프로그래밍 연습

실습 #3

과제 풀이

실습

실습1

연산자 우선 순위를 코드 실행으로 직접 확인하기 위한 예제

```c
#include <stdio.h>

int main(void)
{
double a = 1, b = 2, c = 3, d = 4;
printf("a = %lf, b = %lf, c = %lf, d = %lf\n\n", a, b, c, d);

printf("a*b/c = %lf\n", a*b/c);
printf("a*b/c = %.3lf\n\n", a*b/c);

printf("1+a*b = %lf\n", 1+a*b);
printf("1+a*b = %03.0lf\n\n", 1+a*b);

printf("a = %2.1lf, b = %2.1lf, c = %2.1lf, d = %2.1lf\n", a, b, c, d);
printf("++a*b-c-- = %lf\n", ++a*b-c--);
printf("a = %2.1lf, b = %2.1lf, c = %2.1lf, d = %2.1lf\n", a, b, c, d);

return 0;
}

```

실습1

➢ 코드분석

```
double a = 1, b = 2, c = 3, d = 4;
```

- 실수형 변수 a, b, c, d를 선언하고, 초기값 설정

```
printf("a = %lf, b = %lf, c = %lf, d = %lf₩n₩n", a, b, c, d);
```

- 변수 a, b, c, d 에 할당된 값 출력

✓ 출력 결과

```
a = 1.000000, b = 2.000000, c = 3.000000, d = 4.000000
```

실습1

➢ 코드분석

```
printf("a*b/c = %lf₩n", a*b/c);
```

- *, / 연산자 우선 순위가 같으므로 associativity에 따라 왼쪽 부터 계산

```
printf("a*b/c = %.3lf₩n₩n", a*b/c);
```

- 위 코드와 연산은 동일. %.3f format 으로 출력 (소수점 3자리까지)

✓ 출력 결과

```
a*b/c = 0.666667
a*b/c = 0.667
```

Operator precedence and associativity	
Operator	Associativity
() ++ *(postfix)* -- *(postfix)*	left to right
+ *(unary)* - *(unary)* ++ *(prefix)* -- *(prefix)*	right to left
* / %	left to right
+ -	left to right
= += -= *= /= etc.	right to left

실습1

➢ 코드분석

```
printf("1+a*b = %lf₩n", 1+a*b);
```

- 연산자 우선 순위가 다르므로 우선 순위가 높은 * 부터 계산

```
printf("1+a*b = %03.0lf₩n₩n", 1+a*b);
```

- 위 코드와 연산은 동일. %03.0f format 으로 출력
  (전체 3자리를 유지하고 빈 공간은 0으로 표기, 소수점 표기하지 않음)

✓ 출력 결과

```
1+a*b = 3.000000
1+a*b = 003
```

Operator precedence and associativity	
Operator	Associativity
() ++ *(postfix)* -- *(postfix)*	left to right
+ *(unary)* - *(unary)* ++ *(prefix)* -- *(prefix)*	right to left
* / %	left to right
+ -	left to right
= += -= *= /= etc.	right to left

실습1

➢ 코드분석

```
printf("a = %2.1lf, b = %2.1lf, c = %2.1lf, d = %2.1lf₩n", a, b, c, d);
```

- 변수 a, b, c, d를 소수 첫째자리까지 출력

```
printf("++a*b-c-- = %lf₩n", ++a*b-c--);
```

- 연산자 우선 순위에 따라 ((++a)*b)-(c--) 와 같은 순위로 계산

```
printf("a = %.1lf, b = %2.1f, c = %2.1lf, d = %2.1lf₩n", a, b, c, d);
```

- ++, -- 연산자에 의해 변경된 변수 값을 출력

✓ 출력 결과

```
a = 1.0, b = 2.0, c = 3.0, d = 4.0
++a*b-c-- = 1.000000
a = 2.0, b = 2.0, c = 2.0, d = 4.0
```

Operator precedence and associativity

Operator	Associativity
() ++ *(postfix)* -- *(postfix)*	left to right
+ *(unary)* - *(unary)* ++ *(prefix)* -- *(prefix)*	right to left
* / %	left to right
+ -	left to right
= += -= *= /= etc.	right to left

실습2

문제
두 자리의 자연수를 입력 받아 각 자릿수의 합을 출력하세요.

입력
첫째 줄에 두 자릿수 자연수 n이 주어진다. (10 <= n <= 99)

출력
각 자릿수의 합을 출력한다.

테스트케이스

입력	출력
10	1
55	10
92	11

실습3

문제
Char형을 이해하기 위한 예제

```c
#include <stdio.h>

int main(){
  char c1 = '3';
  char c2 = 'b';
  printf("%c, %c\n", c1, c2);
  printf("%d, %d\n\n",c1, c2);

  int n = c1 - 48;
  c2 = c2 - 32;

  printf("%d\n", n);
  printf("%c\n", c2);

  return 0;
}
```

실습3

참고) 아스키코드 테이블

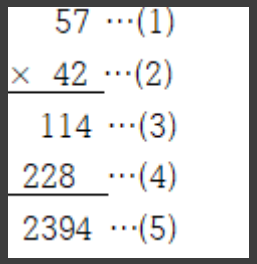
ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

과제

과제1 (파일명 : hw1.c)

문제.
두 자리의 자연수 곱셈은 다음과 같이 이루어집니다.



(1), (2)가 주어질 때 (3), (4), (5)를 출력하세요.

입력 : 첫째 줄에 두 자리의 자연수 a, b (10 <= a, b <= 99)
출력 : 각 줄에 (3), (4), (5)에 해당되는 자연수

테스트케이스

입력	출력
57 42	114 228 2394

과제2 (파일명 : hw2.c)

문제. 아주 간단한 암호
숫자로 된 암호가 주어졌을 때 이를 해독하기 위해 숫자의 순서와 대응되는 소문자 알파벳으로 변환 후 오른쪽으로 다섯 칸 shift 합니다. a의 오른쪽은 b, b의 오른쪽은 c, …, z의 오른쪽은 a입니다. 이때 주어지는 암호는 하나의 소문자 알파벳과 대응합니다. (ex : 11이 주어지면 이는 ff가 아닌 p입니다.)

입력 : 첫째 줄에 암호 자연수 n (1 <= n <= 26)
출력 : 해독 된 소문자 알파벳

테스트케이스

입력	출력
11	p
1	f
26	e