 무엇일까요? 그리고 그 이유는 무엇일까요?

```c
#include <stdio.h>
#include <cs50.h>

float sum(int length, int arr[]);

int main(void){
    const int TOTAL = 3;

    int i = 0;
    int score[TOTAL];
    score[i] = get_int("score: ");
    i++;

    while (score[i] >= 0){
        score[i] = get_int("score: ");
        i++;
    }


    printf("%f\n", sum(i, score));
}

float sum(int length, int arr[]){
    int sum = 0;
    int i;
    for(i = 0; i < length; i++){
        sum += arr[i];
    }
    return sum / (float) length;
}
```

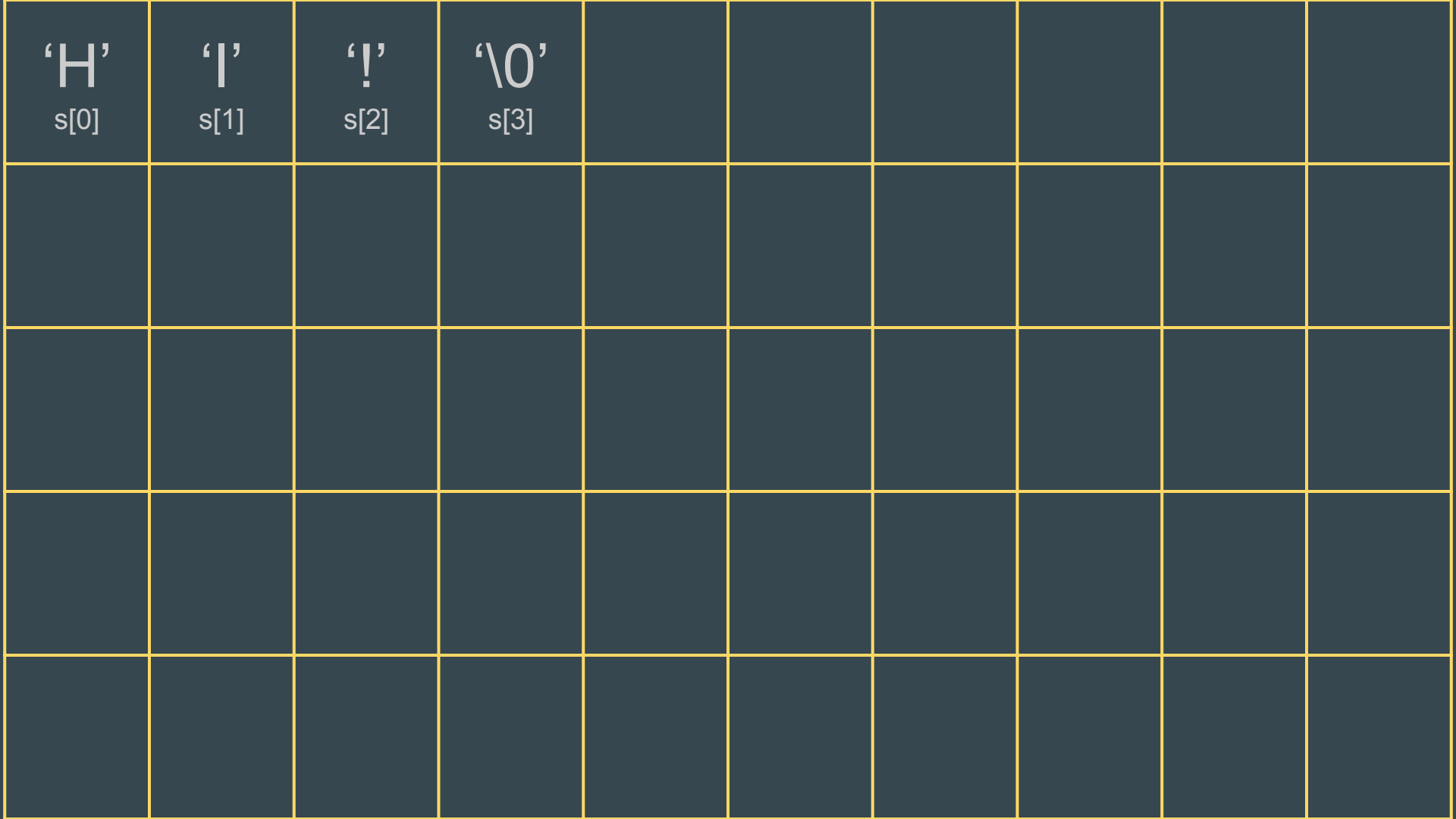
Q. return type? Void? Function 이름?

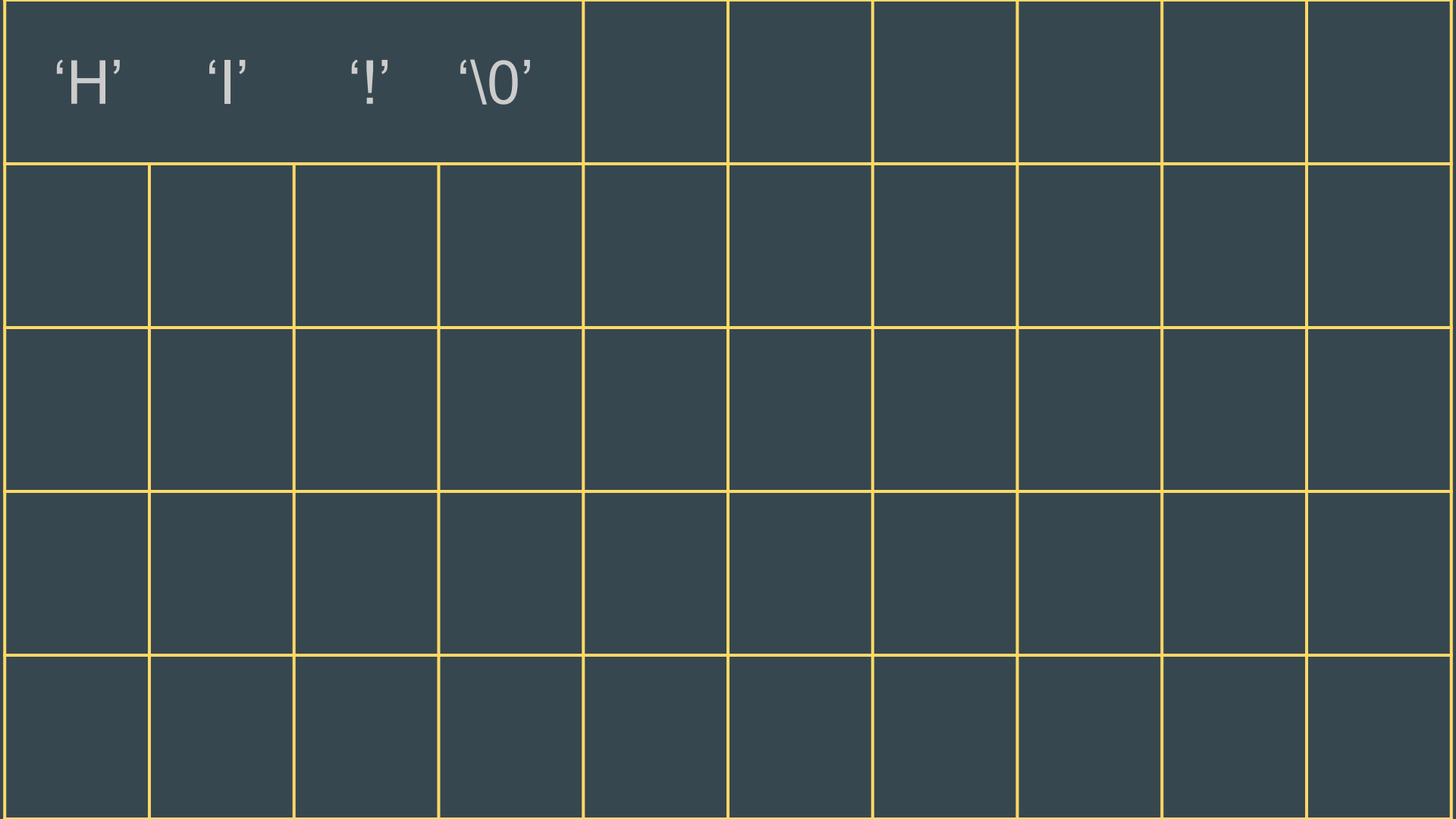
```c
#include <stdio.h>

int resu;

void sum(int x, int y){
    resu = x + y;
}

int main(){
    int a = 3;
    int b = 2;
    sum(a,b);
    printf("The sum is %d", resu);

}

```

Q1. glabal variable? (글로벌 변수)

Q2. void임에도 불구하고 main function이 sum function에서 값을 받아올 수 있는 이유는 뭘까요?

STRING


‘H’
s[0]
‘I’
s[1]
‘!’
s[2]
‘\0’
s[3]


‘H’ ‘I’ ‘!’ ‘\0’

72 s[0]	73 s[1]	33 s[2]	0 s[3]						

‘H’	‘I’	‘!’	‘\0’	‘B’	‘Y’	‘E’	‘!’	‘\0’	

‘H’ s[0]	‘I’ s[1]	‘!’ s[2]	‘\0’ s[3]	‘B’ w[0]	‘Y’ w[1]	‘E’ w[2]	‘!’ w[3]	‘\0’ w[4]	

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com


input
Algorithm
output

Cryptography


128√e980
ILove980
Posted at
Espangrish.com


1
I love you
Cipher
Cipher text

I love you

73 108 111 118 101 121 111 117

74 108 111 118 101 121 111 117

74 109 111 118 101 121 111 117

74 109 112 118 101 121 111 117

74 109 112 119 101 121 111 117

74 109 112 119 102 121 111 117

74 109 112 119 102 122 111 117

74 109 112 119 102 122 112 117

74 109 112 119 102 122 112 118

J m p w f z p v

cryptography

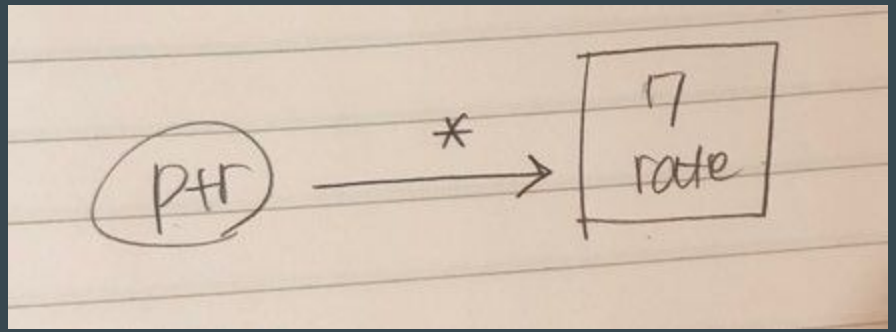
왜 중요할까요?

이메일

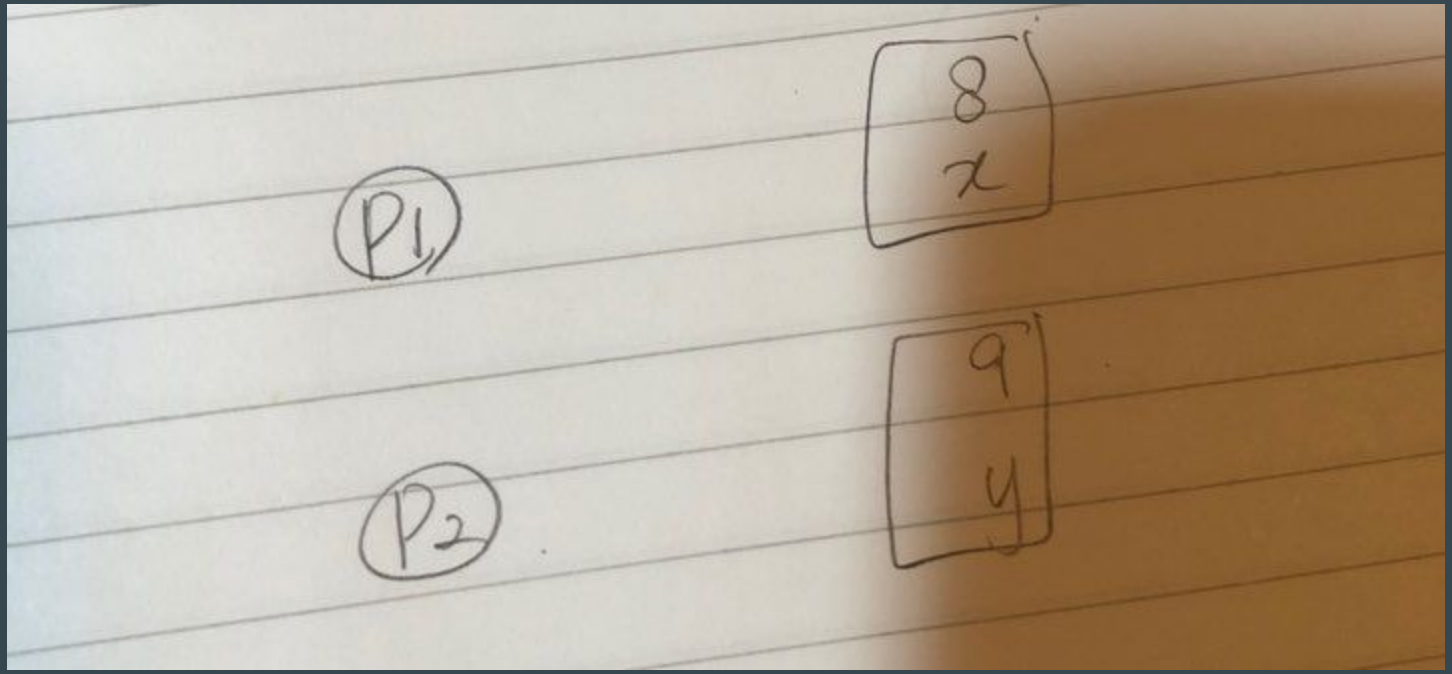
개인정보 보호

중요 작업

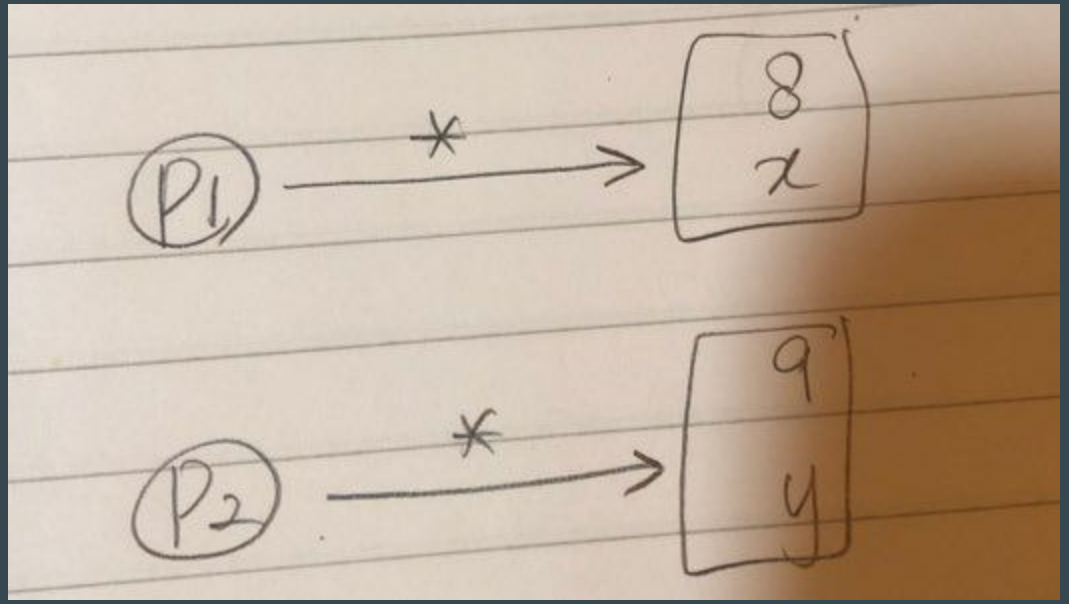
pointer



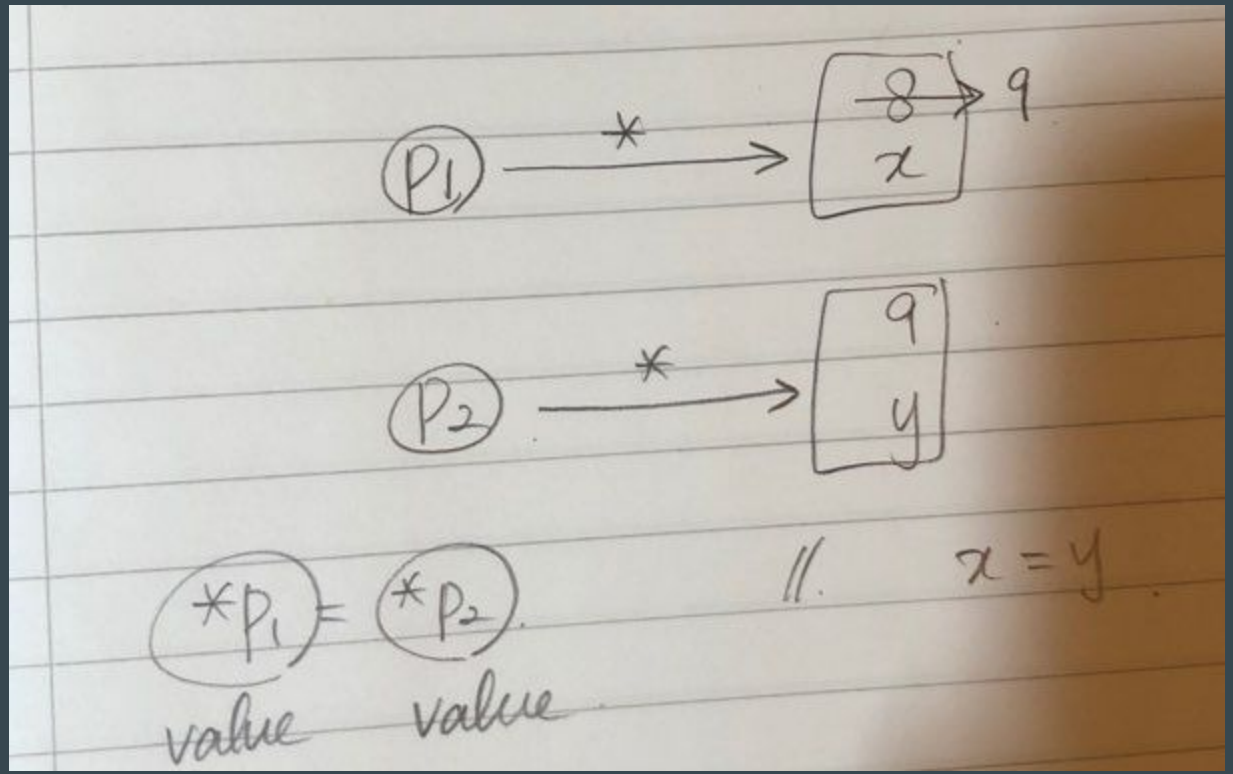
pointer



pointer



pointer



pointer