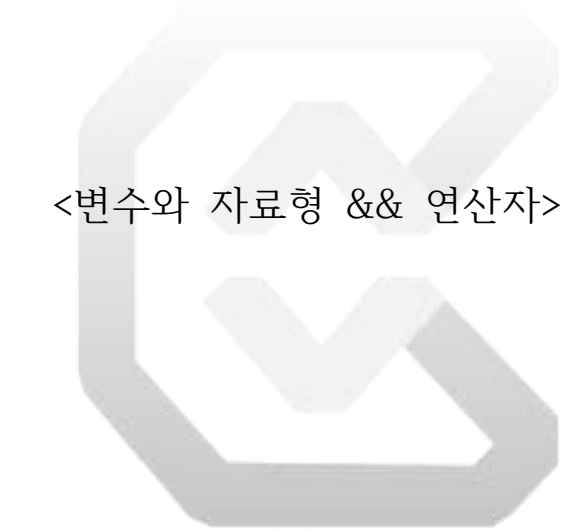
CAPS

C언어 스터디
1.5주차

<변수와 자료형 && 연산자>



CAPS

CAPS

변수

1. 정수형 - 오차가 없습니다. 하지만 최솟값과 최댓값이 있습니다.
    가. int -> %d
    나. long long -> %ll
2. 문자형
    가. char -> %c
3. 실수형 - 오차가 있습니다. 다만, 최솟값과 최댓값이 없습니다.
    가. float -> %f
    나. double -> %lf
4. 전역 변수, 지역변수
5. 포인터

```
#include<stdio.h>
int main(void) {
	int a;
	a = 1;
	printf("a는 %d\n", a); // 최대값은 약 2'000'000'000입니다.

	long long b = 1'000'000'000'000'000;
	printf("b는 %lld\n", b); // 최대값은 약 2'000'000'000'000'000'000

	char c = 'A';
	printf("c는 %c\n", c);
	printf("c는 %d\n", c); // 문자열은 사실 int 입니다 (정확히는 short)

	float d = 1.01;
	printf("d는 %f\n", d);

	double e = 1.11111;
	printf("e는 %lf\n", e); // double이 float 보다 더 정확하고 큰 값을 저장할
수 있습니다.
	printf("e는 %.9lf\n", e); // .n으로 소수점 n번째 자리수까지 출력 가능합니다.

	double f = 1.11111;
	printf("f는 %d\n", f); // '어떤' 자료형을 '어떻게' 읽을까? 가 중요합니다.
	printf("f는 %d\n", (int)f);

	bool g = true;
	printf("%d", g);
}
```

연산자

연산자는 함수다.

```
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>
int main(void) {
	int a;
	int b;
	a = 1;
	b = 1 + 2; // f(1,2)->3
	printf("%d\n", b);
	b = 1 + a;
	printf("%d\n", b);
	a = a + 7;
	printf("%d\n", a);
} // = 는 오른쪽에서 왼쪽입니다. 기억하세요!
```

괄호가 우선 연산!!

1. 사칙연산 +, -, *, /, %

```
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>
int main(void) {
	int a;
	int b;
	a = 5;
	b = 2;
	printf("%d\n", a + b);//7
	printf("%d\n", a - b);//3
	printf("%d\n", a * b);//10
	printf("%d\n", a / b);//2
	printf("%d\n", a % b);//1
	printf("%d\n", (a + b)*b);//14

	double c, d;
	c = 5;
	d = 2;
	printf("%lf\n", c / d);//2.5000000
}
```

2. 비교연산 >, <, ==, !=, >=, <=

```
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>
int main(void) {
	int a = 1;
	int b = 2;
	int c = 2;
	printf("%d\n", a < b);//1 크다
	printf("%d\n", a > b);//0 작다
	printf("%d\n", b <= c);//1 같거나 크다
	printf("%d\n", a <= b);//1
	printf("%d\n", a == b);//0 같다
	printf("%d\n", a != b);//1 다르다
	printf("%d\n", b == c);//1
} // 두 수를 받고, 1과 0을 뱉는 함수라 생각하세요!
```

3. +=, -=, *=, /=, %=, ++, --

```
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>
int main(void) {
	int a = 1;
	printf("%d\n", a);
	a += 2; // a에 2를 직접 더한다.
	printf("%d\n", a);//3
	a -= 3; // a에 3를 직접 뺀다.
	printf("%d\n", a);//0
	a++; // a에 1을 더한다
	printf("%d\n", a);//1
	a--; // a에 1을 뺀다
	a--; // a에 1을 뺀다
	printf("%d\n", a);//-1
} // 처음에는 잘 손에 안 익을 겁니다. 사칙연산에 익숙해지고 사용해도 무방합니다!
```

```
a=a+2;
a+=2;

a=a+1;
a+=1;
a++;
```

4. bool연산 &&(and), ||(or), !(not)

```
#include<stdio.h>
int main(void) {
	int a = 1;
	int b = 0;
	printf("%d\n", a && a);//1
	printf("%d\n", a && 0);//0
	printf("%d\n", a || 0);//1
	printf("%d\n", b || 0);//0
	printf("%d\n", !a);//0
	printf("%d\n", !2);//0
//	신기하죠?
//	0이면 거짓, 0이 아니면 참입니다. 즉 3이라는 것도 참이 될 수 있어요.
//	이때 bool 연산을 통해 "나온 값"은 1혹은 0으로 고정됩니다.
	printf("%d\n", (a && a) && 1);//1
	printf("%d\n", ((a && a) && 1) && 1);//1
	printf("%d\n", 1 || ((a && a) && 1));//1
	printf("%d\n", !(1 || ((a && a) && 1)));//0
}
```

5. 비트마스킹 &(and), |(or), ^(xor), >>, <<(shift)

```
#include<stdio.h>
void printBinary(int n); // 무시해 주세요
int main(void) {
    int a = 3;
    int b = 10;
    printBinary(a);    // ..0011
    printBinary(b);    // ..1010
    printBinary(a&b); // ..0010
    printBinary(a|b); // ..1011
    printBinary(a^b); // ..1001
    printBinary(a<<1);// ..0110
    printBinary(a>>1);// ..0001
}
void printBinary(int n) { // 무시해 주세요
    for (int i = 31; i >= 0; i--) {
        int k = n >> i;
        if (k & 1)printf("1");
        else printf("0");
    }
    printf("\n");
}
```

6. 형변환

```c
#include<stdio.h>
int main(void) {
	int a=10;
	double b=30;
	b=a/3;
	printf("%lf\n", b);//3.0000
	//int 자료형에 3을 나눈 int형을 double에 욱여 넣기 (자동 형변환)

	b = (double)(a / 3);
	printf("%lf\n", b);//3.0000
	//int 자료형에 3을 나눈 int형을 double이라고 정의하기 (강제 형변환)

	b = (double)a / 3;
	printf("%lf\n", b);//3.3333
	//int 자료형을 강제로 double이라고 하고, double에 3을 나누니 3.3333

	b = ((double)a) / 3;
	printf("%lf\n", b);//3.3333
	//int 자료형을 강제로 double이라고 하고, double에 3을 나누니 3.3333
}
/*
하지만 주의해야 합니다. 컴파일러 마다 결과가 달라질 수 있거든요.
int 10에 3을 나누면 3입니다
double 10에 3을 나누면 3.3333입니다.

그래서 네 번째의 경우처럼 명확하게 처리하는 게 좋습니다.
자동 형변환보다는 강제 형변환을 잘 써 봅시다.
그리고 애초에 형변환을 안 쓰는 상황으로 잘 끌어 봅시다.
*/
```

과제)

1. 34721 역사를 걸으면 동국이 보이고
   https://www.acmicpc.net/problem/34721
   https://github.com/sungjoonyoung/BOJ/blob/main/30000/34721.cpp
   동국대학교 알고리즘 대회에 나왔던 문제입니다. 알고리즘 대회? 해볼만 하네요.
2. 1000 A+B
   https://www.acmicpc.net/problem/1000
   https://github.com/sungjoonyoung/BOJ/blob/main/00000/1000.cpp
3. 1001 A-B
   https://www.acmicpc.net/problem/1001
   https://github.com/sungjoonyoung/BOJ/blob/main/00000/1001.cpp

4. 10998 A×B
   https://www.acmicpc.net/problem/10998
   https://github.com/sungjoonyoung/BOJ/blob/main/10000/10998.cpp
5. 1008 A/B
   https://www.acmicpc.net/problem/1008
   https://github.com/sungjoonyoung/BOJ/blob/main/00000/1008.cpp
6. 10869 사칙연산
   https://www.acmicpc.net/problem/10869
   https://github.com/sungjoonyoung/BOJ/blob/main/10000/10869.cpp
7. 11382 꼬마 정민
   https://www.acmicpc.net/problem/11382
   https://github.com/sungjoonyoung/BOJ/blob/main/10000/11382.cpp
8. 11654 아스키 코드
   https://www.acmicpc.net/problem/11654
   https://github.com/sungjoonyoung/BOJ/blob/main/10000/11654.cpp
9. 8393 합 (풀면? 똑똑이)
   https://www.acmicpc.net/problem/8393
   https://github.com/sungjoonyoung/BOJ/blob/main/00000/8393.cpp
10. 2884 알람 시계 (풀면 밥 사줍니다.)
    https://www.acmicpc.net/problem/2884
    https://github.com/sungjoonyoung/BOJ/blob/main/00000/2884.cpp
11. 2753 윤년 (풀면 밥 사줍니다.)
    https://www.acmicpc.net/problem/2753
    https://github.com/sungjoonyoung/BOJ/blob/main/00000/2753.cpp