

2023/03/27

# APPDONG

## C언어 멘토링

2주차  
멘토 : 김민수



# Contents

**01. 형 변환(타입 캐스팅)**

**02. 조건문**

**03. 반복문**

**04. 문제 해결**

# ■ 형 변환(타입 캐스팅)

## 1. 형 변환 연산자

형 변환 연산자는 일시적으로 피연산자를 원하는 형태로 바꿔준다.  
실제 메모리의 피연산자의 값은 변하지 않는다.

`int a = 10;` 이라고 `a`를 정수로 선언했을 때, 일시적으로 실수값으로 변경하려면 `(double)a` 와 같이 작성하면 된다.

(자료형)피연산자

# 형 변환(타입 캐스팅)

```
1  #include <stdio.h>
2
3  int main() {
4      int a = 20, b = 3;
5      double res;
6
7      res = (double)a / (double)b;
8      printf("a = %d, b = %d\n", a, b);
9      printf("a / b = %f\n", res);
10
11     return 0;
12 }
```

정수와 정수의 연산 결과는 정수 값이 나오기 때문에 정확한 연산 결과를 얻으려면 일시적으로 실수형으로 변환을 해야한다.

Microsoft Visual Studio 디버그 콘솔

```
a = 20, b = 3
a / b = 6.666667
```

```
C:\Users\minsu\Desktop\studyC\Project1\
개).
이 창을 닫으려면 아무 키나 누르세요...
```

# ■ 형 변환(타입 캐스팅)

## 2. 자동 형 변환

컴파일 과정에서 피연산자 간의 형태가 다르면 형태를 일치시키는 작업을 수행한다.

기본 규칙은 크기가 작은 값이 크기가 큰 값으로 바뀌는 것이다.

ex) 정수와 실수의 연산에서 정수가 실수로 자동으로 변환되어 연산된다.

단, 대입 연산의 경우 무조건 좌변의 변수 형태에 맞게 저장된다.  
(메모리에 값이 저장되는 경우이기 때문)

데이터의 손실이 발생할 수 있기 때문에 유의해야한다.

# 형 변환(타입 캐스팅)

```
1  #include <stdio.h>
2
3  int main() {
4      int a = 20, b = 3;
5      double res;
6
7      res = (double)a / b;
8      printf("a = %d, b = %d\n", a, b);
9      printf("a / b = %f\n", res);
10
11     return 0;
12 }
```

a는 double, b는 int이기 때문에  
b의 형태가 double로 **자동 형 변환**된 후  
연산이 된다.

Microsoft Visual Studio 디버그 콘솔

```
a = 20, b = 3
a / b = 6.666667
```

```
C:\Users\minsu\Desktop\studyC\Project1\
개).
이 창을 닫으려면 아무 키나 누르세요...
```

# 형 변환(타입 캐스팅)

```
1 #include <stdio.h>
2
3 int main() {
4     int a;
5     double b;
6
7     b = 2.4;
8     a = b;
9
10    printf("%d\n", a);
11
12    return 0;
13 }
```

Microsoft Visual Studio 디버그 콘솔

```
2
C:\Users\minsu\Desktop\studyC\Project1\x64
).
이 창을 닫으려면 아무 키나 누르세요..._
```

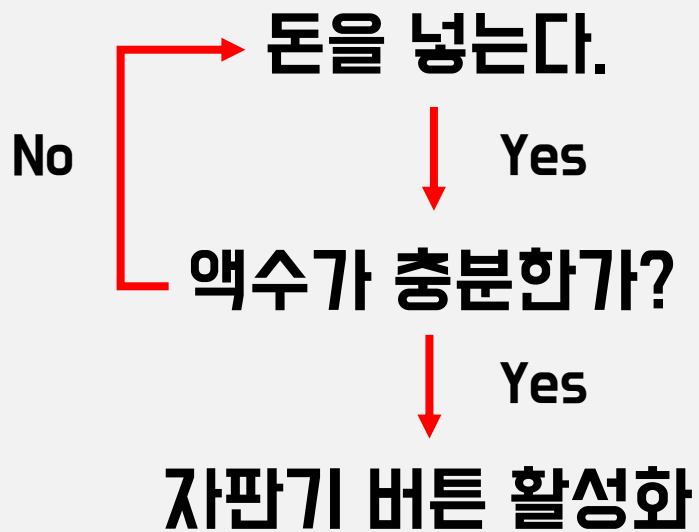
**double형 데이터를 int형 데이터로 대입하는 과정에서  
데이터 손실이 발생**

warning C4244: '=': 'double'에서 'int'(으)로 변환하면서 데이터가 손실될 수 있습니다.

# 조건문

상황에 따라 실행 하거나 실행하지 않아야 할 때 조건문을 통해 제어할 수 있다.

예시) 자판기





# 조건문

## 1. if문

```
if(조건식)
{
    내부 코드
}
```



조건식이 참이면 내부 실행  
거짓인 경우는 실행 안함

# 조건문

```
1  #include <stdio.h>
2
3  int main() {
4      double n, m;
5
6      printf("두 수를 입력하세요 >> ");
7      scanf_s("%lf %lf", &n, &m);
8
9      printf("%f를 %f로 나눈 결과 : %f\n", n, m, n / m);
10
11     return 0;
12 }
```

Microsoft Visual Studio 디버그 콘솔

두 수를 입력하세요 >> 10 3  
10.000000를 3.000000로 나눈 결과 : 3.333333  
C:\Users\minsu\Desktop\studyC\Project1\x64\Debug  
(x64) .  
이 창을 닫으려면 아무 키나 누르세요...

겉으로는 문제가 없어 보이는 코드

-> 실제로 실행도 잘 된다

# 조건문

```
1  #include <stdio.h>
2
3  int main() {
4      double n, m;
5
6      printf("두 수를 입력하세요 >> ");
7      scanf_s("%lf %lf", &n, &m);
8
9      printf("%f를 %f로 나눈 결과 : %f\n", n, m, n / m);
10
11     return 0;
12 }
```

Microsoft Visual Studio 디버그 콘솔

두 수를 입력하세요 >> 10 0  
10.000000를 0.000000로 나눈 결과 : inf

C:\Users\minsu\Desktop\studyC\Project1\  
개).  
이 창을 닫으려면 아무 키나 누르세요...

10 나누기 0을 해보면 이상한 결과가 나옴

-> 0을 나누는 것은 금지되어있음.

# 조건문

```
1  #include <stdio.h>
2
3  int main() {
4      double n, m;
5
6      printf("두 수를 입력하세요 >> ");
7      scanf_s("%lf %lf", &n, &m);
8
9      if (m == 0) {
10         printf("0으로 나눌 수 없습니다.");
11
12         return 0;
13     }
14
15     printf("%f를 %f로 나눈 결과 : %f\n", n, m, n / m);
16
17     return 0;
18 }
```

Microsoft Visual Studio 디버그 콘솔

두 수를 입력하세요 >> 10 0  
0으로 나눌 수 없습니다.  
C:\Users\minsu\Desktop\studyC\Project1\  
개).  
이 창을 닫으려면 아무 키나 누르세요...

**if문으로 조건을 걸어서  
오류를 피할 수 있다.**

# 조건문

## 2. 관계 연산자 & 논리 연산자

### \* 관계 연산자 \*

- 1)  $\geq$  : 좌변이 우변보다 크거나 같으면 참
- 2)  $>$  : 좌변이 우변보다 크면 참
- 3)  $\leq$  : 좌변이 우변보다 작거나 같으면 참
- 4)  $<$  : 좌변이 우변보다 작으면 참
- 5)  $==$  : 좌변과 우변이 같으면 참
- 6)  $!=$  : 좌변과 우변이 다르면 참

# 조건문

## 2. 관계 연산자 & 논리 연산자

### \* 논리 연산자 \*

- 1) &&(AND) : 좌변과 우변을 AND연산 (둘 다 true일 때 true)
- 2) ||(OR) : 좌변과 우변을 OR연산 (둘 중 하나가 true면 true)
- 3) !(NOT) : 참을 거짓으로, 거짓을 참으로 반전

# 조건문

```
1  #include <stdio.h>
2
3  int main() {
4      int n;
5
6      printf("시험 점수 입력 >> ");
7      scanf_s("%d", &n);
8
9      if (n >= 90) printf("학점 : A\n");
10     if (n >= 80 && n < 90) printf("학점 : B\n");
11     if (n >= 70 && n < 80) printf("학점 : C\n");
12     if (n >= 60 && n < 70) printf("학점 : D\n");
13     if (n < 60) printf("학점 : F\n");
14
15     return 0;
16 }
```

Microsoft Visual Studio 디버그 콘솔

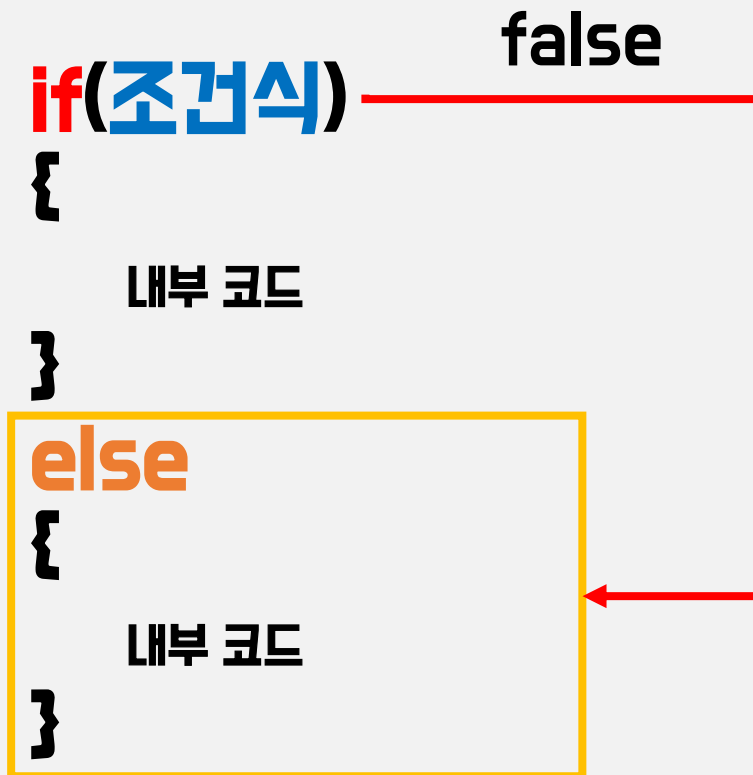
시험 점수 입력 >> 79  
학점 : C

C:\Users\minsu\Desktop\studyC\Project1\x  
개).  
이 창을 닫으려면 아무 키나 누르세요...

관계 연산자와 논리 연산자를 잘 활용하면  
복합적인 조건도 쉽게 처리할 수 있다.

# 조건문

## 3. if - else문



if문에서 조건에 해당되지 않을 경우  
else에서 받아서 처리를 한다.



# 조건문

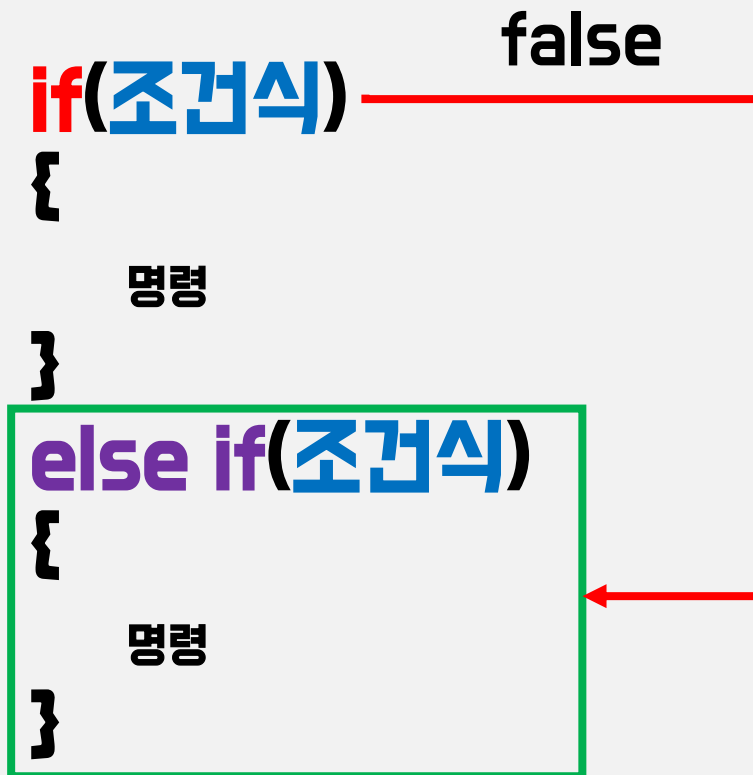
```
1  #include <stdio.h>
2
3  int main() {
4      double n, m;
5
6      printf("두 수를 입력하세요 >> ");
7      scanf_s("%lf %lf", &n, &m);
8
9      if (m == 0) {
10         printf("0으로 나눌 수 없습니다.");
11
12         return 0;
13     }
14
15     printf("%f를 %f로 나눈 결과 : %f\n", n, m, n / m);
16
17     return 0;
18 }
```

```
1  #include <stdio.h>
2
3  int main() {
4      double n, m;
5
6      printf("두 수를 입력하세요 >> ");
7      scanf_s("%lf %lf", &n, &m);
8
9      if (m == 0) {
10         printf("0으로 나눌 수 없습니다.");
11     }
12     else {
13         printf("%f를 %f로 나눈 결과 : %f\n", n, m, n / m);
14     }
15
16     return 0;
17 }
```

특정 경우를 제외한 나머지 경우들을  
어떻게 처리할 지 제어할 수 있다.

# 조건문

## 4. if - else if문



if문에서 조건에 해당되지 않을 경우  
else if로 넘어가서 조건을 확인한다.

else if는 중첩으로 계속 사용 가능하다.

# 조건문


```
1  #include <stdio.h>
2
3  int main() {
4      int n;
5
6      printf("정수 입력 >> ");
7      scanf_s("%d", &n);
8
9      if (n > 0) {
10         printf("n은 양의 정수\n");
11     }
12     else if (n == 0) {
13         printf("n은 0\n");
14     }
15     else {
16         printf("n은 음의 정수\n");
17     }
18
19     return 0;
20 }
```

if - else if - else를 사용해서  
조건을 묶어서 처리할 수 있다.

# 조건문

```
1  #include <stdio.h>
2
3  int main() {
4      int n;
5
6      printf("시험 점수 입력 >> ");
7      scanf_s("%d", &n);
8
9      if (n >= 90) printf("학점 : A\n");
10     if (n >= 80 && n < 90) printf("학점 : B\n");
11     if (n >= 70 && n < 80) printf("학점 : C\n");
12     if (n >= 60 && n < 70) printf("학점 : D\n");
13     if (n < 60) printf("학점 : F\n");
14
15     return 0;
16 }
```

```
1  #include <stdio.h>
2
3  int main() {
4      int n;
5
6      printf("시험 점수 입력 >> ");
7      scanf_s("%d", &n);
8
9      if (n >= 90) printf("학점 : A\n");
10     else if (n >= 80 && n < 90) printf("학점 : B\n");
11     else if (n >= 70 && n < 80) printf("학점 : C\n");
12     else if (n >= 60 && n < 70) printf("학점 : D\n");
13     else printf("학점 : F\n");
14
15     return 0;
16 }
```



# 조건문

```
if (조건1) {  
    명령1;  
} else {  
    if (조건2) {  
        명령2;  
    } else {  
        if (조건3) {  
            명령3;  
        } else {  
            ...  
        }  
    }  
}
```



```
if (조건1) {  
    명령1;  
} else if (조건2) {  
    명령2;  
} else if (조건3) {  
    명령3;  
} else {  
    ...  
}
```

**if - else 보다 간단하게 표현할 수 있다**

# 조건문

## 5. switch문

동일한 변수에 대해 비교, 조건문이 반복되는 경우  
if - else를 나열하는 것 보다 더 깔끔하고 효율적으로 표현할 수 있다.

변수 값에 따라 실행할 코드를 설계할 수 있다.

# 조건문

## switch문 기본 구조

```
switch ( 조건식 )
{
    case 상수식1:
        명령1;
        break;
    case 상수식2:
        명령2;
        break;
    default:
        명령3;
        break;
}
```

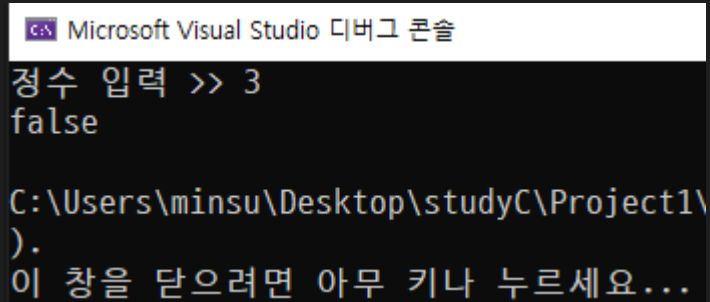
조건식에는 변수  
상수식에는 값이 들어간다. (이 때 값은 무조건 상수)

각 case마다 break;를 설정하지 않으면  
그 다음 case로 계속 넘어가서 실행되기 때문에  
break;를 설정해서 정상적으로 switch문을 종료해야 한다.

default는 if문에서 else와 같은 역할을 한다. (없어도 상관은 없음)

# 조건문

```
1  #include <stdio.h>
2
3  int main() {
4      int n;
5
6      printf("정수 입력 >> ");
7      scanf_s("%d", &n);
8
9      switch (n % 3) {
10         case 0:
11             printf("false\n");
12             break;
13
14         default:
15             printf("true\n");
16         }
17
18     return 0;
19 }
```



Microsoft Visual Studio 디버그 콘솔

정수 입력 >> 3  
false

C:\Users\minsu\Desktop\studyC\Project1\  
)  
이 창을 닫으려면 아무 키나 누르세요...

**default는 switch 블록에서 어느 위치에 있어도 결과는 같다.  
일반적으로 맨 마지막 위치에 추가한다.**



# 반복문

수 많은 반복적인 작업을 효율적으로 수행하기 위해 사용

일정 조건을 만족하는 동안 같은 실행문을 반복

반복문 종류

- for문
- while문
- do while문

# 반복문

## 1. for문

```
for ( 초기식; 조건식; 증감식 )  
{  
    명령;  
}
```

**초기식** -> 제어 변수를 초기화  
(반복문을 얼마나 반복할 지)

ex)  $i = 1;$

**조건식** -> 제어 변수가 만족해야 될 조건

ex)  $i \leq 10;$

**증감식** -> 1회 반복 수행 시 제어 변수의  
값을 설정

ex)  $i++;$

# 반복문

```
1  #include <stdio.h>
2
3  int main() {
4      int subject, score, sum = 0;
5      double average;
6
7      printf("입력할 과목의 개수 >> ");
8      scanf_s("%d", &subject);
9
10     for (int i = 0; i < subject; i++) {
11         printf("과목의 점수를 입력하세요 >> ");
12         scanf_s("%d", &score);
13         sum += score;
14     }
15
16     average = (double)sum / subject;
17
18     printf("평균 : %f\n", average);
19
20     return 0;
21 }
```

Microsoft Visual Studio 디버그 콘솔

```
입력할 과목의 개수 >> 3
과목의 점수를 입력하세요 >> 10
과목의 점수를 입력하세요 >> 20
과목의 점수를 입력하세요 >> 30
평균 : 20.000000
```

```
C:\Users\minsu\Desktop\studyC\Project1\xo
).
이 창을 닫으려면 아무 키나 누르세요...■
```

**i가 0 1 2 ... subject-1 일 때까지 반복 수행**  
**i가 subject가 되는 순간 for문 탈출**

# 반복문

## 2. while문

```
while ( 조건식 )  
{  
    명령;  
}
```

조건식 -> while문이 계속 반복될 조건이 들어감.  
조건식이 참(true)인 경우 계속 반복.  
거짓(false)인 경우 탈출.

**ex)  $i \leq 100$**

( $i$ 가 100이하 일 때 동안은 명령을 계속 반복해라)

# 반복문

```
1  #include <stdio.h>
2
3  int main() {
4      int subject, score, sum = 0;
5      double average;
6
7      printf("입력할 과목의 개수 >> ");
8      scanf_s("%d", &subject);
9
10     int count = subject;
11
12     while (count--> 0) {
13         printf("과목의 점수를 입력하세요 >> ");
14         scanf_s("%d", &score);
15         sum += score;
16     }
17
18     average = (double)sum / subject;
19
20     printf("평균 : %f\n", average);
21
22     return 0;
23 }
```

Microsoft Visual Studio 디버그 콘솔

```
입력할 과목의 개수 >> 3
과목의 점수를 입력하세요 >> 10
과목의 점수를 입력하세요 >> 20
과목의 점수를 입력하세요 >> 30
평균 : 20.000000
```

```
C:\Users\minsu\Desktop\studyC\Project1\src\main.c(12): warning C4013: 'scanf_s' used before definition
).
이 창을 닫으려면 아무 키나 누르세요..._
```

**count가 0이 되는 순간 탈출**  
(0 값은 false 나머지는 true이기 때문)

# 반복문

## 3. do while문

```
do  
{  
    명령;  
} while ( 조건식 )
```

먼저 명령을 1회 수행하고 나서 조건식이 참인지 검사  
조건식이 참이 아니더라도 명령이 1회 실행된다.

조건식이 참인지 먼저 검사를 하는 while과 차이가 있다.

# 반복문

```
1  #include <stdio.h>
2
3  int main() {
4      int subject, score, sum = 0;
5      double average;
6
7      printf("입력할 과목의 개수 >> ");
8      scanf_s("%d", &subject);
9
10     int count = subject;
11
12     do {
13         printf("과목의 점수를 입력하세요 >> ");
14         scanf_s("%d", &score);
15         sum += score;
16     } while (count-- > 1);
17
18     average = (double)sum / subject;
19
20     printf("평균 : %f\n", average);
21
22     return 0;
23 }
```

C:\ Microsoft Visual Studio 디버그 콘솔

```
입력할 과목의 개수 >> 3
과목의 점수를 입력하세요 >> 10
과목의 점수를 입력하세요 >> 20
과목의 점수를 입력하세요 >> 30
평균 : 20.000000
```

```
C:\Users\minsu\Desktop\studyC\Project1\src>
).
이 창을 닫으려면 아무 키나 누르세요...■
```

**명령을 먼저 1회 수행한 후 조건 검사  
count가 1이 되는 순간 탈출**

# 반복문

## 4. break와 continue

**break문은 반복문 안에서 반복을 즉시 끝낼 때 사용**

→ 조건에 따라 중간에 임의로 반복문을 탈출할 때

**continue는 반복문의 일부를 건너뛸 때 사용**

→ 반복문 블록을 탈출하는 것은 아님. 현재 단계를 건너뛰고 그 다음 단계로 이동



# 숫자 맞추기 게임 (UP & DOWN) 만들어 보기

## 1. 게임 시작 설정

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4
5  int main() {
6      int option, number, answer, front, tail;
7
8      printf("숫자 맞추기 게임에 오신 것을 환영합니다~\n");
9      printf("-----\n");
10     printf("게임 옵션\n");
11     printf("0 : 게임 종료, 1 : 게임 시작\n");
12 }
```

난수 생성을 위해 `stdlib.h` 추가

시드 값을 시간으로 설정해서 랜덤 설정을 위해 `time.h` 추가

화면에 출력될 게임 옵션 설정

# 숫자 맞추기 게임 (UP & DOWN) 만들어 보기

## 1. 게임 시작 설정

```
13 while (1) {  
14     printf("옵션을 선택하세요 >> ");  
15     scanf_s("%d", &option);  
16  
17     if (option == 0 || option == 1) break;  
18     else printf("입력 오류. 다시 입력하세요\n");  
19 }  
20
```

옵션 이외의 값이 입력되었을 때 예외처리 후 재 입력 유도

# 숫자 맞추기 게임 (UP & DOWN) 만들어 보기

## 2. option == 1 (게임 시작 옵션을 선택했을 때)

```
21  if (option) {  
22      printf("*** 숫자 맞추기 게임 시작 ***\n");  
23  
24      front = 1;  
25      tail = 100;      숫자 범위 초기화  
26  
27      srand((unsigned int)time(NULL));      시드 설정  
28      number = rand() % 100 + 1;  
29  
                                     1 ~ 100 사이 난수 생성
```

# 숫자 맞추기 게임 (UP & DOWN) 만들어 보기

## 2. option == 1 (게임 시작 옵션을 선택했을 때)

```
30 while (1) {  
31     printf("%d ~ %d 사이 숫자를 맞춰주세요\n", front, tail);  
32     printf("숫자 입력 >> ");  
33     scanf_s("%d", &answer);
```

사용자에게 숫자 입력 받음

```
35 if (answer < front || answer > tail) {  
36     printf("입력 오류. 다시 입력하세요.\n");  
37     continue;  
38 }  
39
```

범위 외의 숫자 입력 시. 재 입력 유도

# 숫자 맞추기 게임 (UP & DOWN) 만들어 보기

## 2. option == 1 (게임 시작 옵션을 선택했을 때)

```
40     if (answer == number) {
41         printf("정답입니다. 게임을 종료합니다.\n");
42         break;
43     }
44     else if (answer > number) {
45         printf("오답입니다. 다운(down)\n");
46         tail = answer;
47         continue;
48     }
49     else {
50         printf("오답입니다. 업(up)\n");
51         front = answer;
52         continue;
53     }
54 }
55 }
```

정답인 경우 게임 종료 (break로 탈출)

사용자 입력이 정답보다 높은 경우  
tail을 사용자 입력으로 설정 후 continue

사용자 입력이 정답보다 낮은 경우  
front을 사용자 입력으로 설정 후 continue

# 숫자 맞추기 게임 (UP & DOWN) 만들어 보기

## 3. option == 0 (게임 시작 옵션을 선택했을 때)

```
56  else {  
57      printf("게임 종료\n");  
58  }  
59
```

게임 종료 메시지 화면에 출력

## 4. 프로그램 종료

```
60  return 0;  
61  }
```

# 문제 해결

백준 단계별로 풀어보기 2단계, 3단계 (<https://www.acmicpc.net/step>)

“조건문” 1번~7번  
“반복문” 1번~12번

단계	제목	설명
1	입출력과 사칙연산	입력, 출력과 사칙연산을 연습해 봅시다. Hello World!
2	조건문	if 등의 조건문을 사용해 봅시다.
3	반복문	for, while 등의 반복문을 사용해 봅시다.
4	1차원 배열	배열을 사용해 봅시다.
5	문자열	문자열을 다루는 문제들을 해결해 봅시다.
6	심화 1	지금까지의 프로그래밍 문법으로 더 어려운 문제들을 풀어봅시다.
7	2차원 배열	배열 안에 배열이 있다면 어떨까요? 2차원 배열을 만들어 봅시다.

해결 못한 문제는 다음 멘토링 시간 전까지 해오기

