

프로그래밍 언어 - C++

최백준 choi@startlink.io

C++

getline

C++

```
getline(cin, s);
```

- 한 줄 다 입력 받기

String

Scanf/printf

Cin / Cout

Cin >> x;

Cout << x;

setprecision

C++

```
#include <iomanip>
#include <iostream>
using namespace std;
int main() {
    double f = 3.14159265358979;
    cout << setprecision(5) << f << '\n';
    cout << setprecision(8) << f << '\n';
    cout << setprecision(10) << f << '\n';
    return 0;
}
```

3.1416 5자리 자르기
3.1415927 8자리 자르기
3.141592654 10자리 자르기

124

10

setprecision

C++

```
#include <iomanip>
#include <iostream>
using namespace std;
int main() {
    double f = 3.14159265358979;
    cout << fixed << setprecision(5) << f << '\n';
    cout << fixed << setprecision(8) << f << '\n';
    cout << fixed << setprecision(10) << f << '\n';
    return 0;
}
```

3.14159 5자리
3.14159265 8자리
3.1415926536 10자리

endl vs '\n'

C

ios_base::sync_with_stdio
(false);

6

- 줄을 바꿀 때, endl을 쓸 것인가 '\n'을 쓸 것인가?
- endl은 줄 바꿈을 출력하고, 스트림을 flush시키는 기능도 포함되어 있기 때문에
- '\n'보다 느리다.

- N찍기 문제: <https://www.acmicpc.net/problem/2741>

- `cout << i << endl;` (4272MS) 4.2초
 - `cout << i << '\n';` (36MS) 36MS 0.036초
 - `printf("%d\n", i);` (20MS)
- 100km

auto

C++11

auto

C++

- 컴파일러가 타입을 추론해서 타입을 결정한다
- 변수의 타입을 명확하게 알 수 있어야 한다.
- 아래 코드는 컴파일 에러

```
auto a,b;
```

```
cin >> a >> b;
```

```
cout << a + b << '\n';
```


auto

C++

- 컴파일러가 타입을 추론해서 타입을 결정한다
- 변수의 타입을 명확하게 알 수 있어야 한다.
- 아래 코드는 올바른 코드

```
auto a = 0, b = 0;
```

```
cin >> a >> b;
```

```
cout << a + b << '\n';
```

a = "abc";

b = 3.14

int

auto

C++

- 이터레이터를 사용할 때 매우 편리한다

```
map<pair<int, int>, vector<pair<int, string>>> d;
for (map<pair<int, int>, vector<pair<int, string>>>::iterator it = d.begin();
     it != d.end(); ++it) {
}
```

- 이런 코드를

```
map<pair<int, int>, vector<pair<int, string>>> d;
for (auto it = d.begin(); it != d.end(); ++it) {
}
```

- 이렇게 줄일 수 있다.

for each

Range-based for

Range-based for

C++

```
vector<int> a = {1, 2, 3, 4, 5};
```

```
for (int i=0; i<a.size(); i++) {  
    cout << a[i] << ' '  
}
```

```
cout << '\n';
```

```
for (int x : a) {  
    cout << x << ' '  
}  
cout << '\n';
```

1 2 3 4 5

Range-based for

C++

```
vector<pair<int,int>> a = {{1, 2}, {3, 4}, {5, 6}};
```

```
for (int i=0; i<a.size(); i++) {  
    cout << a[i].first + a[i].second << ' ' ;  
}
```

```
cout << '\n';
```

```
for (auto &p : a) {  
    cout << p.first + p.second << ' ' ;  
}
```

```
cout << '\n';
```

Range-based for

C++

```
int sum = 0;  
for (auto x : {1, 2, 3, 4}) {  
    sum += x;  
}  
cout << "sum = " << sum << "\n";
```

Handwritten annotations: A box around the range-based for loop. Arrows point from the elements 1, 2, 3, and 4 in the initializer list to the variable `x` in the loop body. The number `10` is written next to the closing brace of the loop.

```
int a[] = {1, 2, 3, 4, 5};
```


```
sum = 0;  
for (auto x : a) {  
    sum += x;  
}  
cout << "sum = " << sum << "\n";
```

Handwritten annotation: The number `15` is written next to the closing brace of the loop.

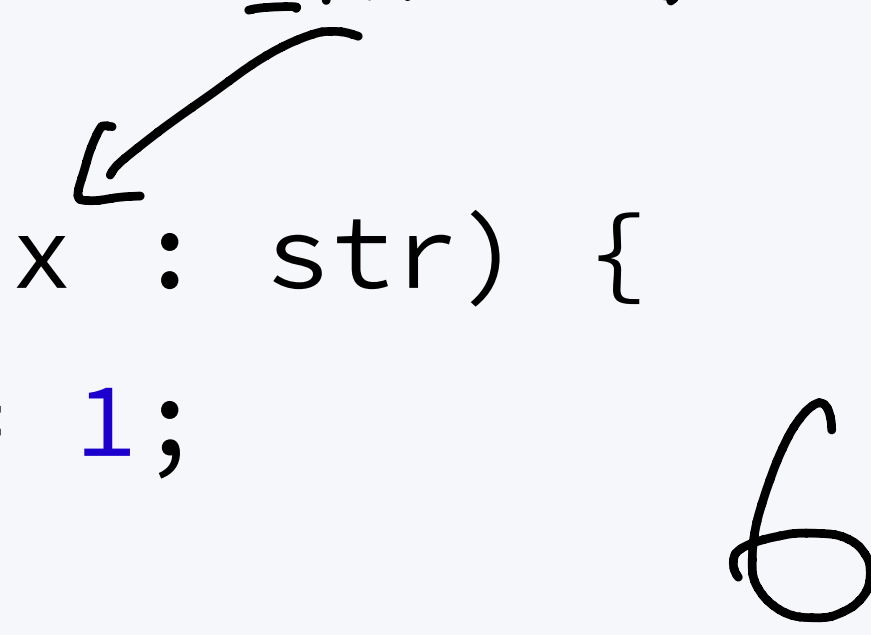
Range-based for

C++

```
const char cstr[] = "string";  
sum = 0;  
for (auto x: cstr) {  
    sum += 1;  
}  
cout << "sum = " << sum << '\n';
```



```
string str = "string";  
sum = 0;  
for (auto x: str) {  
    sum += 1;  
}  
cout << "sum = " << sum << '\n';
```



초기화 리스트

초기화 리스트

C++

```
vector<int> a;  
a.push_back(1);  
a.push_back(3);  
a.push_back(7);  
a.push_back(13);  
a.push_back(50);
```

```
vector<int> a = {1, 3, 7, 13, 50};
```

초기화 리스트

C++

```
struct Person {  
    string name;  
    int age;  
};
```

```
set<int> s = {1, 2, 3, 4, 5};
```

```
map<int, string> m = { {20, "a"}, {10, "hi"} };
```

```
Person p = {"you", 20};
```

key Val

```
map<int, vector<pair<int, int>>> m2 = {  
    {10, {{1, 2}, {3, 4}}},  
    {20, {{5, 6}, {7, 8}, {9, 10}}}  
};
```

람다 함수

C++

Lambda Function

오류
정답

람다 함수

C++

[캡처] (함수 인자) {함수 내용}

```
int sum(int x, int y) {  
    return x + y;  
}
```

```
cout << sum(1, 2) << '\n';
```

```
cout << [](int x, int y) {  
    return x + y;  
}(1, 2) << '\n';
```

람다 함수

C++

[캡처] (함수 인자) {함수 내용}

```
auto sum2 = [] (int x, int y) {  
    return x + y;  
};
```

```
cout << sum2(1, 2) << '\n';
```

생일 출력하기

22

<https://www.acmicpc.net/problem/2555>

```
#include <iostream>
using namespace std;
int main() {
    auto print = []{
        cout << "10/14" << "\n";
    };
    print();
}
```

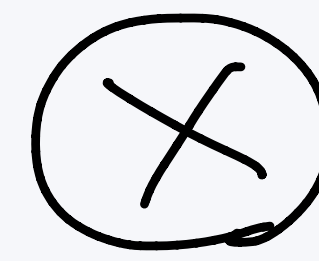
X보다 작은 수

<https://www.acmicpc.net/problem/10871>

```
int n, x;
cin >> n >> x;
auto is_less = [&](int number) {
    return number < x;
};

for (int i=0; i<n; i++) {
    int num;
    cin >> num;
    if (is_less(num)) {
        cout << num << ' ';
    }
}

cout << '\n';
```



$N \gg 1$

람다 함수

C++

[캡처] (함수 인자) {함수 내용}

캡처에 &를 넣으면 선언하는 시점에서 바깥에 있는 변수를 모두 사용할 수 있다

&x와 같이 어떤 변수를 사용할 것인지 적을 수도 있다

&는 참조이고, =는 값 복사이다

여러 개는 ,를 이용할 수 있다

람다 함수

C++

```
int x = 10;
```

```
int y = 20;
```

```
auto f = [&x, y]() {
```

```
    x += 10;
```

```
    //y += 10;
```

```
};
```

```
cout << "x = " << x << ", y = " << y << "\n";
```

```
f();
```

```
cout << "x = " << x << ", y = " << y << "\n";
```

```
f();
```

```
cout << "x = " << x << ", y = " << y << "\n";
```

10

20

20

20

30

20

람다 함수

C++

앞 페이지에서 주석을 해제하면 컴파일 에러를 받게 되는데

컴파일을 하려면

`auto f = [&x,y]() mutable {` 로 선언은 바꿔야 하지만

실행 결과는 달라지지 않는다

람다 함수

C++

함수의 변수 타입은 `#include <functional>` 에 선언되어 있다

`function<리턴타입(콤마로 구분한 인자의 타입들)>`

람다 함수

C++

```
function<void()> print = [] {  
};
```

```
function<void(int)> print2 = [](int x) {  
};
```

```
function<int(int, int)> sum = [](int x, int y) {  
    return x+y;  
};
```

피보나치 수 5

<https://www.acmicpc.net/problem/10870>

```
#include <functional>
#include <iostream>
using namespace std;
int main() {
    int n;
    cin >> n;
    function<int(int)> f = [&](int n) {
        if (n <= 1) return n;
        else return f(n-1) + f(n-2);
    };
    cout << f(n) << '\n';
    return 0;
}
```

사칙연산

<https://www.acmicpc.net/problem/10869>

```
vector<function<int(int,int)>> d;  
d.push_back([](int x, int y) {  
    return x + y;  
});  
d.push_back([](int x, int y) {  
    return x - y;  
});  
d.push_back([](int x, int y) {  
    return x * y;  
});  
d.push_back([](int x, int y) {  
    return x / y;  
});
```

x, y

$x + y$

$x - y$

$x * y$

x / y

$x \% y$

사칙연산

<https://www.acmicpc.net/problem/10869>

```
d.push_back([](int x, int y) {  
    return x % y;  
});
```

```
for (auto &f : d) {  
    cout << f(a, b) << '\n';  
}
```