프로그래밍	**5. 자료형에 따른 표현방식, 그리고 형변환**	일자: 2024. . .
학습 목표	☐ 자료형에 따른 표현방식 ☐ 묵시적 형변환과 명시적 형변환 개념 학습	하이디:

1. 자료형에 따른 표현방식

1) 자료형 복습

종류	형식 이름	메모리 크기	값의 범위
문자형 변수	char	1 Byte	-128 ~ 127 / 0 ~ 255
정수형 변수	int	4 Byte	-2,147,483,648 ~ 2,147,483,647
	short	2 Byte	-32,768 ~ 32,767
	long long (int)	8 Byte	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807
실수형 변수	float	4 Byte	3.4 * 10 ^ (-38) ~ 3.4 * 10 ^ (38)
	double	8 Byte	1.7 * 10 ^ (-308) ~ 1.7 * 10 ^ (308)

2) 문자형

- 상기 표에서 char 자료형의 범위가 정수로 표현된 것을 확인할 수 있다.
- 이 말인즉슨, 이러한 숫자에 문자를 대응시켜 문자 하나를 표현하고 있다는 뜻이다.

① ASCII 코드 표현방식

- ASCII 코드는 C언어가 기본적으로 사용하는 문자 인코딩 방식이다.
- 문자 하나를 7비트로 표현하므로 128개의 문자를 출력할 수 있다.
- 128개 문자는 52개의 영문 대소문자와 10개의 숫자, 32개의 특수 문자, 1개의 공백 문자, 그리고 출력이 불가능한 33개의 제어문자로 구성된다.

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

3) 정수형

① signed

- signed는 기본적으로 붙어있는 키워드로, 총 저장할 수 있는 양을 반으로 쪼개 음수/양수의 부호를 포함하게끔 저장하도록 하는 역할을 한다.
- 예를 들어 (signed) int의 경우 -2,147,483,648부터 2,147,483,647까지의 수를 저장할 수 있다.
- signed는 정수형뿐만이 아니라 문자형, 실수형에도 기본적으로 붙어 그 모습이 생략되어 있다.

② unsigned

- unsigned 키워드를 사용하게 되면, 부호 없이 총 저장할 수 있는 양을 그대로 저장함으로써 저장할 수 있는 크기를 늘릴 수 있다.
- 예를 들어 unsigned int의 경우 0부터 4,294,967,295까지를 저장할 수 있다.
- unsigned 또한 정수형뿐만이 아니라 문자형, 실수형에도 붙을 수 있다.

③ 오버플로우와 언더플로우

- 오버플로우(Overflow)란 해당 타입이 표현할 수 있는 최대 범위보다 더 큰 수를 저장할 때, 데이터가 해당 변수의 최상위 비트를 벗어나 인접 비트를 덮어씌워 잘못된 결과가 나타나게 되는 현상이다.
- 언더플로우(Underflow)란 반대로 해당 타입이 표현할 수 있는 최소 범위보다 더 작은 수를 저장할 때, 데이터가 해당 변수의 최하위 비트를 벗어나 인접 비트를 덮어씌워 잘못된 결과를 나타내는 현상이다.
- 이와 같은 현상들이 발생하면 전혀 예상치 못한 결과를 얻을 수 있으므로, 데이터를 저장할 때는 언제나 해당 데이터 타입의 최대/최소 크기를 고려해야 한다.

4) 실수형

① 부동소수점 표현방식

- 실수는 셀 수 없이 무한하고, 컴퓨터가 저장할 수 있는 용량은 한계가 있다. 이 사실은 컴퓨터가 표현하는 실수 데이터에는 오차가 있을 수밖에 없는 기술적 한계를 만들어 낸다.
- 이러한 까닭으로 실수형 데이터의 타입을 결정할 때는 표현 범위 이외에도 유효 자릿수의 고려도 필요하다.

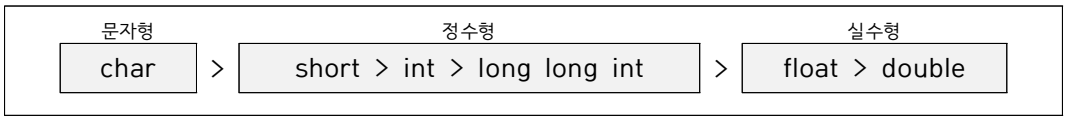
2. 형변환

- C언어에서 다른 타입끼리의 연산은 우선 피연산자들을 모두 같은 타입으로 만든 후에 수행된다.

1) 묵시적 형변환

① C 컴파일러가 자동으로 실행해주는 타입 변환으로, 자동 형변환으로도 불린다.

② 묵시적 형변환에서는 항상 데이터의 손실이 최소화되는 방향으로 진행된다. 즉 범위가 작은 쪽에서부터 큰 쪽으로, 아래와 같이 일방향적으로 변환된다.



2) 명시적 형변환

① 코더가 타입 캐스트 연산자(**(자료형)**)를 사용하여 직접 수행하는 타입 변환으로, 강제 형변환으로도 불린다.

3. 실습

1) LAB5_1 (오버플로우)

int형 변수에 2147483647, 2147483648을 저장한 다음 출력해보라. 어떤 값이 출력되는가?

2) LAB5_2 (실수 타입의 유효 자릿수)

초기값을 3.1415926535897932로 하는 float, double 변수를 각각 하나씩 선언하라. 이후 각 변수에 대하여 소수점 20자리까지 출력하여 결과를 확인하고 아래에 기입하라.

- **float형 변수의 유효 자릿수는 __자리**이다.
- **double형 변수의 유효 자릿수는 __자리**이다.

3) LAB5_3 (char형 연산, 형변환)

알파벳 대문자를 하나 입력받아 바로 다음 알파벳을 출력하라. 'Z'가 들어오면 'A'가 출력되도록 하라.

힌트는 아래와 같다.

- ASCII에서 'A'부터 'Z'까지는 연속적이다.
- 알파벳의 개수는 26개이다.

```
Enter a character: A
The character after 'A' is B
```

```
Enter a character: Z
The character after 'Z' is A
```

4) LAB5_4 (char형 연산)

문자를 하나 입력받아 대문자면 소문자로, 소문자면 대문자로 변환하여 출력하라. 알파벳이 아닌 것이 들어오면 "ERROR!"를 출력하고 종료하도록 하라.

힌트는 아래와 같다.

- ASCII에서 'A'부터 'Z'까지, 'a'부터 'z'까지도 연속적이다.
- 'A'는 아스키코드 값으로 65, 'a'는 97이다.

```
Enter a character: A
Converted to lowercase: a
```

```
Enter a character: p
Converted to uppercase: P
```

```
Enter a character: 8
ERROR!
```

5) LAB5_5 (형변환)

두 개의 피연산자(int)를 받아 첫 번째 피연산자를 두 번째 피연산자로 나눈 실수형 결괏값, 정수형 몫과 나머지를 차례로 출력하는 프로그램을 작성하라.

```
Enter the first operand: 50
Enter the second operand: 7

50 / 7 = 7.143
몫 : 7, 나머지 : 1
```

```
Enter the first operand: 1000
Enter the second operand: 237

1000 / 237 = 4.219
몫 : 4, 나머지 : 52
```

6) LAB5_6 (형변환)

LAB5_5를 참고하여 LAB4_4를 사칙연산이 전부 가능하도록 보완하라.

요구사항 및 가정은 아래와 같다.

- 연산자로 '/'가 입력된 경우 두 번째 피연산자는 반드시 0이 오지 않는다.
- 나누기 계산 시 소수점 5자리 이상의 결괏값이 나오는 식은 입력되지 않는다.

보완점을 고민해보라.

- '/'의 입력을 받는다면 추가해야 할 부분은 있는가?
- 변수는 그대로 충분한가?
- 이외 변경, 보완해야 하는 점은 있는가?

```
Enter an operator: /
Enter the first operand: 1
Enter the second operand: 4
1 / 4 = 0.25
Right answer
```

```
Enter an operator: /
Enter the first operand: 50
Enter the second operand: 5
50 / 5 = 0
Wrong!
10.000000 is the right answer
```