인트아이 정연한

2022-2 인트아이 C++ 수업 자료

C++ 기초 3강 반복문, 조건문

CONTENTS

같은 작업을 여러번 시도하려면

기본적으로 같은 작업을 여러번 시도한다 했을 때, 직관적으로는 단순히 해당 작업을 하는 코드를
여러번 작성하면 된다고 생각할 것이다.
그러나 C++에는 이런 귀찮은 작업을 대신 해주는 문법이 있는데, 이름하야 반복문이다.
이름 그대로 반복을 위해 사용되는 문법인데, 기본적으로 **for, while, do-whlie**이 있다.
아래는 반복문을 왜 써야 하는지에 대한 예시이다.

```
// 만약 반복문이 없다면...
void 구구단(int 단) {
    cout << 단 << " * 1 = " << 1 * 단 << endl;
    cout << 단 << " * 2 = " << 2 * 단 << endl;
    cout << 단 << " * 3 = " << 3 * 단 << endl;
    cout << 단 << " * 4 = " << 4 * 단 << endl;
    cout << 단 << " * 5 = " << 5 * 단 << endl;
    cout << 단 << " * 6 = " << 6 * 단 << endl;
    cout << 단 << " * 7 = " << 7 * 단 << endl;
    cout << 단 << " * 8 = " << 8 * 단 << endl;
    cout << 단 << " * 9 = " << 9 * 단 << endl;
}
```

for문으로 구구단을 만들어보자

여러 반복문 중, for문은 순차 반복에 용이하다.
아래처럼 코드를 작성하고, 실행을 시켜보면 오른쪽 사진처럼 구구단 2단이 출력되게 된다.

코드를 분석해보면, for문은 for(선언부; 조건부; 증감부) 로 나뉘게 되는데,
선언부: 반복을 위한 변수를 선언하는 부분
조건부: 선언부에서 선언한 변수를 통해 반복을 할지 말지를 결정하는 조건을 체크한다.
증감부: for문의 내용을 다 실행한 후, 반복을 위한 변수를 증가 또는 감소 시키는 역할을 한다.

```cpp
int main() {
    // i나 j같은 반복을 위한 변수를 통해 반복 조건을 체크한다.
    for (int i = 1; i < 10; ++i) {
        cout << 2 << " * " << i << " = " << 2 * i << endl;
    }
    return 0;
}
```

```
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
```

코드를 분석해보자

코드를 보면 for (int i = 1; i < 10; ++i) 라고 되어있는데,
우선 i 를 선언하고 1로 초기화 후, 조건을 검사한다.
이때 i는 1이기 때문에 i < 10 이라는 조건에 부합하고 따라서 for 문 안의 내용을 실행시킨다.
그리고 for 문 안의 내용을 다 실행시켰다면, 다시 맨 위로 올라와 ++i를 통해 i 의 값을 1 증가시키고,
다시 조건을 검사해준다.
이때, i는 2로, 여전히 10 미만이므로, 다시 한 번 반복하고, 이것을 9번 반복해준다.

정리하자면, i 선언 -> 조건 검사 -> 본문 실행 -> i값 증가 -> 조건 검사 -> 본문 실행 -> ... -> 조건 검사 -> 탈출

```cpp
int main() {
    // i나 j같은 반복을 위한 변수를 반복자라 한다.
    for (int i = 1; i < 10; ++i) {
        cout << 2 << " * " << i << " = " << 2 * i << endl;
    }
    return 0;
}
```

```
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
```

while문은 뭐지?

while 문은 while (조건부) 의 형태로 이루어져있으며, 보면 알겠지만 for문에서 선언부와 증감부가 빠진 형태이다.
while 문 역시 조건부의 내용이 true 일 때 반복문의 본문을 실행시키는데,
for 문이 순차 반복이라면, while 문은 반복조건이 좀 더 강조 된, 조건 반복이라 볼 수 있다.
아래의 코드를 보면 반복을 위한 변수가 선언되지도 않았는데도 잘만 반복이 되는 것을 볼 수 있다.

```cpp
// while (조건부)

int main() {
    int a = 0;
    cout << "10을 입력해주세요" << endl;
    cin >> a;

    // a가 10이 아닐 때, 반복
    while (a != 10) {
        cout << "a가 10이 아닙니다. 다시 입력해주세요." << endl;
        cin >> a;
    }
    return 0;
}
```

```
10을 입력해주세요
4
a가 10이 아닙니다. 다시 입력해주세요.
8
a가 10이 아닙니다. 다시 입력해주세요.
6
a가 10이 아닙니다. 다시 입력해주세요.
10
```

do-while문은 또 뭐지?

do-while문은 while문과 비슷하지만, 조건을 검사하지 않고, 최소 한 번은 실행하는 반복문으로
반복문의 본문에서 반복 조건에 해당하는 값이 변경되는 경우 등 아주 제한적인 경우에 사용 된다.
오른쪽의 코드를 보면, 문자열을 입력받고, 그 문자열 안의 각 문자를 소문자 -> 대문자, 대문자 -> 소문자
로 바꾸는 동작을 하고 있는데, while 조건을 보니 문자 != '₩0' 가 조건으로 들어와 있는 것을 볼 수 있다.

이것은 문자열 안의 특정 문자가 ₩0 이라는 문자가 아닐 때 반복을 하겠다 라는 뜻인데,
₩0는 제어문자라 불리는 특수한 문자 중 하나인 NULL문자로 ₩와 0이 각각 따로가 아니라
₩0 자체가 하나의 문자이다.
여기서는 폰트의 차이로 역슬래시가 아니라 원화기호로 표시되지만, 실제로는 해당 기호는 역슬래시가 맞다.

아무튼 ₩0은 문자열의 끝을 나타내는 제어 문자이기 때문에, 저 조건은 다시말해
문자열의 끝에 도달하면 반복을 종료하겠다. 라는 뜻이 된다.

```
int main() {
    string 문자열 = "";
    char 문자 = '\0'; // 문자를 입력 받을 변수 선언
    int i = 0; // 반복을 위한 반복자

    cout << "문자열 입력 : ";
    //cin >> 문자열; // 띄어쓰기는 포함 안함
    getline(cin, 문자열); // 띄어쓰기를 포함 함

    do {
        문자 = 문자열[i];

        if (문자 >= 'A' && 문자 <= 'Z')
            문자열[i] = 소문자로_변환(문자);
        else if (문자 >= 'a' && 문자 <= 'z')
            문자열[i] = 대문자로_변환(문자);
        else
            문자열[i] = 문자;
        i++;
    } while (문자 != '\0'); // NULL문자

    cout << "변환된 결과 ==> " << 문자열 << endl;
}
```

```
문자열 입력 : abCDeFg
변환된 결과 ==> ABcdEfG
```

조건분기를 만들어내는 조건문

C++에서 조건문은 두가지가 있는데, 하나는 if문이고, 나머지 하나는 switch-case문이다.
조건문은 조건분기를 만들어내는데,
조건 분기란, 특정한 조건에 부합하는 경우, 이하에 속하는 지시사항들의 실행 여부를 결정하는 일을 의미한다.
예를 들어, 내가 차를 타고 가고 있다고 가정해보자. 이때, 신호등이 있다면 신호등의 신호에 따라 행동을 달리할 것이다.

아래의 코드를 보면 신호등의 값에 따라 차가 갈지 말지를 결정하는 조건 분기가 있다.
신호등이 빨간불이라면 멈출 것이고, 빨간불이 아니라 노란불이라면 속도를 줄일 것이고,
빨간불도, 노란불도 아니라면 원래 속도로 지나갈 것이다.

```cpp
int main() {
    int 신호등 = 빨간불;

    if (신호등 == 빨간불) {
        cout << "차를 멈췄습니다." << endl;
    }
    else if (신호등 == 노란불) {
        cout << "속도를 줄여 천천히 지나갑니다." << endl;
    }
    else {
        cout << "원래 속도로 지나갑니다." << endl;
    }
    return 0;
}
```

```
차를 멈췄습니다.
```

switch-case문은 뭐지?

이번에도 신호등을 예시로 들었다.
switch문은 특이하게 조건식에 오는 값이 어떤 값인지에 따라서 케이스가 나뉘게되는데,
신호등의 값이 빨간불이라면, case 빨간불에 멈출 것이고,
신호등의 값이 노란불이라면, case 노란불에 멈출 것이고,
신호등의 값이 빨간불도 노란불도 아니라면 default에 멈출것이다.
추가로, case는 정수값만 올 수 있는데, 지금 보기에는 정수가 아니라 웬 변수가 들어가 있는 것 처럼 보인다.
이것은 enum을 쓴 것인데, 이것은 추후 class를 나가면서 같이 배울 예정이다.

```
int main() {
    int 신호등 = 노란불;

    switch (신호등) {
    case 빨간불:
        cout << "차를 멈췄습니다." << endl;
        break;
    case 노란불:
        cout << "속도를 줄여 천천히 지나갑니다." << endl;
        break;
    default:
        cout << "원래 속도로 지나갑니다." << endl;
        break;
    }

    return 0;
}
```

속도를 줄여 천천히 지나갑니다.

백준 10871번 X보다 작은 수

문제

정수 N개로 이루어진 수열 A와 정수 X가 주어진다. 이때, A에서 X보다 작은 수를 모두 출력하는 프로그램을 작성하시오.

입력

첫째 줄에 N과 X가 주어진다. (1 ≤ N, X ≤ 10,000)
둘째 줄에 수열 A를 이루는 정수 N개가 주어진다. 주어지는 정수는 모두 1보다 크거나 같고, 10,000보다 작거나 같은 정수이다.

출력

X보다 작은 수를 입력받은 순서대로 공백으로 구분해 출력한다. X보다 작은 수는 적어도 하나 존재한다.

예제 입력 1 복사	예제 출력 1 복사
10 5 1 10 4 9 2 3 8 5 7 6	1 4 2 3