

2주차 : 변수, 사칙 연산, 연산자, 조건문 , if문

강사: 이주은

Ch 0 . 목록

Ch .1

- 변수
- 산술 연산자
- 대입 연산자
- 증감 연산자
- 비교 연산자
- 논리 연산자

Ch . 2

- if 문
- switch 문

Ch1 . 변수와 연산자

변수란?

변수6 變數

1. 어떤 상황의 가변적 요인.
2. [수학] 어떤 관계나 범위 안에서 여러 가지 값으로 변할 수 있는 수.

변수란?

컴퓨터인터넷IT용어대사전

변수

[variable 📖]

- (1) 일반적으로는 미리 정해진 범위(range) 내에서 값(value)이 변할 수 있는 수를 대표하는 문자이다.
 - (2) 프로그래밍 언어인 COBOL에서는 프로그램 실행중에 값이 바뀌는 경우가 있는 데이터 항목이다. 또 FORTRAN에서는 배열도 배열 요소도 아닌 데이터 항목이며, 영자명(symbolic name)으로 식별된다. 인용하거나 정의하는 것이 가능하다.
 - (3) 형용사로서, 목적이나 용도(application) 등에 따라서 변경이 가능한 것, 또는 변화하는 것 등을 표시하는 의미로 사용된다. 예를 들면 가변 길이 레코드(variable length record) 등.
 - (4) 어떤 주어진 적용에 있어서 실제의 값이 할당되기까지, 값이 정해지지 않은 또는 기지(既知)의 범위에 값이 정해지지 않은 것.
 - (5) (프로그래밍에 있어서) 문자 또는 문자의 집합이며, 값을 참조하여 컴퓨터 프로그램의 실행중에는 어드레스에 대응하고 있는 것.
 - (6) 다른 값을 취할 수 있으나 그런 시점에서는 한 가지 값만을 취하는 언어 대상물.
- [주] 한 가지의 변수를 취할 수 있는 값은, 보통 어떤 정해진 데이터형으로 한정된다.

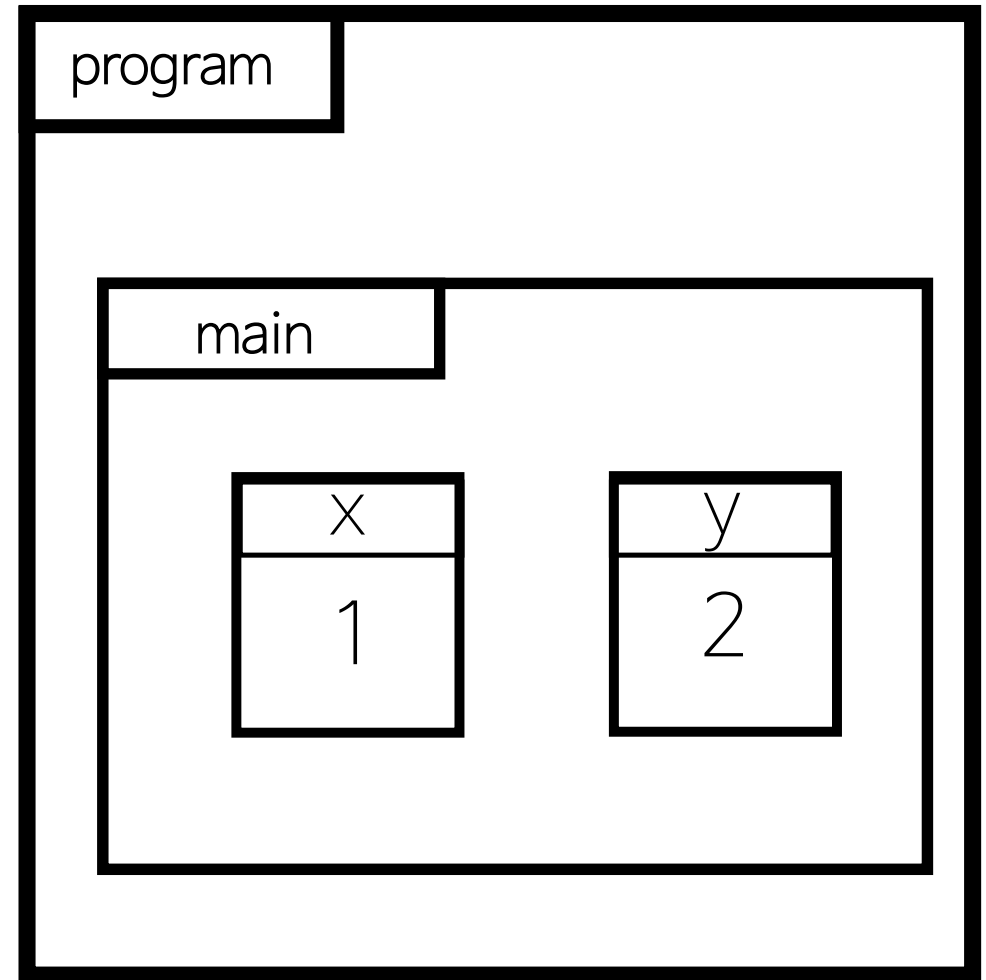
변수 (variable)

변수의 선언과 정의

```
1  #include<stdio.h>
2
3  int main(void) {
4      int x; // 변수의 선언
5      x = 1; // 변수의 정의
6
7      int y = 2; //변수의 선언과 정의를 동시에
8
9      return 0;
10 }
11
```

main 함수 내에 int형 변수 x를 지역변수로 선언을 해주고,
밑에 줄에서 x의 값을 정의해주었어요. (지역변수는 뒤에 나와요)

y처럼 한번에 선언과 정의를 동시에 해줄 수도 있어요



지역변수와 전역변수

전역변수(Global variable)

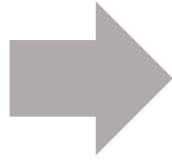
- 함수 밖에서 선언되는 변수
- 프로그램 시작부터 종료까지 존재
 - 어디서나 접근 가능
- 초기화 하지 않을 경우
'0' 으로 초기화

지역변수(Local variable)

- 중괄호 내에 선언되는 변수
- 선언된 지역에서 벗어나면 소멸
- 선언된 지역 내에서만 접근 가능
 - 초기화 하지 않을 경우
'쓰레기값' (?) 으로 초기화

Cf) 쓰레기값

```
1  #include<stdio.h>
2
3  int main(void) {
4      int a;
5      printf("%d", a);
6  }
```



Debug Error!

Program: C:\Users\W
Module: C:\Users\W\...
File:

Run-Time Check Failure #3 - The variable 'a' is being used without being initialized.

(Press Retry to debug the application)

지역변수 a를 선언만 해주고 정의해주지 않았더니 에러가 떠요.

자바처럼 초기화 하지 않았을 때 0으로 지정해주는 언어도 있지만, C언어에서는 지역변수의 값이 초기화되지 않았을 때 그 변수에 값을 대입하지 않아요.

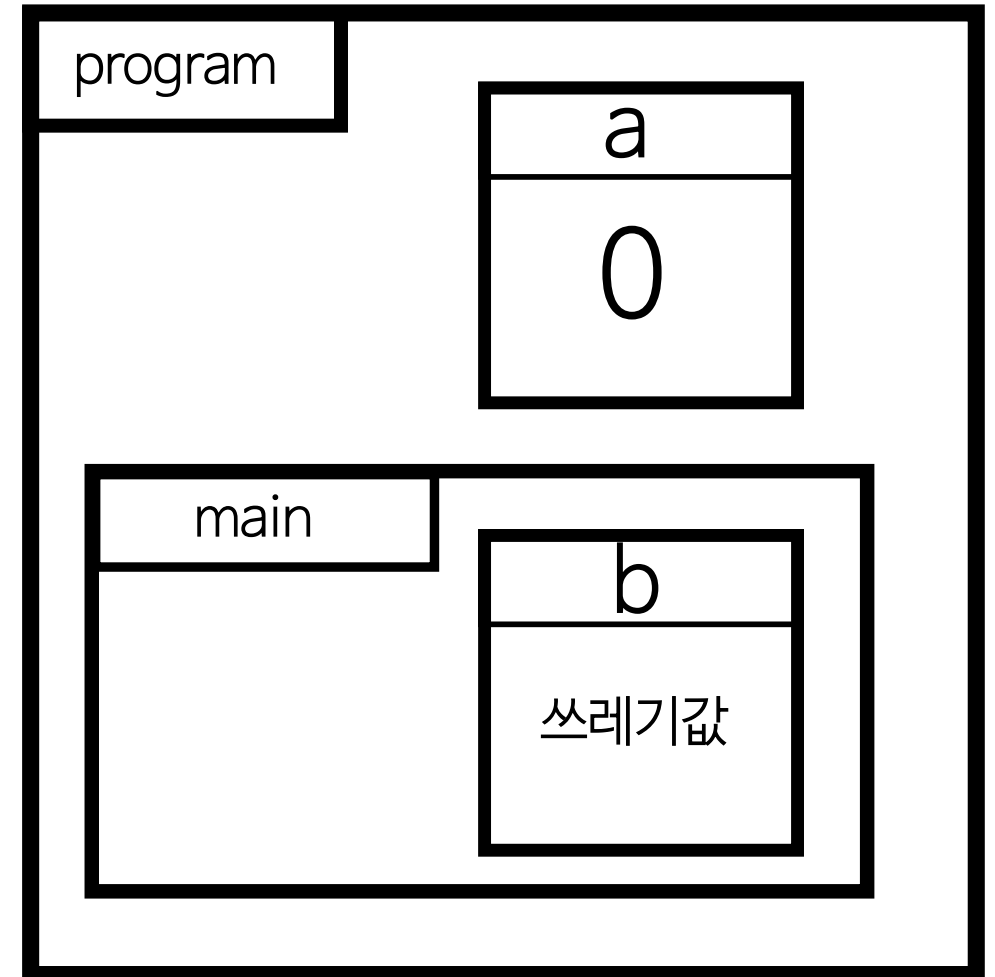
말그대로 아무 의미 없는 '쓰레기값'이 들어가게 돼요.

지역변수와 전역변수

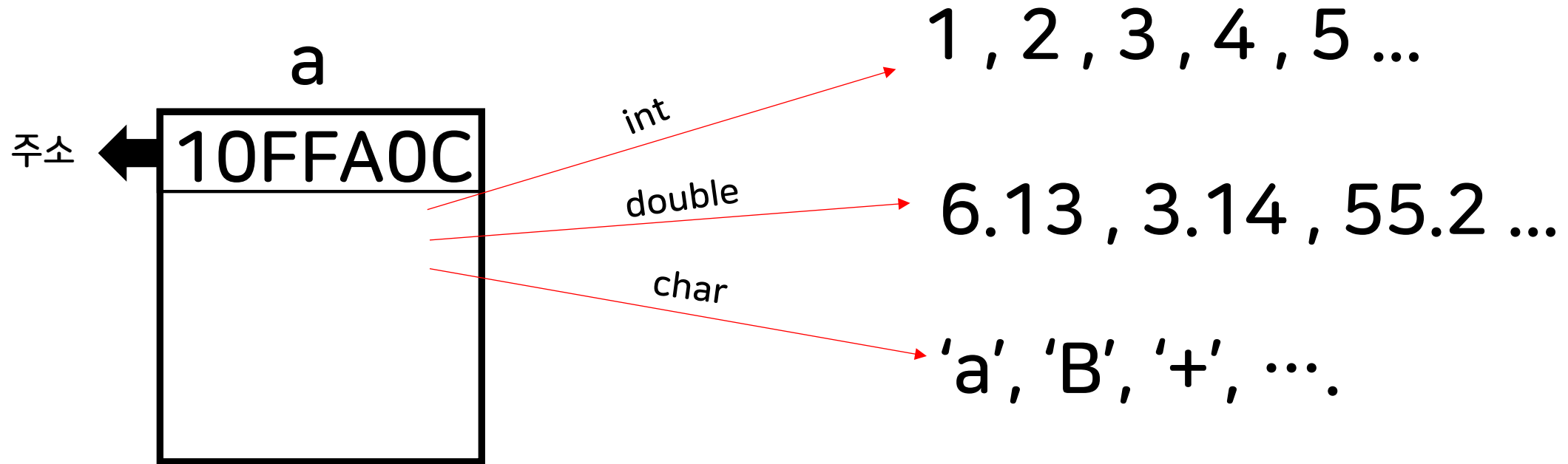
```
1  #include<stdio.h>
2
3  int a; //전역변수
4
5  int main(void) {
6      int b; //지역변수
7
8      return 0;
9  }
10
```

전역 변수는 정의하지 않았을 때, 메모리에 0이 들어가네요.

지역 변수는 쓰레기 값이 들어갔어요.



변수의 형식



메모리 자체에는 어떤 값도 들어갈 수 있어요 !
=> 변수의 형식이 중요한 이유

변수의 형식

| 유형 | 자료형 | 크기(byte) | 포맷 형태 | 범위 |
|---------|--------------|----------|--------|--|
| 정수 | int | 4 | %d | -2,147,483,648 ~ 2,147,483,647 |
| | unsigned int | 4 | %u | 0 ~ 4,294,967,295 |
| | long long | 8 | %lld | -9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 |
| | short | 2 | %d | -32,768 ~ 32,767 |
| 실수 | float | 4 | %f | 3.4E+/-38(7개의 자릿수) |
| | double | 8 | %lf | 1.7E+/-38(7개의 자릿수) |
| 문자(문자열) | char | 1 | %c(%s) | -128~127(15개의 자릿수) |
| 논리(T/F) | bool | 1 | - | - |

다양한 형식의 선언

```
1  #include<stdio.h>
2
3  int main() {
4      int x = 6;
5      char c = 'b';
6      double d = 3.14;
7      return 0;
8  }
```

산술 연산자

| 연산자 | 의미 |
|-----|-----|
| = | 대입 |
| + | 덧셈 |
| - | 뺄셈 |
| * | 곱셈 |
| / | 몫 |
| % | 나머지 |

뒤에 나오는 비교연산자(==)와 헷갈리지 말자!

연산자를 사용할 때는 같은 자료형끼리 사용해요.

같은 연산자라도 자료형에 따라 결과가 달라질 수 있어요.

산술 연산자 예시 1 >

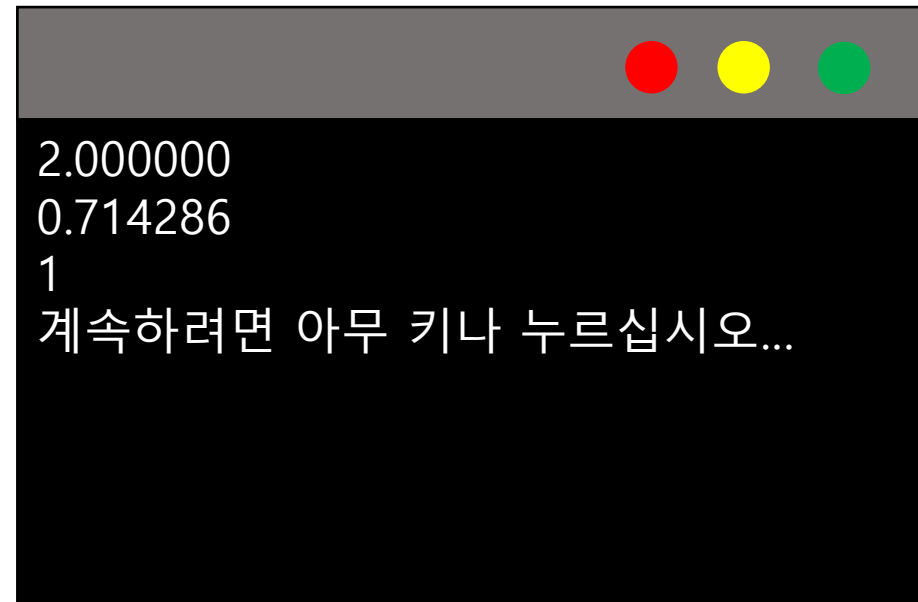
```
1  #include<stdio.h>
2
3  int main(void) {
4      int a = 2, b = 3;
5      printf("%d\n", a + b);
6
7      return 0;
8  }
```

5
계속하려면 아무 키나 누르십시오...

산술 연산자 예시 2>

Cf) (typename) : 형 변환 연산자

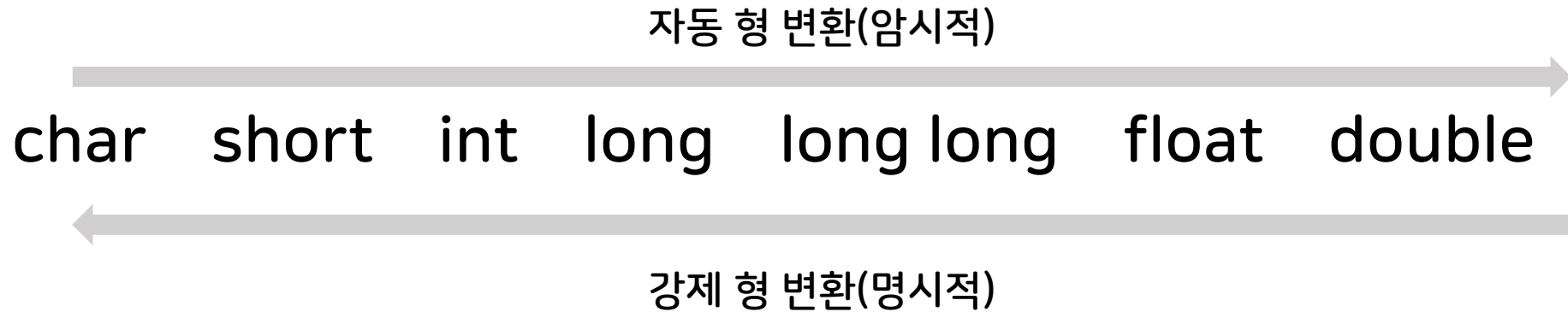
```
1  #include<stdio.h>
2
3  int main(void) {
4      int a = 5, b = 2;
5      double c = 5, d = 7;
6
7      printf("%lf\n", (double)(a / b));
8      printf("%lf\n", c / d);
9      printf("%d", a%b);
10
11     return 0;
12 }
13
```



A terminal window with a dark background and a grey title bar containing three colored circles (red, yellow, green). The output of the program is displayed as follows:

```
2.000000
0.714286
1
계속하려면 아무 키나 누르십시오...
```

cf) 형 변환(타입 캐스팅)

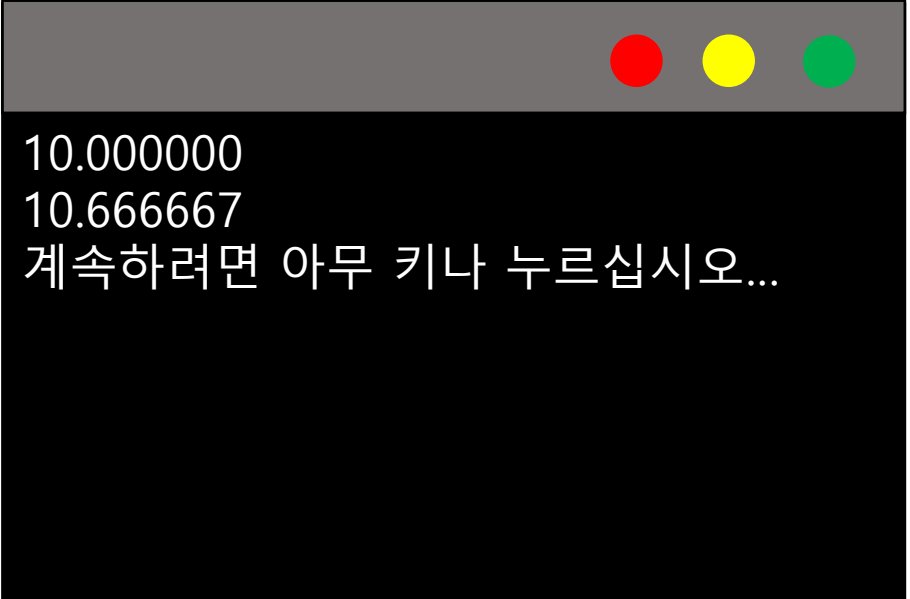


자동 형 변환 : 자료형의 크기가 큰 쪽으로 자동 변환되는 것

강제 형 변환 : 프로그래머가 강제로 자료형을 변환하는 것


cf) 형 변환

```
1  #include<stdio.h>
2
3  int main(void) {
4      int a = 32;
5      int b = 3;
6      float c;
7
8      c = a/b; //경고가 발생함
9      printf("%f\n", c);
10
11     c = (float)a / b;
12     printf("%f\n", c);
13 }
```



10.000000
10.666667
계속하려면 아무 키나 누르십시오...

cf) 형 변환(타입 캐스팅)

| 코드 | 설명 |
|---|---|
|  C4244 | '=': 'int'에서 'float'(으)로 변환하면서 데이터가 손실될 수 있습니다. |

앞의 코드 8번째 줄에서 경고가 발생하게 됩니다.

Int/int 의 결과는 int인데 float에 대입을 했기 때문이에요.

11번째 줄에서 (float)a 타입 캐스팅을 해주면 해결되겠죠?

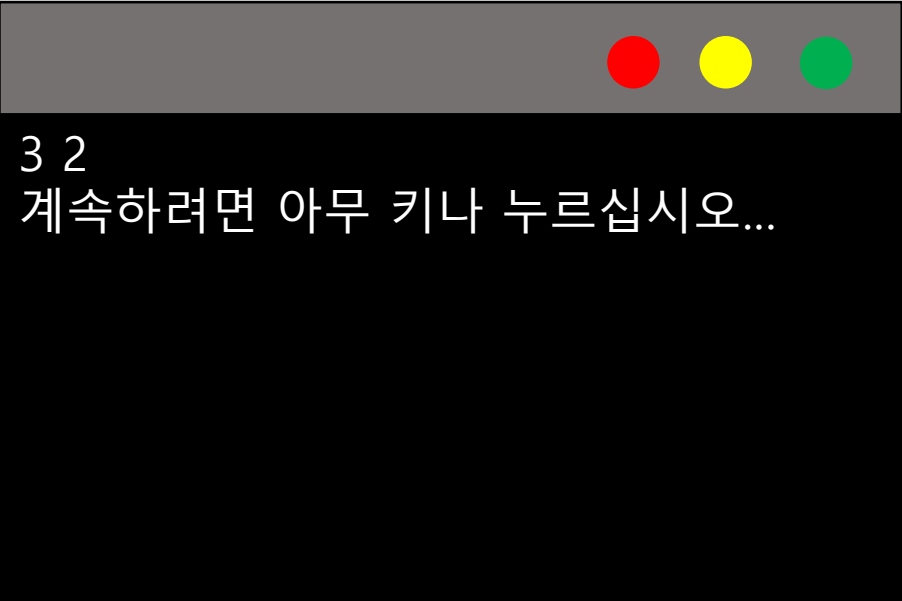
대입 연산자

| 연산자 | 의미 |
|-----------------|--------------------|
| <code>+=</code> | <code>a=a+b</code> |
| <code>-=</code> | <code>a=a-b</code> |
| <code>*=</code> | <code>a=a*b</code> |
| <code>/=</code> | <code>a=a/b</code> |
| <code>%=</code> | <code>a=a%b</code> |

`a=a+b;`
이렇게 쓰는 것 보다는
`a+=b;`
이렇게 쓰는 게
코딩하기 편하겠죠?

대입 연산자

```
1  #include<stdio.h>
2
3  int main(void) {
4      int a = 5;
5      int b = 8;
6      a -= 2;
7      b %= 3;
8      printf("%d %d\n", a, b);
9  }
```

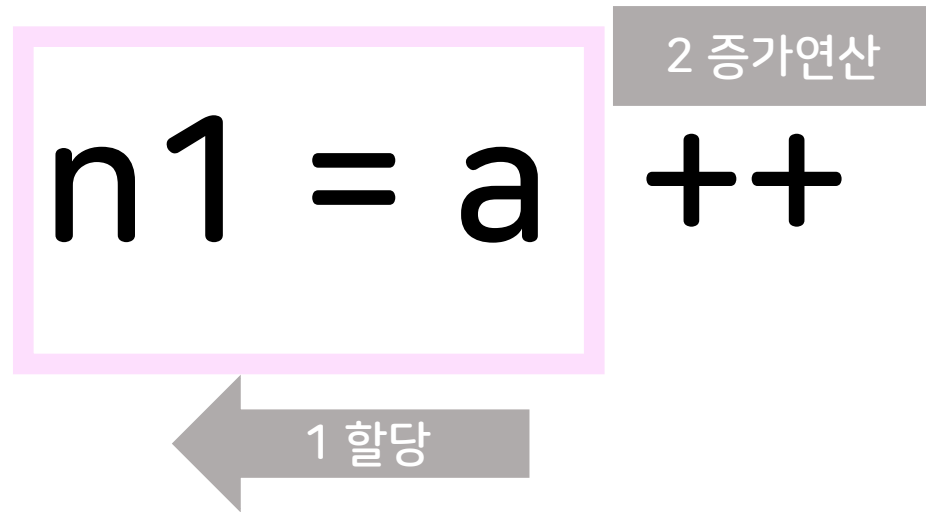


3 2
계속하려면 아무 키나 누르십시오...

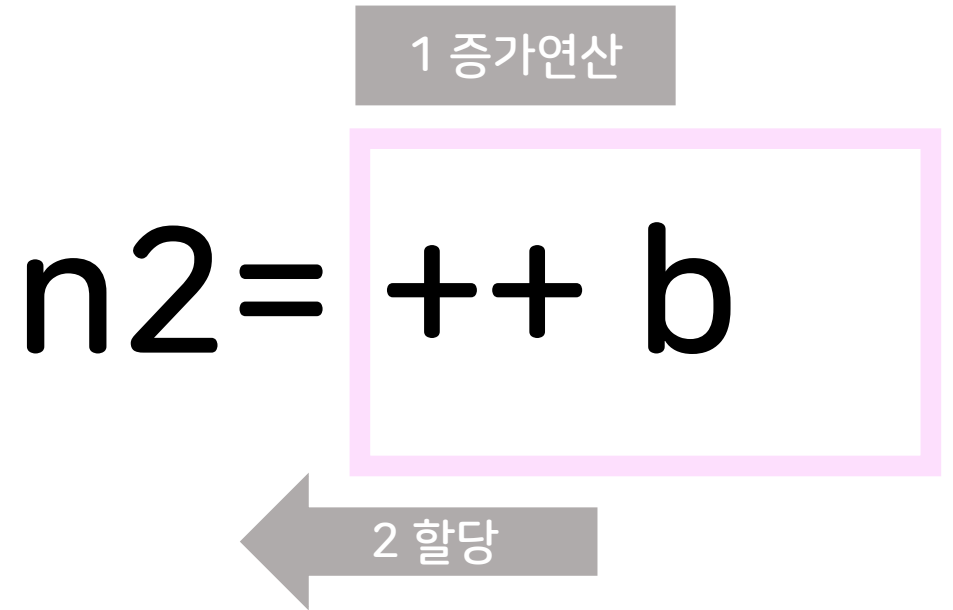
증감 연산자

| 연산자 | 의미 |
|-----|---------------------|
| a++ | 연산 후 증가 (후위 연산자) |
| a-- | 연산 후 감소 |
| ++a | 증가 후 연산 (전위 연산자) |
| --a | 감소 후 연산 |

증감 연산자



후위 연산자(postfix)



전위 연산자(prefix)

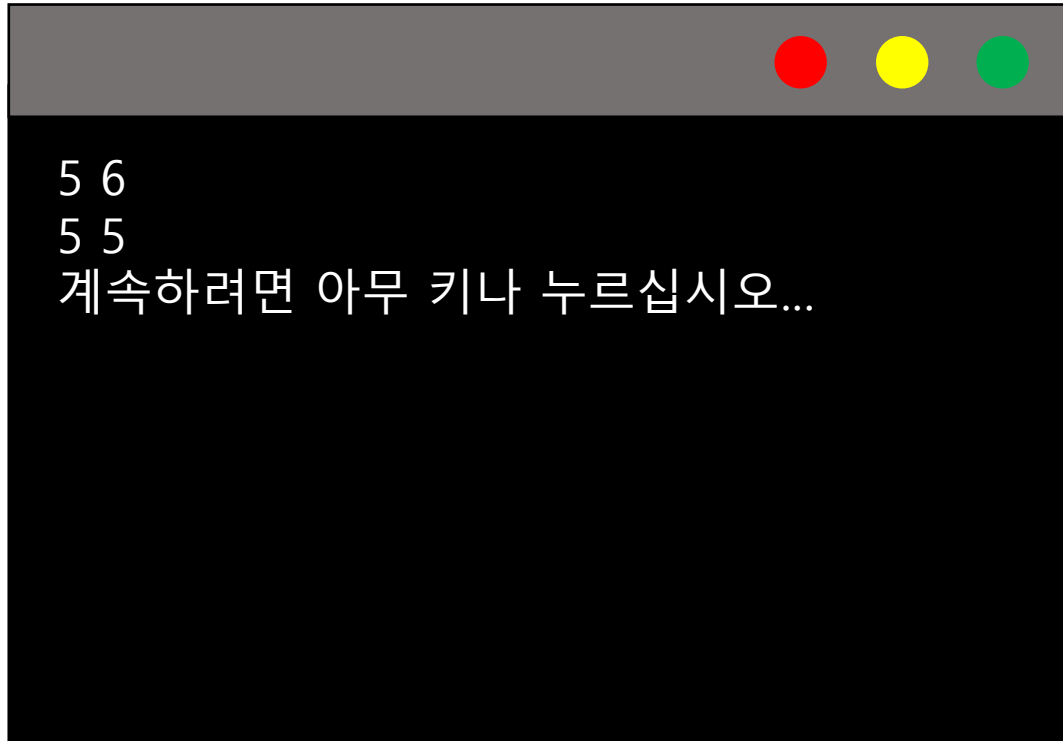
다음 예제를 통해 살펴봅시다

증감 연산자

```
1  #include<stdio.h>
2
3  int main(void) {
4      |
5      |     int a = 5;
6      |
7      |     printf("%d %d\n", a++, a);
8      |     printf("%d %d\n", --a, a);
9      |
10     |     return 0;
11     |
12 }
```

결과가 어떻게 나올까요 ..??
실행해보기 전에 어떻게 나올지 생각해 봅시다

증감 연산자



```
5 6  
5 5  
계속하려면 아무 키나 누르십시오...
```

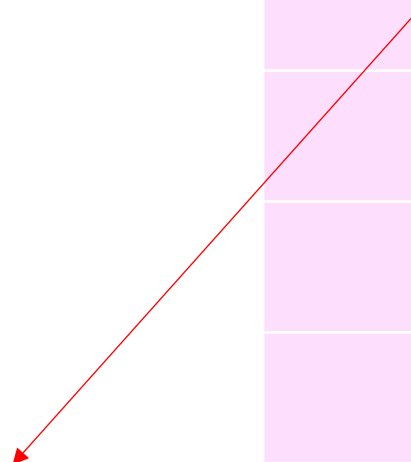
연산자의 위치를 보고 알아야겠죠?

교수님에 따라 다르지만
증감 연산자 부분은 소입설 시험에서
잘 나오는 부분이니 스스로 다른 예제들을
만들어보면서 확인해보세요!

비교 연산자

비교 연산자는
true or false를 리턴해요.

앞에서 나왔던
대입연산자(=)와 구분을 잘하자
(코딩할 때 자주 하는 실수!)

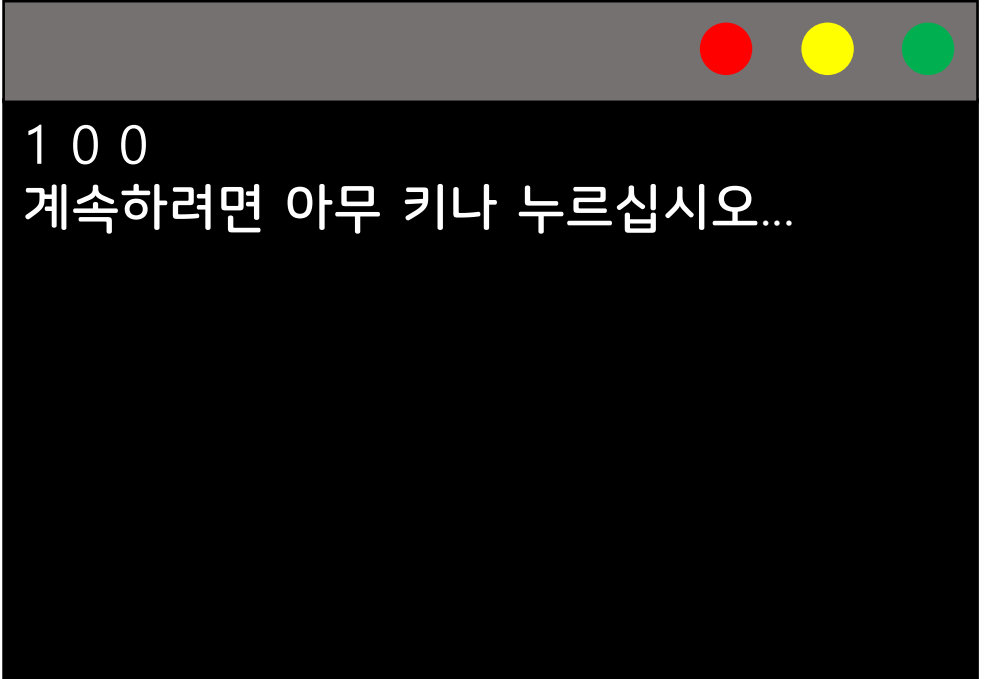


| 연산자 | 의미 |
|----------------------|------------|
| <code>a==b</code> | a와 b가 같은가 |
| <code>a!=b</code> | a와 b가 다른가 |
| <code>a>b</code> | a가 b보다 큰가 |
| <code>a<b</code> | a가 b보다 작은가 |
| <code>a>=b</code> | a가 b 이상인가 |
| <code>a<=b</code> | a가 b 이하인가 |

비교 연산자

Bool 자료형이 정의된 헤더파일을 선언해 줍시다

```
1 #include<stdio.h>
2 #include<stdbool.h>
3
4 int main(void) {
5     //
6     int a = 5, b = 5, c = 2;
7     bool b1 = (a == b);
8     bool b2 = (a == c);
9     bool b3 = (a != b);
10
11     printf("%d %d %d\n", b1, b2, b3);
12 }
```



1 0 0
계속하려면 아무 키나 누르십시오...

C언어에서는
True == 0을 제외한 모든 값
False == 0

논리 연산자

| 연산 | 의미 |
|---------------|--------------------------------|
| a&&b (AND) | a,b 둘다 참인가 |
| a b (OR) | a,b 둘 중 하나라도 참인가 |
| !a (NOT) | a가 true 면 false , false 면 true |

논리 연산자

논리곱(&&,AND)

| A | B | 결과값 |
|---|---|-----|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

논리합(||,OR)

| A | B | 결과값 |
|---|---|-----|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

논리 연산자

```
1  #include<stdio.h>
2  #include<stdbool.h>
3
4  int main(void) {
5      |
6      |     bool a = true;
7      |     bool b = true;
8      |     bool c = false;
9      |     printf("%d\n", a&&b);
10     |     printf("%d\n", a || c);
11     |     printf("%d\n", !a);
12     |     return 0;
13     | }
```

1
1
0

계속하려면 아무 키나 누르십시오...

연산자 우선순위

| 연산자 | 내용 |
|------------------|------------|
| (), [] | 괄호 |
| !, ++, -- | 부정/증감 연산자 |
| * / % | 곱셈/나눗셈 연산자 |
| + - | 덧셈/뺄셈 연산자 |
| < <= > >= | 관계 연산자 |
| == != | |
| && | 논리곱 연산자 |
| | 논리합 연산자 |
| ?: | 조건 연산자 |
| = += -= *= /= %= | 대입/할당 연산자 |

여기까지 연습문제를 풀어봅시다

백준 : www.acmicpc.net



BOJ 1000
A+B



BOJ 1001

A-B



BOJ 10998

AXB



BOJ 1008

A/B



BOJ 10869

사칙연산

Ch 2 . 조건문

조건문

조건문에서는 프로그래머가 명시한 Boolean 자료형 조건이 '참'인지 '거짓' 인지에 따라 계산이나 상황을 수행해요.

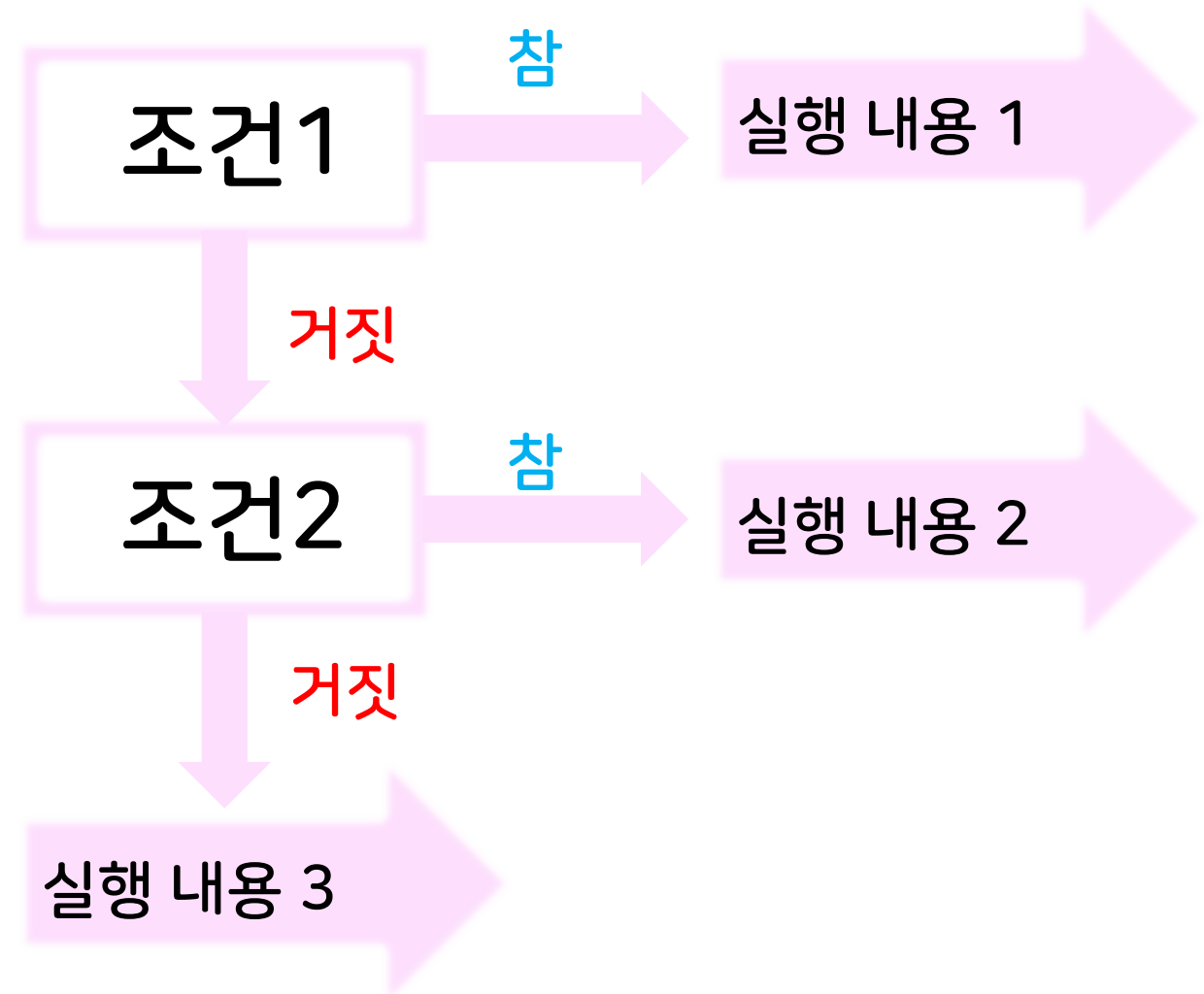


참 == true == 0을 제외한 모든 숫자 ->실행!
거짓 == false == 0

If문

```
#include <stdio.h>

int main()
{
    if(조건1){
        (실행 내용1);
    }
    else if(조건2){
        (실행 내용2);
    }
    else{
        (실행 내용3);
    }
    return 0;
}
```



If문

```
1 #include<stdio.h>
2 #include<stdbool.h>
3
4 int main(void) {
5     int a = 2;
6     if (a == 1) {
7         printf("%d\n", a * 2);
8     }
9     else if (a == 2) {
10        printf("%d\n", a * 3);
11    }
12    else {
13        printf("%d\n", a);
14    }
15    return 0;
16 }
17
```

6

계속하려면 아무 키나 누르십시오...

If문

```
#include <stdio.h>

int main()
{
    if(조건1){
        (실행 내용1);
    }
    if(조건2){
        (실행 내용2);
    }

    return 0;
}
```

조건1

참

실행 내용 1

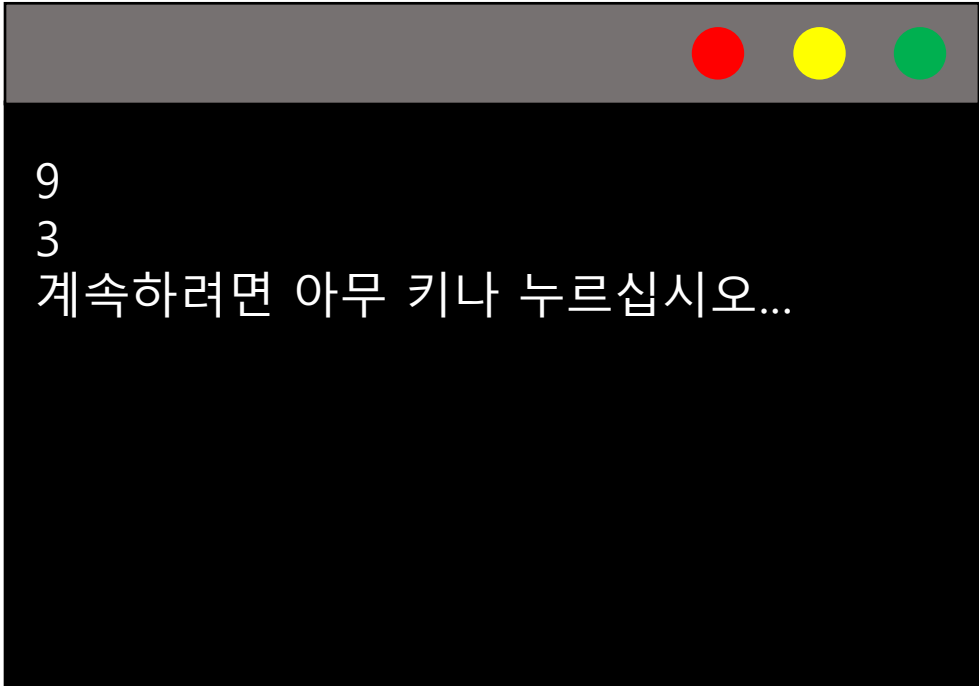
조건2

참

실행 내용 2

If문

```
1  #include<stdio.h>
2  #include<stdbool.h>
3
4  int main(void) {
5
6
7      int a = 3;
8      if (a * 3 == 9) {
9          printf("%d\n", a * 3);
10     }
11     if (a < 5) {
12         printf("%d\n", a);
13     }
14     return 0;
15 }
```



```
9
3
계속하려면 아무 키나 누르십시오...
```

If문도 배웠으니 풀고 지나갑시다



BOJ 2753

윤년



BOJ 1330

두 수 비교하기

switch 문

```
#include <stdio.h>

int main()
{
    switch (변수){
        case (변수가 만족시키는 조건 1):
            (실행 내용 1);
            break;
        case (변수가 만족시키는 조건 2):
            (실행 내용 2);
            break;
        default:
            (실행 내용 3);
            break;
    }
    return 0;
}
```

switch문은 if문과 같은 조건 제어문 입니당!

if문과 다른 점은 < , > 와 같은 부등식을 쓰지 못한다는 겁니다.

비교할 변수가 어떤 값을 가지냐에 따라 실행이 되는 거죠.

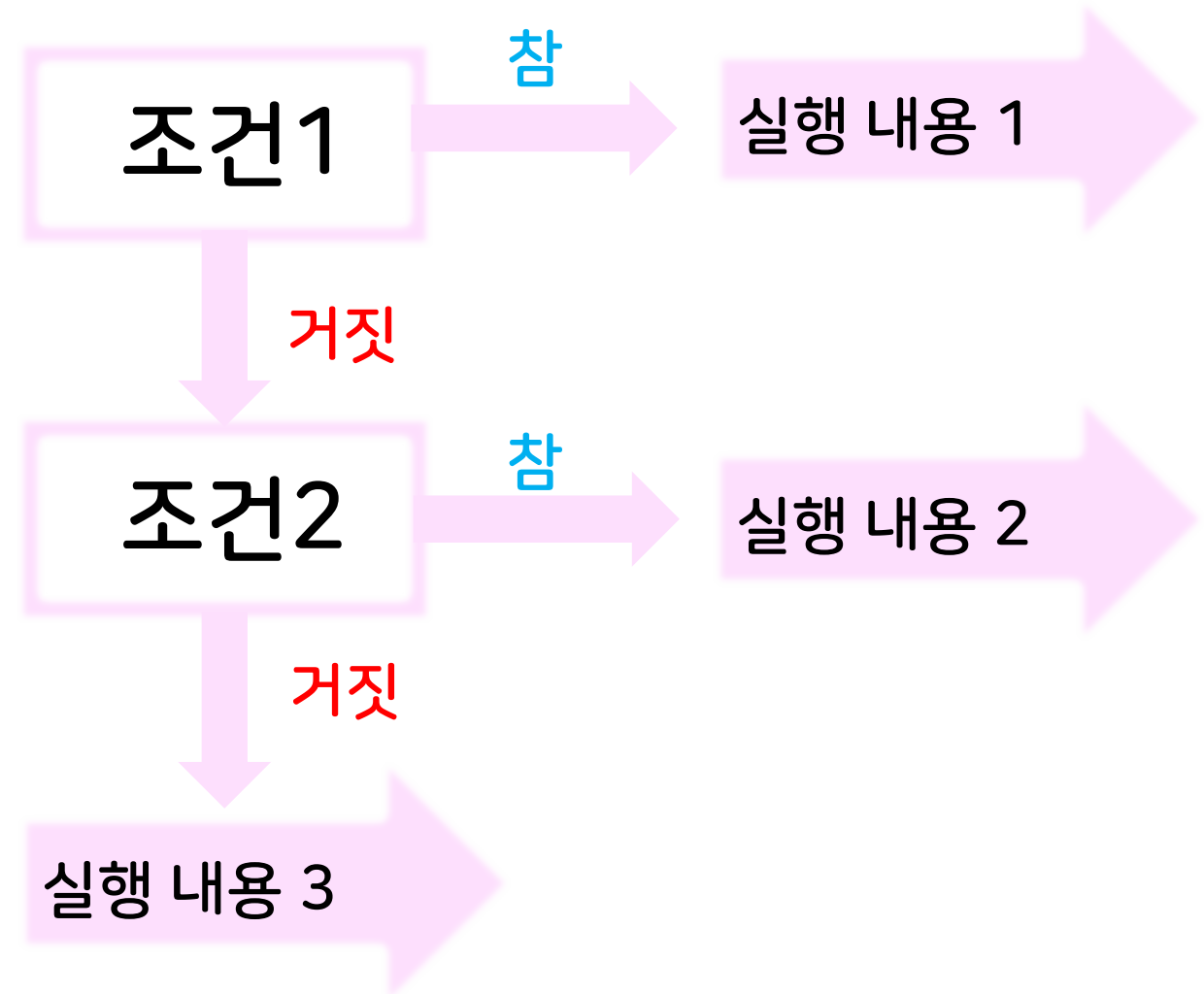
그래서 모든 switch문은 if문으로 바꿀 수 있어요.

상황에 따라 if문과 switch문 중에 골라서 코딩할 수 있겠죠?

switch 문

```
#include <stdio.h>

int main()
{
    switch (변수){
        case (변수가 만족시키는 조건 1):
            (실행 내용 1);
            break;
        case (변수가 만족시키는 조건 2):
            (실행 내용 2);
            break;
        default:
            (실행 내용 3);
            break;
    }
    return 0;
}
```



switch 문

```
#include <stdio.h>

int main()
{
    switch (변수){
        case (변수가 만족시키는 조건 1):
            (실행 내용 1);
            break;
        case (변수가 만족시키는 조건 2):
            (실행 내용 2);
            break;
        default:
            (실행 내용 3);
            break;
    }
    return 0;
}
```

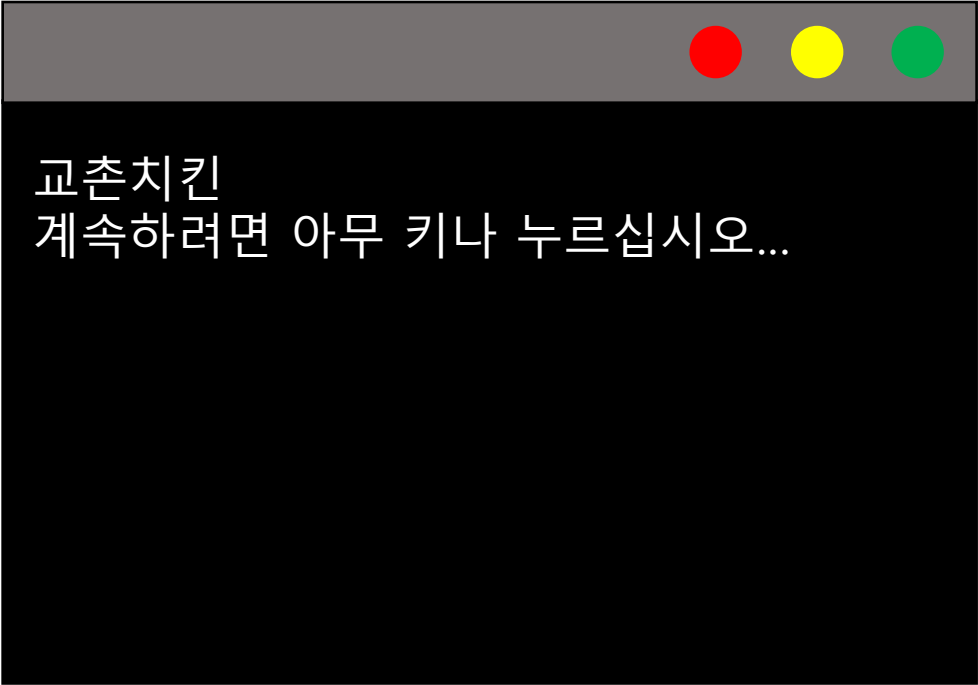
switch 문은 괄호 안의 변수의 값과 동일한 값을 가지는 case로 가서 실행 내용을 실행해요

만약 변수의 값과 동일한 값을 가지는 case가 없다면..?

default 문으로 가요!

switch 문

```
1  #include<stdio.h>
2
3  int main() {
4      int a = 1;
5      switch (a) {
6          case 0:
7              printf("네네치킨");
8              break;
9          case 1:
10             printf("교촌치킨");
11             break;
12          case 2:
13             printf("푸라닭 치킨");
14             break;
15          default:
16             printf("먹지마");
17             break;
18      }
19  }
```



교촌치킨
계속하려면 아무 키나 누르십시오...

If문의 중첩

```
1  #include<stdio.h>
2
3  int main(void) {
4      int a, b, c;
5      scanf("%d %d %d", &a, &b, &c);
6      if ((a == 2) && (b % 3 == 0) && (c != 5)) {
7          printf("%d %d %d", a, b, c);
8      }
9      else {
10         printf("0");
11     }
12     return 0;
13 }
```

If 조건문이 더 길다면 이렇게 코드를 짜는 것이 불편하지 않을까?

다르게 짜는 방법은 없을까?

If문의 중첩

```
1  #include<stdio.h>
2
3  int main(void) {
4
5      int a, b, c;
6      scanf("%d %d %d", &a, &b, &c);
7      if (a == 2) {
8          if (b % 3 == 0) {
9              if (c != 5) {
10                 printf("%d %d %d", a, b, c);
11             }
12             else {
13                 printf("0");
14             }
15         }
16         else {
17             printf("0");
18         }
19     }
20     else {
21         printf("0");
22     }
23     return 0;
24 }
```

앞의 코드와 이 코드는 같아요.

하지만, if 조건문의 조건이 무조건 짧다고 좋은 것도, if문이 여러 개 중첩된다고 해서 좋은 것도 아니에요.

상황에 따라 방법을 골라서 코딩합시다!

끝

수고하셨습니다 여러분~~~~