

# C++ Programming

1<sup>st</sup> Study: From C to C++ (1/4)

- printf, scanf → cin, cout
- bool type
- auto keyword

C++ Korea 옥찬호 (utilForever@gmail.com)



printf, scanf → cin, cout

Changed Input / Output Method from C to C++

# printf, scanf → cin, cout

- 헤더 파일의 변화: `stdio.h` → `iostream` (Input / Output Stream)
- 입출력 함수의 변화
  - 출력: `printf` → `std::cout`
    - `cout`은 삽입 연산자 `<<`를 사용해 무언가를 콘솔 화면에 출력
  - 입력: `scanf` → `std::cin`
    - `cin`은 삽입 연산자 `>>`를 사용해 콘솔로부터 무언가를 저장
  - `<<`나 `>>`를 화살표 방향이라고 생각하면 쉽게 이해 가능
  - 입력이나 출력 형식을 지정하지 않고 사용 (`%d`, `%c` 등)

# printf → cout: Example

C

```
#include <stdio.h>

int main()
{
    printf("Hello, World!");

    return 0;
}
```

C++

```
#include <iostream>

int main()
{
    std::cout << "Hello, World";

    return 0;
}
```

# printf → cout: Example

C

```
#include <stdio.h>

int main()
{
    int x = 10, y = 20;

    printf("x = %d, y = %d\n", x, y);

    return 0;
}
```

C++

```
#include <iostream>

int main()
{
    int x = 10, y = 20;

    std::cout << "x = " << x
               << ", y = " << y << std::endl;

    return 0;
}
```

# scanf → cin: Example

C

```
#include <stdio.h>

int main()
{
    int x = 0;

    printf("Input the number: ");
    scanf("%d", &x);

    printf("x = %d\n", x);

    return 0;
}
```

C++

```
#include <iostream>

int main()
{
    int x = 0;

    std::cout << "Input the number:";
    std::cin >> x;

    std::cout << "x = " << x
              << std::endl;

    return 0;
}
```

# bool type

New type in C++

# bool type

- C에서는 조건문의 참, 거짓을 표현하기 위해 int를 사용
  - 참 : 0을 제외한 모든 값
  - 거짓 : 0
- C++에서는 참과 거짓을 명확하게 사용하기 위해, bool이라는 타입이 추가됨
  - 참 : true
  - 거짓 : false



# bool type: Example

## C

```
#include <stdio.h>

int main()
{
    int i = 1;

    if (i)
        printf("i is true!\n");
    else
        printf("i is false!\n");

    return 0;
}
```

## C++

```
#include <iostream>

int main()
{
    bool b = true;

    if (b)
        std::cout << "b is true!"
        << std::endl;
    else
        std::cout << "b is false!"
        << std::endl;

    return 0;
}
```

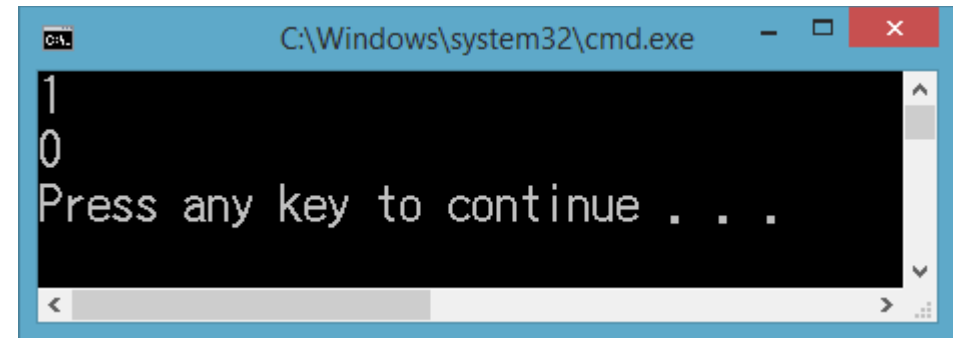
# bool type: Example

```
#include <iostream>

int main()
{
    bool b = true;

    std::cout << b << std::endl;
    std::cout << !b << std::endl;

    return 0;
}
```



# bool type: Example

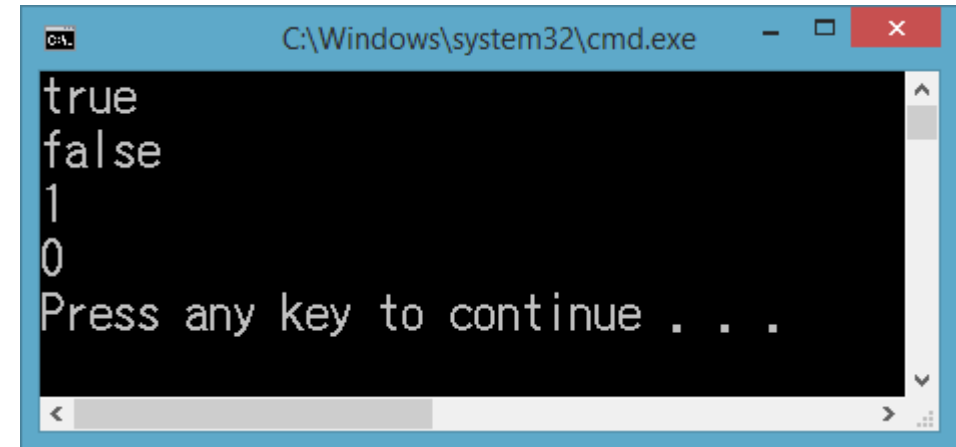
```
#include <iostream>

int main()
{
    bool b = true;

    std::cout.setf(std::ios::boolalpha);
    std::cout << b << std::endl;
    std::cout << !b << std::endl;

    std::cout.unsetf(std::ios::boolalpha);
    std::cout << b << std::endl;
    std::cout << !b << std::endl;

    return 0;
}
```



A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window has a black background with white text. The output of the program is displayed as follows:

```
true
false
1
0
Press any key to continue . . .
```

The output shows the boolean values 'true' and 'false' when the `boolalpha` flag is set, and the integer values '1' and '0' when it is unset. The prompt "Press any key to continue . . ." is shown at the bottom.

# auto keyword

Compile-Time Type Deduction in C++11

# auto keyword

- 표현식의 타입을 결정하는 건 때로는 어려운 일
- C++11에 추가된 auto 타입 지정자를 사용하면 컴파일 타임에 타입을 추론해 어떤 타입인지 결정함
- 기본 내장 타입을 포함해 컴파일 타임에 추론 가능한 모든 타입에서 사용 가능
- 만약 컴파일 타임에 추론이 불가능하다면, 오류가 발생함

# auto keyword: Example

## C++98

```
#include <iostream>

int main()
{
    int i1 = 10;
    double d1 = 3.14;

    return 0;
}
```

## C++11

```
#include <iostream>

int main()
{
    auto i2 = 10;
    auto d2 = 3.14;

    return 0;
}
```

# auto keyword: Example

C++98

```
#include <iostream>
#include <vector>
#include <string>
#include <tuple>

int main()
{
    std::vector<std::tuple<std::string, int, double>> vStudents;

    for (std::vector<std::tuple<std::string, int, double>>::iterator iter =
        vStudents.begin(); iter != vStudents.end(); ++iter) { ... }

    return 0;
}
```

# auto keyword: Example

## C++11

```
#include <iostream>
#include <vector>
#include <string>
#include <tuple>

int main()
{
    std::vector<std::tuple<std::string, int, double>> vStudents;

    for (auto iter = vStudents.begin(); iter != vStudents.end(); ++iter) { ... }

    return 0;
}
```