



20190919 실습 2주차

박건호

KITRI BoB 8th

RTDCS

devgunho.github.io



1. 연산자 익히기

- AND

$$7 \text{ AND } 8 = 0$$

$$0111(2) \text{ AND } 1000(2) = 0$$

- OR

$$7 \text{ OR } 8 = 15$$

$$0111(2) \text{ OR } 1000(2) = 1111(2)$$



1. 연산자 익히기 (2)

```
#include <iostream>

using namespace std;

int main()
{
    int input1 = 0, input2 = 0;
    cin >> input1 >> input2;

    cout << "두 값의 AND연산 결과 : " << (input1 & input2) << endl;
    cout << "두 값의 OR연산 결과 : " << (input1 | input2) << endl;
    cout << "두 값의 XOR연산 결과 : " << (input1 ^ input2) << endl;

    return 0;
}
```



2. 어느 비트가 켜져 있나요? (10)

- 입력은 0에서 15까지의 정수만 입력

3

0001	is	ON!
0010	is	ON!
0100	is	OFF
1000	is	OFF

11

0001	is	ON!
0010	is	ON!
0100	is	OFF
1000	is	ON!

15

0001	is	ON!
0010	is	ON!
0100	is	ON!
1000	is	ON!



3. 배열을 활용한 입력값들의 평균 구하기 (2)

- 1에서 20까지의 (중복)입력 (-1 입력까지)
- 배열 `int array[]` 활용, 입력값들의 평균 출력
- 예시 1)
 - 입력 : 3, 3, 1, -1
 - 출력 : 2.33 (소수 2번째 자리까지)
- 예시 2)
 - 입력 : 3, 4, 1, 5, 5, 20, 20, 20, -1
 - 출력 : 9.75 (소수 2번째 자리까지)

3. 배열을 활용한 입력값들의 평균 구하기 (2)

```
#include <iostream>

using namespace std;

int main()
{
    int array[21] = { 0, };
    double avg = 0;
    int input = 0;
    int count = 0;
    while (true)
    {
        cin >> input;
        if (input == -1) break;
        else if ((input < 1) && (input > 20))
        {
            cout << "wrong input!" << endl;
            break;
        }
        else
        {
            array[input]++;
            count++;
        }
    }

    for (int i = 0; i < 21; i++)
        avg += (array[i] * i);

    printf("%.2f\n", avg / count);
    return 0;
}
```



4. 배열을 활용한 정렬 (10)

- 1에서 20까지의 (중복)입력 (-1 입력까지)
- 배열 `int array[]` 을 활용하여 큰 수부터 출력
 - 입력 : 3, 4, 1, 5, 5, 20, 20, 20, -1
 - 출력 : 20, 20, 20, 5, 5, 4, 3, 1



4. 배열을 활용한 정렬 (10)

Microsoft Visual Studio Debug Console

```
3
4
1
5
5
20
20
20
-1
20 20 20 5 5 4 3 1
D:\#_Workspace#19_ComputerProgramming#CP_Lecture_
Press any key to close this window . . .
```




5. 2차원 배열을 활용한 구구단 출력 (2)

- 한줄에 9개씩 입력받은 $N(1 \leq N \leq 100)$ 개의 줄(N단)까지 출력하는 구구단을 출력하기

Microsoft Visual Studio Debug Console

```
3
1, 2, 3, 4, 5, 6, 7, 8, 9,
2, 4, 6, 8, 10, 12, 14, 16, 18,
3, 6, 9, 12, 15, 18, 21, 24, 27,
```

Microsoft Visual Studio Debug Console

```
9
1, 2, 3, 4, 5, 6, 7, 8, 9,
2, 4, 6, 8, 10, 12, 14, 16, 18,
3, 6, 9, 12, 15, 18, 21, 24, 27,
4, 8, 12, 16, 20, 24, 28, 32, 36,
5, 10, 15, 20, 25, 30, 35, 40, 45,
6, 12, 18, 24, 30, 36, 42, 48, 54,
7, 14, 21, 28, 35, 42, 49, 56, 63,
8, 16, 24, 32, 40, 48, 56, 64, 72,
9, 18, 27, 36, 45, 54, 63, 72, 81,
```

D:\Workspace\19_ComputerProgramming\CP_L...
Press any key to close this window . . .

5. 2차원 배열을 활용한 구구단 출력 (2)

```
#include <iostream>
using namespace std;

#define HEIGHT 100
#define WIDTH 9

int main() {
    int table[HEIGHT][WIDTH];
    int r, c;
    int input = 0;
    cin >> input;

    for (r = 0; r < HEIGHT; r++)
        for (c = 0; c < WIDTH; c++)
            table[r][c] = (r + 1) * (c + 1);

    for (r = 0; r < HEIGHT; r++) {
        if ( ) break;
        for (c = 0; c < WIDTH; c++) {
            cout << table[r][c] << ", ";
        }
        cout << endl;
    }
}
```



6. 2차원 배열을 활용한 출력 (10)

- 25 * 25 문자 배열을 공백으로 초기화하고
입력 값에 따라($1 \leq \text{input} \leq 25$) 'ㄷ' 모형으로
'*' 를 출력하는 함수를 구현한다.



6. 2차원 배열을 활용한 출력 (10)

3
★★
★
★★

5

★★★★

★

★

★

★★★★

 Microsoft

7

 *

 *

 *

 *

 *

10

★★★★★★★★

★

★

★

★

★

★

★

★

★★★★★★★★



7.1. Reference

- 자료형 변수명 = 값;
- 자료형 & **별명** = 기존 변수명;

여기서 &는 주소(address)를 의미하는 것이 아니라 **참조(reference)**를 의미한다.

7.1. Reference (2)

- reference type은 객체의 **별칭**으로 사용된다.
- ref와 value는 **동의어**로 취급된다.
- 참조형의 주소는 참조되는 주소의 값이다.

```
#include <iostream>

int main()
{
    int x = 5; // normal integer
    int& y = x; // y is a reference to x
    int& z = y; // z is also a reference to x

    //////////////////////////////////////

    int value = 5; // normal integer
    int& ref = value; // reference to variable value

    value = 6; // value is now 6
    ref = 7; // value is now 7

    std::cout << value << std::endl; // prints 7
    ++ref;
    std::cout << value << std::endl; // prints 8

    /*
    std::cout << &x << std::endl;
    std::cout << &y << std::endl;
    std::cout << &z << std::endl;

    std::cout << &value << std::endl;
    std::cout << &ref << std::endl;
    */

    return 0;
}
```



7.2. Reference As Function Parameters

참조형은 함수 매개 변수로 많이 사용되는데,
값을 복사하는데 비용을 줄여 성능을 향상시
킬 수 있다.



7.2. swap() by Reference (2)

```
#include <iostream>

using namespace std;

void swapVal(int a, int b)
{
    // cout << &a << " " << &b << endl;
    int temp = a;
    a = b;
    b = temp;
}

// 괄호 속의 & 는 "주소 연산자"가 아니라
// "참조 연산자"이고 C++에서만 가능
void swapRef(int& a, int& b) {
    // cout << &a << " " << &b << endl;
    int temp = a;
    a = b;
    b = temp;
}
```

```
int main(void) {
    int i = 300, j = 500;

    cout << "i = " << i << "j = " << j << endl;
    // cout << &i << " " << &j << endl;

    swapVal(i, j);
    cout << "i = " << i << "j = " << j << endl;

    swapRef(i, j); // 참조(Reference)에 의한 호출
    cout << "i = " << i << "j = " << j << endl;

    return 0;
}
```




7.3. Ranged-Based **for** Statement

- C++ 11에서는 범위 기반 for 문(ranged-based for statement)이라는 새로운 유형의 루프를 도입하여 더 간단하고 안전하게 배열 등의 모든 요소를 반복하는 방법을 제공한다.
- 문법 :
 for (element_declaration : array)
 statement;
- 설명 :
 Loop는 각 array의 요소를 반복하여 element_declaration에 선언된 변수에 현재 배열 요소의 값을 할당한다.
 최상의 결과를 얻으려면 element_declaration이 **배열 요소와 같은 자료형**이어야 한다.
 그렇지 않으면 형 변환이 발생한다.



7.3. Ranged-Based **for** Statement (2)

```
#include <iostream>
int main()
{
    // the initializer may be a braced-init-list
    for (int n : {0, 1, 2, 3, 4, 5})
        std::cout << n << ' ';
    std::cout << '\n';

    // Iterating over array
    int a[] = { 0, 1, 2, 3, 4, 5 };
    for (int n : a)
        std::cout << n << ' ';
    std::cout << '\n';

    // Just running a loop for every array
    // element
    for (int n : a)
        std::cout << "In loop" << ' ';
    std::cout << '\n';

    // Printing string characters
    std::string str = "Geeks";
    for (char c : str)
        std::cout << c << ' ';
    std::cout << '\n';
    return 0;
}
```

- 실행 과정 설명 :

먼저 for Loop가 실행되고 변수(n, c)에 배열(또는 문자열)의 첫 번째 **요소'값'**이 할당된다.

그 다음 출력하는 명령문이 실행되고 다시 Loop의 변수가 두 번째 **요소'값'**으로 할당된다.

for 루프는 반복할 배열에 원소가 남아있지 않을때까지 이와 같은 작업을 차례로 반복한다.

여기서 중요한 점은 배열에 대한 인덱스가 아니라 배열의 요소'값'이 변수에 할당된다는 점이다.



7.3. Ranged-Based **for** Statement (2)

```
#include <iostream>

int main()
{
    // the initializer may be a braced-init-list
    for (auto n : {0, 1, 2, 3, 4, 5})
        std::cout << n << ' ';
    std::cout << '\n';

    // Iterating over array
    int a[] = { 0, 1, 2, 3, 4, 5 };
    for (auto n : a)
        std::cout << n << ' ';
    std::cout << '\n';

    // Just running a loop for every array
    // element
    for (auto n : a)
        std::cout << "In loop" << ' ';
    std::cout << '\n';

    // Printing string characters
    std::string str = "Geeks";
    for (auto c : str)
        std::cout << c << ' ';
    std::cout << '\n';
    return 0;
}
```

for (element_declaration : array)
statement;

에서 배열 요소와 같은 자료형을 가져야 하므로, **auto** 키워드를 사용해서 C++이 자료형을 추론하도록 하는 것이 이상적이다.



7.3. Ranged-Based **for** Statement

- 형 변환 발생? **auto!**
- 값으로 할당? (=값이 복사?) **Reference(&)!**

```
// Iterating over array
int a[] = { 0, 1, 2, 3, 4, 5 };
for (auto& n : a)
    std::cout << n << ' ';
std::cout << '\n';
```



7.3. Ranged-Based **for** Statement

- 값이 변하지 않게 보호해주세요!

const! (상수 \leftrightarrow 변수)

```
// Iterating over array
int a[] = { 0, 1, 2, 3, 4, 5 };
for (const auto& n : a)
    std::cout << n << ' ';
std::cout << '\n';
```



문자열 입력

- "cin.getline()" 함수와 "std::getline()" 함수?
둘 다 "getline()"이라는 이름의 함수이고,
문자열을 입력 받는 데에 사용된다.
 - cin.getline() 함수는 문자 배열이며 마지막 글자가 '\0'(terminator)인 c-string을 입력 받는 데에 사용한다.
 - 이에 반해, std::getline() 함수는 원하는 구분자(delimiter)를 만날 때 까지 모든 문자들을 입력 받아서 하나의 string 객체에 저장한다.



8. "computer" 문자열 검색 후 삭제 (2)

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string s;
    getline(cin, s);
    int index = s.find("computer");
    for (int i = 0; i < s.length(); i++)    // 문자열 s의 길이까지 Loop
    {
        if (i == index)
            i += 8;    // computer가 8글자이기 때문에...
        else
            cout << s[i];
    }
    return 0;
}
```

Microsoft Visual Studio Debug Console

```
aaacomputeraaa
aaaaaa
D:\_Workspace\19_ComputerProgramming\CP_Lecture_Code\Debug\CP_Lecture_Code.exe (process 222)
Press any key to close this window . . .
```

Microsoft Visual Studio Debug Console

```
A computer is a machine that can be instructed to carry out sequences of arithmetic ...
A  is a machine that can be instructed to carry out sequences of arithmetic ...
D:\_Workspace\19_ComputerProgramming\CP_Lecture_Code\Debug\CP_Lecture_Code.exe (process 222)
Press any key to close this window . . .
```



9 문자열의 숫자 변환 - stoi (2)

yyyy-mm-dd 형태로 2줄을 입력 받고,
yyyy년 mm월 dd일 형태로 바꾼뒤,
받은 날의 차이를 출력한다.

입력은 2019-09-dd 형태로 고정.



9. 문자열의 숫자 변환 – stoi (2)

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    string str="",str2="";
    int year = 0, month = 0, day = 0;
    int year2 = 0, month2 = 0, day2 = 0;

    //cout << "yyyy-mm-dd : ";
    getline(cin, str, '\n');
    //cout << "yyyy-mm-dd : ";
    getline(cin, str2, '\n');

    year = stoi(str.substr(0, 4));
    month = stoi( );
    day = stoi( );
}
```

```
year2 = stoi( );
month2 = stoi( );
day2 = stoi( );
```

```
cout << year << "년 " << month << "월 " << day << "일" << endl;
cout << year2 << "년 " << month2 << "월 " << day2 << "일" << endl;
cout << "차이는 : " << (day2 - day) << "일" << endl;
return 0;
```

```
2019-09-11
2019-09-22
2019년 9월 11일
2019년 9월 22일
차이는 : 11일
```

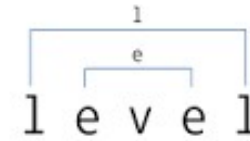


10. Pallendrome String (10)

회문



회문 또는 팰린드롬은 거꾸로 읽어도 제대로 읽는 것과 같은 문장이나 낱말이다. 보통 낱말 사이에 있는 띄어쓰기나 문장 부호는 무시한다. 위키백과



Microsoft Visual Studio

```
aaabbb
NO.
```

```
D:\Workspace
Press any key
```

Microsoft Visual Studio

```
level
YES!
```

```
D:\Worksp
Press any
```

Microsoft Visual Studio

```
aaaaaaaa
YES!
```

```
D:\Workspace#1:
Press any key to
```