2장. 변수와 자료형

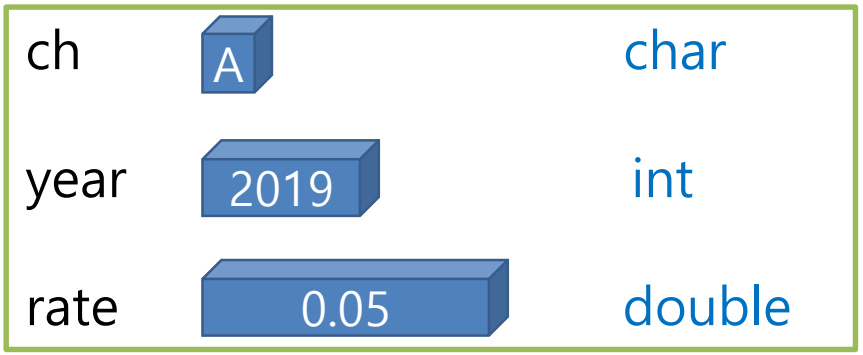
Visualstudio 2019

변수(Variable)

- **변수란?**
  - 프로그램 내부에서 사용하는 데이터를 저장해두는 메모리 공간
  - 한 순간에 한 개의 값만을 저장한다.(배열-여러 개 저장)
- **변수의 선언 및 사용**
  - 자료형 **변수이름**;
  - 자료형 **변수이름 = 초기값;**

변수 선언문

```
char ch;

int year = 2019;

double rate = 0.05;
```



변수의 선언과 사용

- **변수이름**
  - 영문자, 숫자, 밑줄 사용{ 예) int num_1 }
  - 변수명의 첫 글자는 밑줄이나 영문자여야 한다.
    숫자로 시작할 수 없고, 공백도 허용되지 않음{ 오류. int 8number }
  - 예약어는 사용할 수 없다 { for, class 등}

```
char name[20] = "leesan";  //선언과 동시에 초기화
int grade;
//int class;      //예약어 사용 불가
int schoolClass;

grade = 2;
schoolClass = 5;

//cout << name << endl;     //leesan
//cout << name[2] << endl;  //e

cout << name << " 학생은 " << grade << "학년 " << schoolClass << "반이다.\n";
```

자료형(data type)

- **자료형이란?**

  데이터를 저장하는 공간의 유형

자료형		용량 (bytes)	주요 용도	범위
정수형	**short**	2	정수 표현	-32768~32767
	int	4	큰 범위의 정수 표현	-2147483648~2147483647
	long	8	Int보다 큰 범위	2^{-63} ~ 2^{63}-1
실수형	**float**	4	실수 표현	10^{-38} ~ 10^{38}
	double	8	정밀한 실수 표현	10^{-380} ~ 10^{380}
문자형	**char**	1	문자 또는 작은 정수 표현	-128~127
논리형	**bool**	1	참 / 거짓, true / false	0, 1

자료형의 메모리 크기

- **자료형의 크기**

```
//자료형의 크기
cout << "int 형 : " << sizeof(int) << "byte" << endl;
cout << "short 형 : " << sizeof(short) << "byte" << endl;
cout << "long 형 : " << sizeof(long) << "byte" << endl;
cout << "float 형 : " << sizeof(float) << "byte" << endl;
cout << "double 형 : " << sizeof(double) << "byte" << endl;
cout << "char 형 : " << sizeof(char) << "byte" << endl;
cout << "bool 형 : " << sizeof(bool) << "byte" << endl;
```

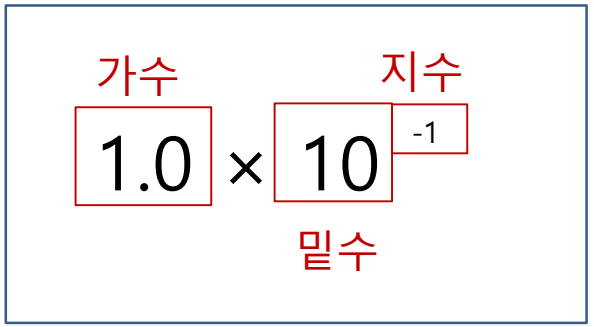
자료형의 메모리 크기

- **자료형의 크기 및 범위**

```
int num1 = 10000000050;  //100억 불가
cout << num1 << endl;

char ch = 65;   //'A'
cout << ch << endl;

char ch2 = 200;  // (0~127)범위 초과
cout << ch2 << endl;

int num = 2147483647;
cout << "변수 num에 저장된 값은 " << num << "입니다." << endl;

num = 2147483648;  //오버 플로우
cout << "변수 num에 저장된 값은 " << num << "입니다." << endl;
```

실수 자료형

- **실수 자료형(float, double)**
  - 실수값 3.14를 표현하면 3이라는 정수부분과 .14라는 소수 부분을 따로 표현할수 있고, 0과 1사이에는 무한개의 실수가 있다.
  - 부동 소수점 방식 : 실수를 지수부와 가수부로나눔.
  - 정밀도의 차이 : float형(소수이하 6자리), double형(소수이하 15자리)



▶0.1을 표현하는 방식

실수 자료형

- **실수 자료형(float, double)**

```
float fNum = 1.123456;
double dNum = 1.1234567890123456;

cout << "float 형 : " << fNum << endl;
cout << "double 형 : " << dNum << endl;
```

문자 자료형

▪ 문자 자료형(char, 배열)

- **아스키 코드(ASCII code)** – 각 문자에 따른 특정한 숫자 값(코드 값)을 부여

  영문자, 숫자 만 표현 가능(문자 1개를 표현할때 홑따옴표(' ')로 감싸준다.)

- **유니 코드(uni code)** – 한글, 중국어등 아스키 코드로 표현할 수 없는 문자를 2바이트 이상의 크기로 표현할수 있는 표준 코드 (문자 1개를 쌍따옴표(" ")로 감싼다.)

```
//문자 자료형
char ch = 'a';
char ch2[3] = "한";      //한글은 배열로 선언(끝에 '\0' 포함)
char ch3[] = "\uD55C";  //유니코드로 한글 입력

cout << "ch = " << ch << endl;
cout << "ch의 아스키 값 = " << (int)ch << endl;  //형 변환

cout << "ch2 = " << ch2 << endl;
cout << "ch3 = " << ch3 << endl;

char cart[4] = "egg";
char cart2[] = "커피";

cout << cart << endl;
cout << cart2 << endl;
```

http://www.unicode.org/charts/PDF/UAC00.pdf

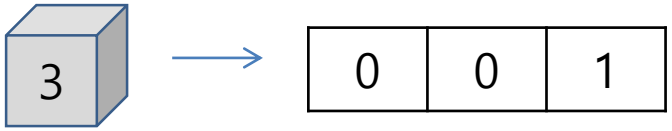
문자 자료형

- **문자열 자료형(string 클래스)**
  - 문자열은 배열로 표현 한다.
  - string 클래스를 사용

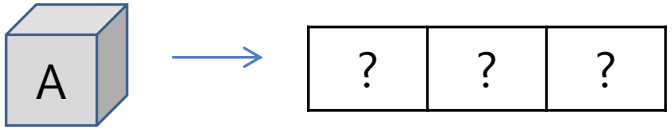
```
string cart = "생수";   //string 자료형
string cartlist[] = { "커피", "생수", "계란" };
cout << cart << endl;

cout << "*** cartList의 계란 조회 ***" << endl;
cout << cartlist[2] << endl;

cout << "*** cartList 목록 출력 ***" << endl;
for (int i = 0; i < 3; i++) {
    cout << cartlist[i] << " ";
}
```

아스키 코드

10진수를 2진수로 변환



문자를 2진수로 변환



저장하는 문자에 해당하는 숫자를 지정하고 메모리에 저장할때는 그 숫자를 비트 단위로 바꾸어 저장

- 아스키 코드(ASCII Code)

아스키 코드는 미국 ANSI에서 표준화한 정보교환용 7비트 부호체계이다.

000(0x00)부터 127(0x7F)까지 총 128개의 부호가 사용된다.

영문 키보드로 입력할 수 있는 모든 기호들이 할당되어 있는 부호 체계이다.

아스키 코드

- 아스키 코드(ASCII Code)

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

문자 자료형

- **bool 자료형**
  - 참과 거짓을 표현하는 논리 자료형이다.
  - 입력은 true / false로 출력은 0 / 1로 한다.

```
//저장(입력) : true / false
bool t = true;
bool f = false;

int iTrue = true;
int iFalse = false;

//출력 : 0 / 1
cout << "t : " << t << endl;
cout << "f : " << f << endl;
cout << "iTrue : " << iTrue << endl;
cout << "iFalse : " << iFalse << endl;

cout << "t의 크기 : " << sizeof(t) << endl;
cout << "f의 크기 : " << sizeof(t) << endl;
cout << "iTrue 의 크기 : " << sizeof(iTrue) << endl;
cout << "iFalse 의 크기 : " << sizeof(iFalse) << endl;
```

문자 자료형

- **bool 자료형**

  - 0이 아닌 수는 모두 1(true)이다.

```
bool b = true;
cout << "b = " << b << endl;    //1

b = false;
cout << "b = " << b << endl;    //0

b = 7;
cout << "b = " << b << endl;    //1

b = 0;
cout << "b = " << b << endl;    //0
```

```
bool b;

b = 10 > 11;
cout << "b = " << b << endl;

b = 10 <= 11;
cout << "b = " << b << endl;

b = (10 == 11);
cout << "b = " << b << endl;

b = (10 != 11);
cout << "b = " << b << endl;
```

컴퓨터에서 데이터 표현하기

- **비트(binary digit)**

  컴퓨터가 표현하는 데이터의 최소 단위로 2진수 하나의 값을 저장할 수 있는 메모리의 크기

  컴퓨터는 0과 1로만 데이터를 저장함(0-> 신호꺼짐, 1-> 신호켜짐)

- 비트로 표현할 수 있는 수의 범위

비트수	표현할 수 있는 범위(십진수)	
1bit	0, 1(0~1)	2^1
2bit	00, 01, 10, 11(0~3)	2^2
3bit	000, 001, 010, 011, 100, 101, 110, 111(0~7)	2^3

10진수를 2진수로 바꾸기

- **진수 표현**

10진수	2진수	16진수	10진수	2진수	16진수
1	00000001	1	9	00001001	9
2	00000010	2	10	00001010	A
3	00000011	3	11	00001011	B
4	00000100	4	12	00001100	C
5	00000101	5	13	00001101	D
6	00000110	6	14	00001110	E
7	00000111	7	15	00001111	F
8	00010000	8	16	000010000	10

자리 올림 발생

부호 있는 수를 표현하는 방법

- **음의 정수는 어떻게 표현할까?**
  - 정수의 가장 왼쪽에 존재하는 비트는 부호비트입니다.
    (양의 정수는 0, 음의 정수는 1을 붙인다.)
  - 음수를 만드는 방법은 2의 보수를 취한다.(1의 보수는 0과 1을 반대로 바꿈)

부호 있는 수를 표현하는 방법

- **10진수, 이진수, 16진수로 데이터 표현하기**

```
int num = 10;
int bNum = 0b1010;
int hNum = 0xA;

cout << num << endl;
cout << bNum << endl;
cout << hNum << endl;
```

형 변환(Type Conversion)

● **자료의 형 변환**

- 표현 범위가 작은 자료형 -> 큰 자료형

  예) float dNum = int iNum;

- 표현 범위가 큰 자료형-> 작은 자료형(데이터가 손실될 수 있음)

  예) int iNum = float dNum;

```
int a = 20, b = 3;
float c, d;

c = a / b;
d = a / (float)b;  //형 변환

cout << c << endl;
cout << d << endl;
```

형 변환 실습 문제

변수 두 개를 선언해서 10과 2.5를 대입하고, 두 변수의 사칙연산의 결과를 정수로 출력해 보세요.

```
int num1 = 10;
double num2 = 2.5;

cout << (int)(num1 + num2) << endl;
cout << (int)(num1 - num2) << endl;
cout << (int)(num1 * num2) << endl;
cout << (int)(num1 / num2) << endl;
```

상수(constant)

- **상수(constant)**
  - 한번 설정해 두면 그 프로그램이 종료 될 때까지 결코 변경될 수 없는 값
  - 상수를 숫자로 직접 나타내는 것보다 이름을 붙여 사용하는 것이 좋다.
  - 상수 이름은 대문자로 관례적으로 사용.

(1) **#define 상수이름 상수값** – 매크로 상수로 정의(전처리, 컴파일되지 않음)

(2) **const** 자료형 **상수이름** = **상수값**;

```
#define PI 3.141592

const double PI = 3.141592;

//PI = 3.14 (변경할 수 없다.)
```

상수(constant)

- 상수(constant)

```
const int MIN_NUM = 1;
const int MAX_NUM = 100;

//MAX_NUM = 200; 상수는 변경 불가

cout << MIN_NUM << endl;
cout << MAX_NUM << endl;

//원의 넓이 계산하기
const double PI = 3.14;
int radius = 5;
double area;

area = PI * radius * radius;
cout << area << endl;
```

매크로 상수

메이저리그는 점점 더 빠른 구속을 추구하고 있다. 올 시즌 메이저리그 포심 패스트볼 평균 구속은 시속 93.2마일(150.0km)에 달한다. 이제는 100마일(160.9km)이 넘는 공도 어렵지 않게 볼 수 있게 됐다.

투수에게 있어 구속이 가장 중요한 요소는 아니다. 구종, 제구, 구위 등 수 많은 요소들이 어우러져야 비로소 뛰어난 투구를 할 수 있다. 하지만 같은 조건이라면 당연히 구속이 빠를수록 유리하다. 구속이 빠를수록 타자들이 공에 대처할 수 있는 물리적인 시간이 줄어들기 때문이다.

매크로 상수

◈ 속도 KM를 마일로 바꾸는 프로그램

```
#include <iostream>
using namespace std; //소속(분류)
#define RATE_KPH_MPH 1.609344  //매크로 상수

int main() {
    //공의 속도(구속) 변환 프로그램 - km -> 마일
    int kph;         //입력 변수
    double mph;     //변환 결과 - 마일의 속도

    cout << "당신의 구속을 입력하세요[km/h]: ";
    cin >> kph;
    mph = kph / RATE_KPH_MPH;

    cout << fixed;
    cout.precision(2);  //소수 2자리 설정

    cout << "공의 속도는 " << mph << "[MPH]입니다." << endl;

    return 0;
}
```

키보드로 데이터 입력 받기

- **입력하기**

<iostream>을 포함하여 cin 객체를 사용하여 입력한다.

연산자 기호는 '>>' 를 사용함

```
int age;
char name[20];

cout << "당신의 나이는 몇살입니까? ";
cin >> age;
cout << "당신의 나이는 " << age << "세 이군요!" << endl;

cout << "당신의 이름은 무엇입니까? ";
cin >> name;
cout << "당신의 이름은 " << name << "이군요!" << endl;
```

덧셈과 뺄셈 계산기

- 고객의 구매 포인트 계산 프로그램

```
int money, point;
char name[30];

cout << "이름과 전화번호를 입력하세요" << endl;
//cin >> name;   //공백을 처리하지 못함
cin.getline(name, sizeof(name));

cout << "구매 금액을 입력하세요 : ";
cin >> money;
point = money * 0.05;

cout << "구매 포인트는 " << point << "점입니다." << endl;
```