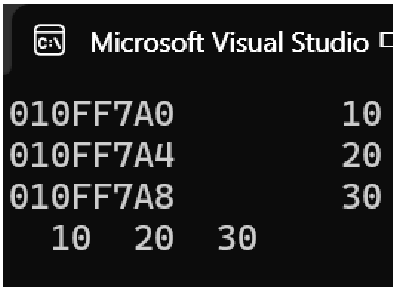
객체지향프로그래밍 LAB #09

<기초문제>

1.

```
#include <iostream>
#include <vector>
#include <iomanip>
using namespace std;
using Matrix = vector<vector<int>>;

// 배열의 경우 주소값(시작주소, 끝주소)을 전달
void print(int* begin, int* end) {
        for (int* curr = begin; curr < end; curr++)
                cout << setw(4) << *curr;

        // while문 구현 부분 - 수업시간에 배운 주소값을 기준으로 while문 작성
        // int* curr = begin;
        // while (curr < end) {
        //   cout << setw(4) << *curr;
        //   curr++;
        // }
        cout << endl;
}

// (+, -) for pointer: 주소값을 증가/감소 (다음 변수 위치)
int main() {
        int list[3] = { 10, 20, 30 };
        cout << list << '₩t' << list[0] << endl;
        cout << list + 1 << '₩t' << list[1] << endl;
        cout << list + 2 << '₩t' << list[2] << endl;

        int* begin = list;
        int* end = list + 3;
        print(begin, end);

        return 0;
}
```

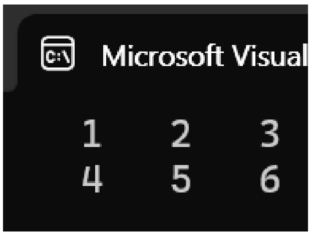


```
010FF7A0        10
010FF7A4        20
010FF7A8        30
  10  20  30
```

2.

```
#include <iostream>
#include <vector>
#include <iomanip>
using namespace std;
using Matrix = vector<vector<int>>;

void print(const Matrix& mat) {
        // vector index를 이용한 for 문 작성
        /*
        for (unsigned row = 0; row < mat.size(); row++) {
            for (unsigned col = 0; col < mat[row].size(); col++) {
            //mat.at(row).at(col);
                cout << setw(4) << mat[row][col];
            }
            cout << endl;
        }
        */

        // vector 원소를 이용한 for 문 작성
        /*
        for (vector<int> row : mat) {
            for (int col : row) {
                cout << setw(4) << col;
            }
            cout << endl;
        }
        */

        //유추 가능한 경우, 자료형 부분을 auto로 치환가능
        // vector<int> row = mat[0];
```

```
    //   ==      auto row = mat[0];
    // auto와 벡터 원소를 이용하여 for문 구현
    for (auto row : mat) {
        for (int col : row) {
            cout << setw(4) << col;
        }
        cout << endl;
    }
}
int main() {
    // 2 x 3 matrix
    // vector<vector<int>> mat(2, vector < int>(3) );
    Matrix mat{ { 1, 2, 3 }, { 4, 5, 6 } };
    mat[0][0] = 1;
    mat[0][1] = 2;
    mat[0][2] = 3;
    mat[1][0] = 4;
    mat[1][1] = 5;
    mat[1][2] = 6;

    print(mat);

    return 0;
}
```

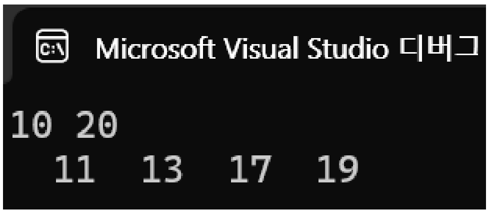


```
   1   2   3
   4   5   6
```

3.

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

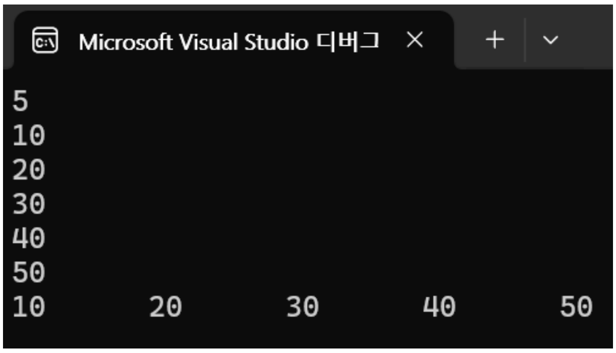
```
//소수 : 1과 자기자신을 제외하고는 약수가 없는 1보다 큰 정수
bool is_prime(int n) {
        if (n < 2)
                return false;
        for (int i = 2; i < n; i++)
                if (n % i == 0)
                        return false; // n을 i로 나눈 나머지가 0이면 false를 리턴
        return true;
}

vector<int> primes(int low, int high) {
        vector<int> result;
        for (int i = low; i <= high; i++)
                if (is_prime(i))
                        result.push_back(i); // 소수이면 (is_prime이 참이면) 뒤에 push
        return result;
}

void print(const vector<int>& v) {
        for (int elem : v) //vector index가 아닌 원소를 이용한 for문
                cout << setw(4) << elem;
        cout << endl;
}

int main() {
        int low, high;
        cin >> low >> high;
        vector<int> vec = primes(low, high);
        print(vec);

        return 0;
}
```



```
10 20
  11  13  17  19
```

4.

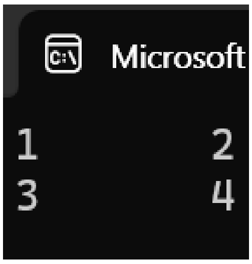
```
#include <iostream>
#include <vector>
using namespace std;

//정적 배열(static array):프로그램 실행중 크기가 고정되어 변경 불가
//동적 배열(dynamic array): 프로그램 실행중(run time) 할당/해제가 가능
int main() {
	const int size = 3;
	int list[size] = { 10, 20, 30 };
	int length = 3;
	cin >> length; // 키보드로부터 배열의 크기를 입력받음
	int* list2 = new int[length]; //동적 배열 선언
	// double* list2 = new double[length]

	int* begin = list2;
	int* end = list2 + length;

	for (int* curr = begin; curr != end; curr++)
		cin >> *curr;

	for (int* curr = begin; curr != end; curr++)
		cout << *curr << '₩t';
	cout << endl;

	delete [] list2; //할당 해제

	return 0;
}
```



```
5
10
20
30
40
50
10      20      30      40      50
```

5.

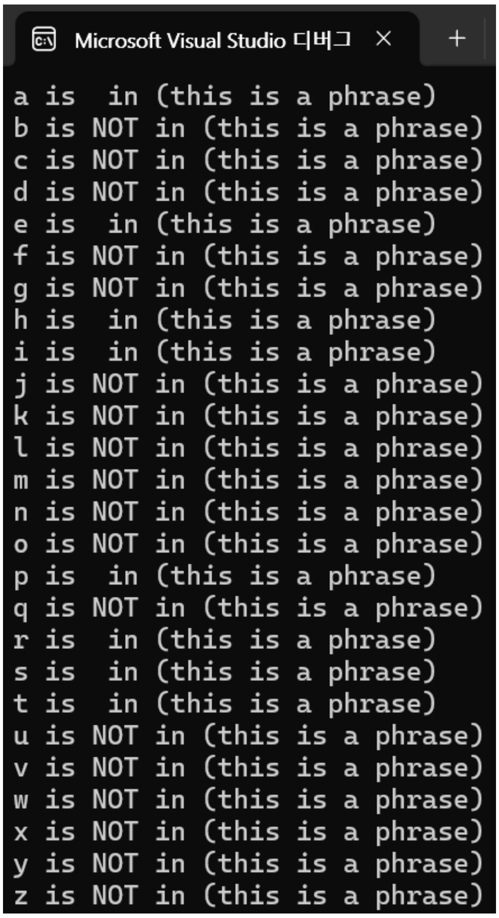
```
#include <iostream>
#include <vector>
using namespace std;

void print(int** m, int nRow, int nCol) {
    for (int i = 0; i < nRow; i++) {
        for (int j = 0; j < nCol; j++)
            cout << m[i][j] << '₩t';
        cout << endl;
    }//2중 for문과 index를 이용하여 배열 원소 출력 - 구분자 : '₩t'
}

int main() {
    int nRow = 2, nCol = 2;
    int** matrix2;
    matrix2 = new int*[nRow]; // 동적배열 선언(행기준)
    for (int i = 0; i < nRow; i++)
        matrix2[i] = new int[nCol]; // 동적배열 선언(열기준)

    matrix2[0][0] = 1; matrix2[0][1] = 2;
    matrix2[1][0] = 3; matrix2[1][1] = 4;

    print(matrix2, nRow, nCol);

    for (int i = 0; i < nRow; i++)
        delete[] matrix2[i];

    delete[] matrix2;

    return 0;
}
```



```
1       2
3       4
```

6.

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

bool found_char(const char* s, char ch) {
    int i = 0;
    while (true) {
        if (s[i] == ch)
            return true;
        else if (s[i] == '₩0')
            return false;
        i++;
    }// s와 ch 만으로 (s,s+1, ...)에 ch가 있는지 true/false return
}
int main() {
    //                    012345(6)
    const char* phrase = "this is a phrase";// ch[]
    // phrase(₩0)==NULL

    for (char ch = 'a'; ch <= 'z'; ch++) { // 'a' == 65, 'z' == 97
        cout << ch << " is ";
        if (!found_char(phrase, ch))
            cout << "NOT";
        cout << " in (" << phrase << ")" << endl;
    }

    return 0;
}
```



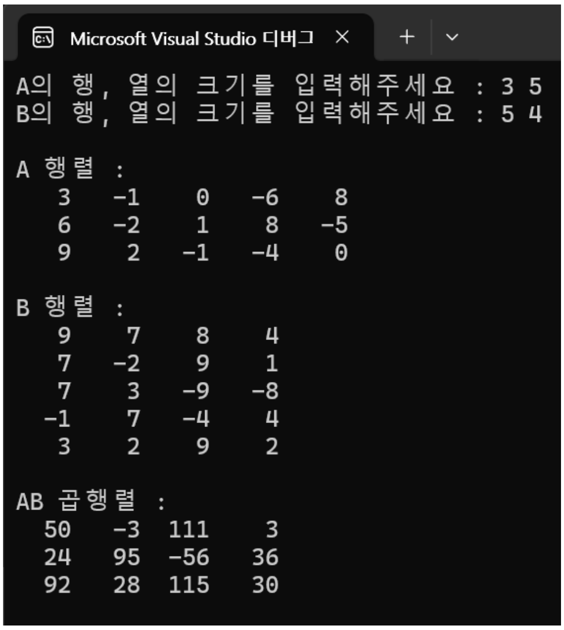
```
a is  in (this is a phrase)
b is NOT in (this is a phrase)
c is NOT in (this is a phrase)
d is NOT in (this is a phrase)
e is  in (this is a phrase)
f is NOT in (this is a phrase)
g is NOT in (this is a phrase)
h is  in (this is a phrase)
i is  in (this is a phrase)
j is NOT in (this is a phrase)
k is NOT in (this is a phrase)
l is NOT in (this is a phrase)
m is NOT in (this is a phrase)
n is NOT in (this is a phrase)
o is NOT in (this is a phrase)
p is  in (this is a phrase)
q is NOT in (this is a phrase)
r is  in (this is a phrase)
s is  in (this is a phrase)
t is  in (this is a phrase)
u is NOT in (this is a phrase)
v is NOT in (this is a phrase)
w is NOT in (this is a phrase)
x is NOT in (this is a phrase)
y is NOT in (this is a phrase)
z is NOT in (this is a phrase)
```

<응용문제>

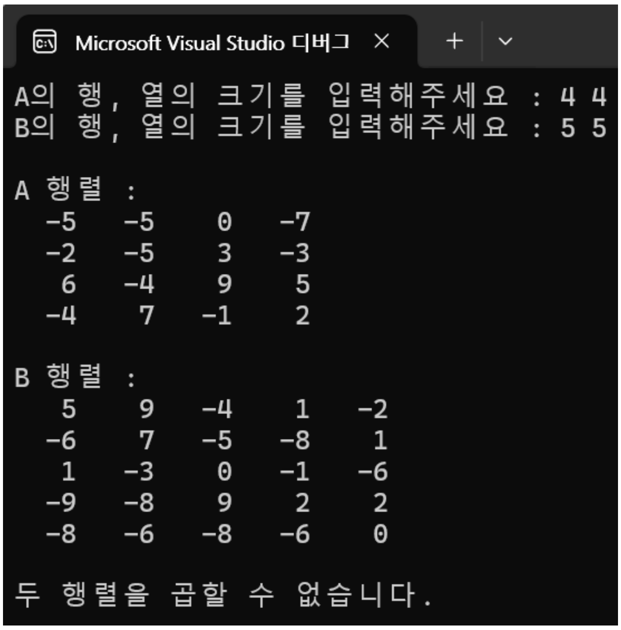
1.

```
#include <iostream>
#include <random>
#include <vector>
#include <iomanip>
using namespace std;
void matrix(vector<vector<int>>& s, int a, int b) {
    random_device r;
    mt19937 gen(r());
    uniform_int_distribution<> dist(-9, 9);
    for (int i = 0; i < a; i++) {
        for (int j = 0; j < b; j++)
            s[i][j] = dist(gen);
    }
}
```

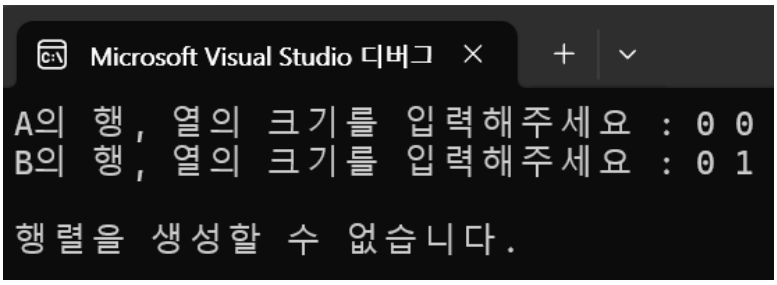
```
void print(vector<vector<int>>& s) {
    for (auto r : s) {
        for (int elem : r) {
            cout << setw(4) << elem << ' ';
        }
        cout << endl;
    }
    cout << endl;
}
void mulmatrix(vector<vector<int>>& AB, vector<vector<int>> A, vector<vector<int>> B) {
    int s = 0;
    for (unsigned int k = 0; k < A.size(); k++) {
        for (unsigned int i = 0; i < B[0].size(); i++) {
            for (unsigned int j = 0; j < A[0].size(); j++) {
                s += A[k][j] * B[j][i];
            }
            AB[k][i] = s;
            s = 0;
        }
    }
}
int main() {
    int ar, ac, br, bc;
    cout << "A의 행, 열의 크기를 입력해주세요 : ";
    cin >> ar >> ac;
    cout << "B의 행, 열의 크기를 입력해주세요 : ";
    cin >> br >> bc;
    cout << endl;
    vector<vector<int>> A(ar, vector<int>(ac)), B(br, vector<int>(bc)), AB(ar, vector<int>(bc));
    if (ar * ac * br * bc == 0) {
        cout << "행렬을 생성할 수 없습니다." << endl;
        return 0;
    }
    matrix(A, ar, ac);
    matrix(B, br, bc);
    matrix(AB, ar, bc);
    cout << "A 행렬 : " << endl;
    print(A);
    cout << "B 행렬 : " << endl;
    print(B);
    if (ac != br) {
        cout << "두 행렬을 곱할 수 없습니다." << endl;
        return 0;
    }
    mulmatrix(AB, A, B);
    cout << "AB 곱행렬 : " << endl;
    print(AB);
}
```



```
A의 행, 열의 크기를 입력해주세요 : 3 5
B의 행, 열의 크기를 입력해주세요 : 5 4

A 행렬 :
   3   -1    0   -6    8
   6   -2    1    8   -5
   9    2   -1   -4    0

B 행렬 :
   9    7    8    4
   7   -2    9    1
   7    3   -9   -8
  -1    7   -4    4
   3    2    9    2

AB 곱행렬 :
  50   -3  111    3
  24   95  -56   36
  92   28  115   30
```



```
A의 행, 열의 크기를 입력해주세요 : 4 4
B의 행, 열의 크기를 입력해주세요 : 5 5

A 행렬 :
  -5   -5    0   -7
  -2   -5    3   -3
   6   -4    9    5
  -4    7   -1    2

B 행렬 :
   5    9   -4    1   -2
  -6    7   -5   -8    1
   1   -3    0   -1   -6
  -9   -8    9    2    2
  -8   -6   -8   -6    0

두 행렬을 곱할 수 없습니다.
```

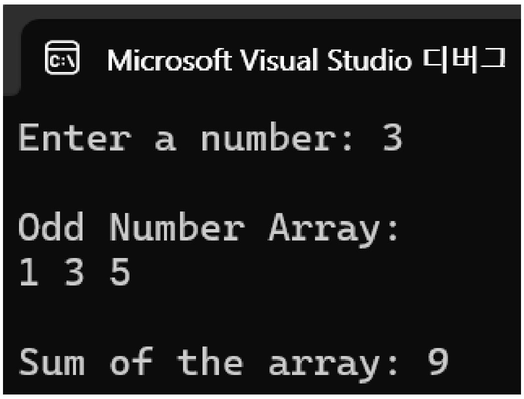


```
A의 행, 열의 크기를 입력해주세요 : 0 0
B의 행, 열의 크기를 입력해주세요 : 0 1

행렬을 생성할 수 없습니다.
```

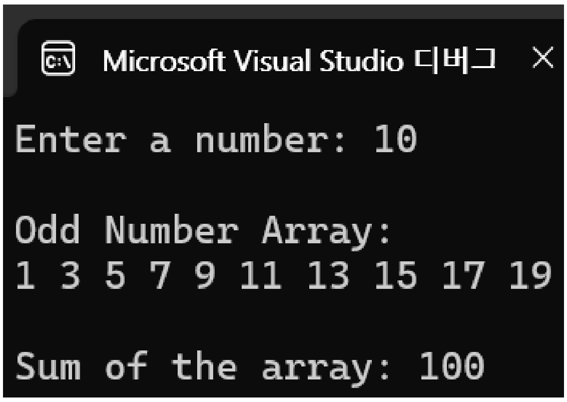
2.

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

int* make_arr(int n) {
        int* a = new int[n];
        for (int k = 0; k < n; k++) {
```

```
        a[k] = 2 * (k + 1) - 1;
    }
    return a;
}
void print_arr(int* a, int n) {
    cout << "₩nOdd Number Array:" << endl;
    for (int i = 0; i < n; i++) {
        cout << a[i] << " ";
    }
    cout << endl;
}
int sum_arr(int* a, int n) {
    int s = 0;
    for (int i = 0; i < n; i++) {
        s += *(a+i);
    }
    return s;
}

int main() {
    int n;
    cout << "Enter a number: ";
    cin >> n;

    int* arr = make_arr(n);
    print_arr(arr, n);

    int sum = sum_arr(arr, n);
    cout << "₩nSum of the array: " << sum << endl;

    delete[] arr;
    return 0;
}
```



```
Enter a number: 3

Odd Number Array:
1 3 5

Sum of the array: 9
```

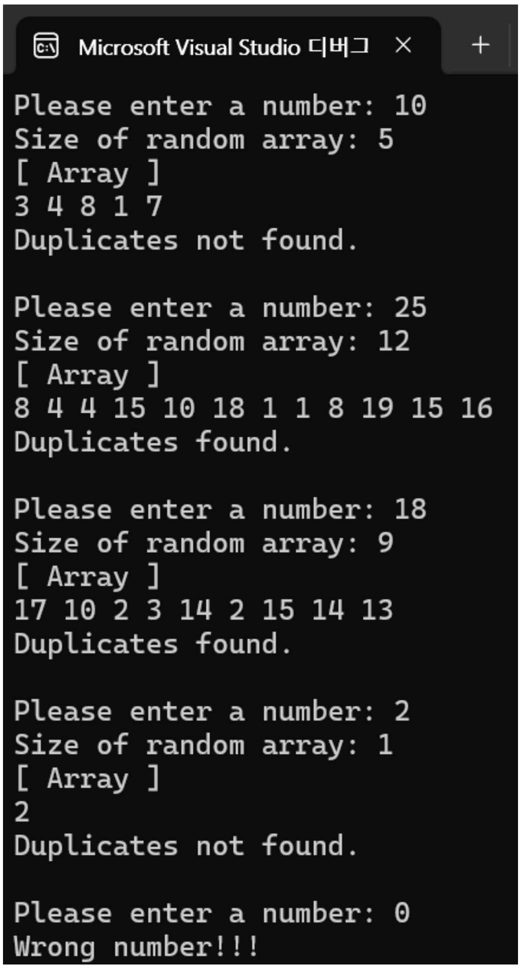


```
Enter a number: 10

Odd Number Array:
1 3 5 7 9 11 13 15 17 19

Sum of the array: 100
```

3.

```
#include <iostream>
#include <random>
#include <string>
#include <vector>
#include <set>
using namespace std;

int main() {
        while (true) {
                int n;
                cout << "Please enter a number: ";
                cin >> n;
                if (n < 2) {
                    cout << "Wrong number!!!";
                    break;
                }
                cout << "Size of random array: " << (n / 2) << endl;
                cout << "[ Array ]" << endl;
                random_device r;
                mt19937 gen(r());
                uniform_int_distribution<> dist(1, n);
                int* a = new int[(n / 2)];
                for (int k = 0; k < (n / 2); k++) {
                    a[k] = dist(gen);
                }
                set<int> p(a, a + (n / 2));
                for (int k = 0; k < (n / 2); k++) {
```

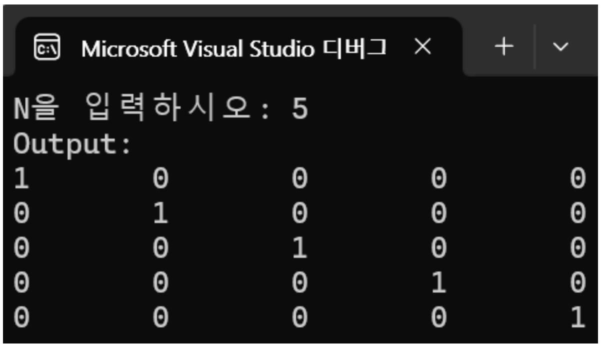
```
            cout << a[k] << ' ';
        }
        cout << endl;
        if (p.size() == (n / 2)) {
            cout << "Duplicates not found." << endl << endl;;
            delete[] a;
            continue;
        }
        else {
            cout << "Duplicates found." << endl << endl;;
            delete[] a;
            continue;
        }
    }
}
```



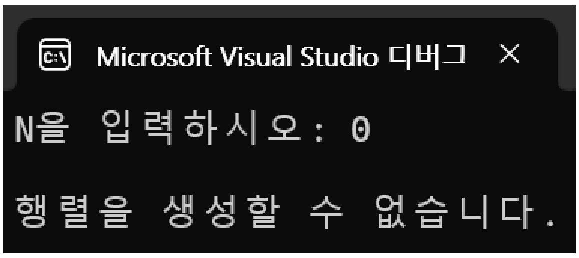
```
Please enter a number: 10
Size of random array: 5
[ Array ]
3 4 8 1 7
Duplicates not found.

Please enter a number: 25
Size of random array: 12
[ Array ]
8 4 4 15 10 18 1 1 8 19 15 16
Duplicates found.

Please enter a number: 18
Size of random array: 9
[ Array ]
17 10 2 3 14 2 15 14 13
Duplicates found.

Please enter a number: 2
Size of random array: 1
[ Array ]
2
Duplicates not found.

Please enter a number: 0
Wrong number!!!
```

4.

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

int** buildTable(int n) {
        int** a = new int* [n];
        for (int i = 0; i < n; i++) {
                a[i] = new int[n];
                for (int j = 0; j < n; j++)
                        a[i][j] = 0;
        }
        return a;
}
void make_identity_matrix(int** table, int n) {
        for (int i = 0; i < n; i++) {
                table[i][i] = 1;
        }
}

void printTable(int** table, int n) {
        for (int i = 0; i < n; i++) {
                for (int j = 0; j < n-1; j++)
                        cout << table[i][j] << setw(8);
                cout << table[i][n-1] << endl;
        }
}

int main() {
        int n = 0;
        cout << "N을 입력하시오: ";
        cin >> n;
        if (n < 1) {
                cout << "₩n행렬을 생성할 수 없습니다.₩n" << endl;
                exit(EXIT_FAILURE);
        }
        cout << "Output: " << endl;
        int** table = buildTable(n);
        make_identity_matrix(table, n);
        printTable(table, n);
        for (int i = 0; i < n; i++)
                delete[] table[i];
        delete[] table;
        return 0;
}
```



```
N을 입력하시오: 5
Output:
1       0       0       0       0
0       1       0       0       0
0       0       1       0       0
0       0       0       1       0
0       0       0       0       1
```

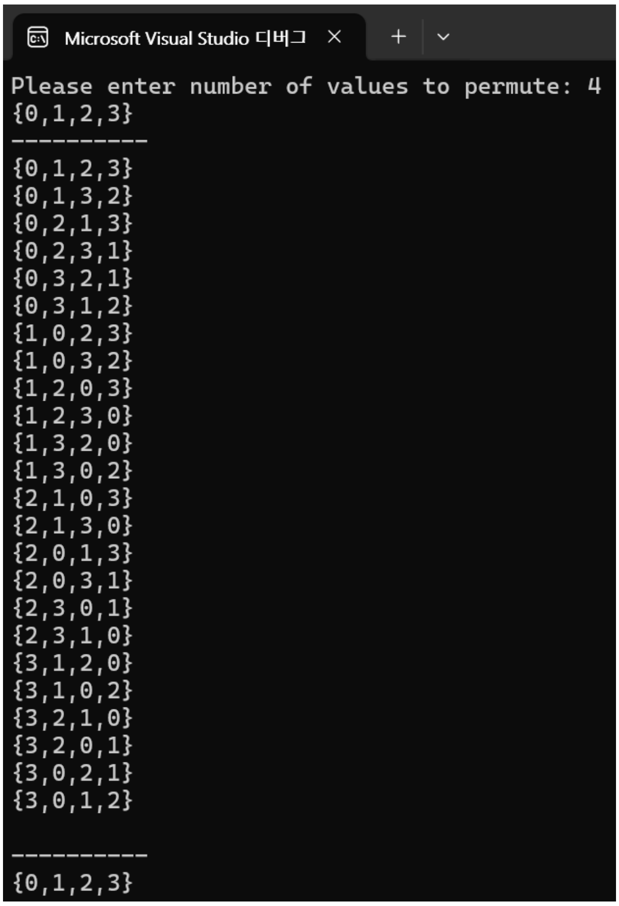


```
N을 입력하시오: 0
행렬을 생성할 수 없습니다.
```

5.

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

void print(const std::vector<int>& a) {
    int n = a.size();
    std::cout << "{";
    if (n > 0) {
        std::cout << a[0];
        for (int i = 1; i < n; i++)
            std::cout << ',' << a[i];
    }
    std::cout << "}";
}

void swap(int& a, int& b) {
    int temp = a;
    a = b;
    b = temp;
}
```

```
void permute(vector<int>& a, int begin, int end) {
    if (begin == end) {
        print(a);
        cout << endl;
    }
    else {
        for (int i = begin; i <= end; i++) {
            swap(a[begin], a[i]);
            permute(a, begin + 1, end);
            swap(a[begin], a[i]);
        }
    }
}

int main() {
    // Get number of values from the user
    std::cout << "Please enter number of values to permute: ";
    int number;
    std::cin >> number;
    // Create the vector to hold all the values
    std::vector<int> list(number);
    // Initialize the vector
    for (int i = 0; i < number; i++)
        list[i] = i;
    // Print original list
    print(list);
    std::cout << "₩n----------₩n";
    // Print all the permutations of list
    permute(list, 0, number - 1);
    std::cout << "₩n----------₩n";
    // Print list after all the manipulations
    print(list);
}
```



```
Please enter number of values to permute: 4
{0,1,2,3}
---------
{0,1,2,3}
{0,1,3,2}
{0,2,1,3}
{0,2,3,1}
{0,3,2,1}
{0,3,1,2}
{1,0,2,3}
{1,0,3,2}
{1,2,0,3}
{1,2,3,0}
{1,3,2,0}
{1,3,0,2}
{2,1,0,3}
{2,1,3,0}
{2,0,1,3}
{2,0,3,1}
{2,3,0,1}
{2,3,1,0}
{3,1,2,0}
{3,1,0,2}
{3,2,1,0}
{3,2,0,1}
{3,0,2,1}
{3,0,1,2}

---------
{0,1,2,3}
```