

객체지향프로그래밍 LAB #09

<기초문제>

1. 아래의 프로그램을 작성하시오. (/*구현*/ 부분을 채울 것, 표의 상단: 소스코드, 하단: 실행결과)

```
#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 (/*구현*/) //수업시간에 배운 주소값을 기준으로 for문 작성
        cout << setw(4) << *curr;

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

// (+, -) for pointer: 주소값을 증가/감소 (다음 변수 위치)
int main() {
    int list[3] = { 10, 20, 30 };
    cout << /*구현*/ << 'Wt' << /*구현*/ << endl;
    cout << /*구현*/ << 'Wt' << /*구현*/ << endl;
    cout << /*구현*/ << 'Wt' << /*구현*/ << endl;

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

    return 0;
}
```

 Microsoft Visual Studio 디버그 콘솔

```
00CFF814      10
00CFF818      20
00CFF81C      30
10 20 30
```

2. 아래의 프로그램을 작성하시오. (/*구현*/ 부분을 채울 것)

```
#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 (/*구현*/) {
    //         for (/*구현*/) {
    //             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 (/*구현*/) {
        for (/*구현*/) {
            cout << setw(4) << /*구현*/;
        }
        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;
}

```

Microsoft Visual Studio 디버그 콘솔

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

3. 아래의 프로그램을 작성하시오. (*/*구현*/* 부분을 채울 것)

```
#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++)
        /*구현*/ // 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++)
        /*구현*/ // 소수이면 (is_prime이 참이면) 뒤에 push
    return result;
}

void print(const vector<int>& v) {
    for (/*구현*/ //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;
}
```

Microsoft Visual Studio 디버그 콘솔

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

4. 아래의 프로그램을 작성하시오. (/*구현*/ 부분을 채울 것)

```
#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 = /*구현*/ //동적 배열 선언
    // double* list2 = new double[length]

    int* begin = /*구현*/
    int* end = /*구현*/

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

    for (int* curr = begin; curr != end; curr++)
        cout << *curr << 'Wt';
    cout << endl;

    /*구현*/ //할당 해제

    return 0;
}
```

Microsoft Visual Studio 디버그 콘솔

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

5. 아래의 프로그램을 작성하시오. (/*구현*/ 부분을 채울 것)

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

void print(int** m, int nRow, int nCol) {
    /*구현*/ //2중 for문과 index를 이용하여 배열 원소 출력 - 구분자 : 'Wt'
}

int main() {
    int nRow = 2, nCol = 2;
```

```

int** matrix2;
matrix2 = /*구현*/ // 동적배열 선언(행기준)
for (int i = 0; i < nRow; i++)
    matrix2[i] = /*구현*/ // 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;
}

```

 Microsoft Visual Studio 디버그 콘솔

```

1      2
3      4

```

6. 아래의 프로그램을 작성하시오. (/*구현*/ 부분을 채울 것)

```

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

bool found_char(const char* s, char ch) {
    /* 구현 */ // s와 ch 만으로 (s,s+1, ...)에 ch가 있는지 true/false return
}

int main() {
    //          012345(6)
    const char* phrase = "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;
}

```

Microsoft Visual Studio 디버그 콘솔

```
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)
```

<응용문제>

1. 행렬 두 개의 크기를 입력하여 생성된 두 행렬의 곱을 출력하는 프로그램을 작성하시오.

- 행렬은 2-D Vector를 이용하여 선언함.
- 행렬의 요소는 -9 이상 9 이하의 정수 중 하나를 랜덤으로 설정함.
- 행렬을 초기화하는 함수, 행렬을 출력하는 함수, 행렬을 곱하는 함수를 구현함.
- 행렬을 생성할 수 없으면 오류메시지를 출력하고 종료함.
- 두 행렬을 곱할 수 없으면 오류메시지를 출력하고 종료함.

1-출력화면:

```
Microsoft Visual Studio 디버그 콘솔
A의 행, 열의 크기를 입력해주세요 : 3 5
B의 행, 열의 크기를 입력해주세요 : 5 4

A 행렬 :
-6  9 -2  5  8
 2 -7 -6 -8  2
-4 -3 -3  3 -4

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

AB 곱행렬 :
-129  62 -54  18
 175 -74  5 -86
 -7 -2 -30  34
```

```
Microsoft Visual Studio 디버그 콘솔
A의 행, 열의 크기를 입력해주세요 : 4 4
B의 행, 열의 크기를 입력해주세요 : 5 5

A 행렬 :
-6  9 -2  5
 8  2 -7 -6
-8  2 -4 -3
-3  3 -4  7

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

두 행렬을 곱할 수 없습니다.
```

```
Microsoft Visual Studio 디버그 콘솔
A의 행, 열의 크기를 입력해주세요 : 0 0
B의 행, 열의 크기를 입력해주세요 : 0 1

행렬을 생성할 수 없습니다.
```

2. 다음은 자연수 n 을 입력 받아, 길이가 n 인 홀수 배열을 만들어 배열과 배열의 합을 출력하는 프로그램이다. 다음 조건에 맞게 함수를 구현 및 수정하시오.

- 함수 `make_arr`에서는 `new`를 이용해 입력 받은 숫자의 크기만큼 배열을 동적으로 할당함.
- 함수 `print_arr`는 포인터 표기법 대신 배열 표기법으로 수정. (`while`을 `for`로 수정가능)
- 함수 `sum_arr`는 배열 표기법을 포인터 표기법으로 수정. (`for`를 `while`로 수정가능)

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

int* make_arr(int n) { /* 구현 */ }
void print_arr(int* a, int n) {
    cout << "Odd Number Array:" << endl;
    while (n) {
        cout << *a << " ";
        a++;
        n--;
    }
    cout << endl;
}

int sum_arr(int* a, int n) {
    int s = 0;
    for (int i = 0; i < n; i++)
        s += a[i];
    return s;
}

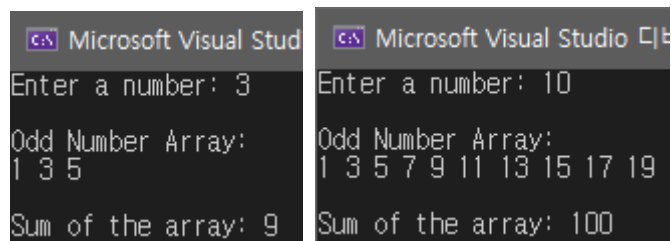
int main() {
    int n;
    cout << "Enter a number: ";
    cin >> n;

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

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

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

2-출력화면:



3. 자연수 n 을 입력 받고, 길이가 $n/2$ 인 난수 배열을 만들어 배열의 성분들이 중복이 있는지 확인하는 프로그램을 작성하시오. 단, 다음 조건을 모두 만족해야 함.

- new를 이용해 입력 받은 숫자의 크기만큼 배열을 동적으로 할당함.
- 배열 내 생성된 난수의 범위는 1이상 n 이하로 설정함.
- 생성된 배열의 크기와 요소 및 중복 유무를 출력함.
- 프로그램은 반복되며, 2보다 작은 숫자를 입력할 경우에 프로그램을 종료함.

3-출력화면:

```
Microsoft Visual Studio 디버그 콘솔
Please enter a number: 10
Size of random array: 5
[ Array ]
2 8 5 1 10
Duplicates not found.

Please enter a number: 25
Size of random array: 12
[ Array ]
25 4 9 13 15 6 21 7 3 12 17 21
Duplicates found.

Please enter a number: 18
Size of random array: 9
[ Array ]
9 4 1 10 7 15 10 5 17
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. 자연수 n 을 입력 받아 $n \times n$ 크기의 2차원 단위행렬을 생성하고 출력하는 프로그램을 작성하시오. 단, 다음 조건을 모두 만족해야 함.

- new를 이용해 2차원 배열을 동적으로 할당함. (hint. 구글에 “2차원배열 동적할당” 검색)
- 함수 buildTable은 배열 생성 및 모든 원소를 0으로 초기화.
- 함수 make_identity_matrix는 대각원소에 1을 대입.
- 함수 printTable은 생성된 대각행렬 출력.

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

int** buildTable(int n) { /* 구현 */ }
void make_identity_matrix(int** table, int n) { /* 구현 */ }
void printTable(int** table, int n) { /* 구현 */ }

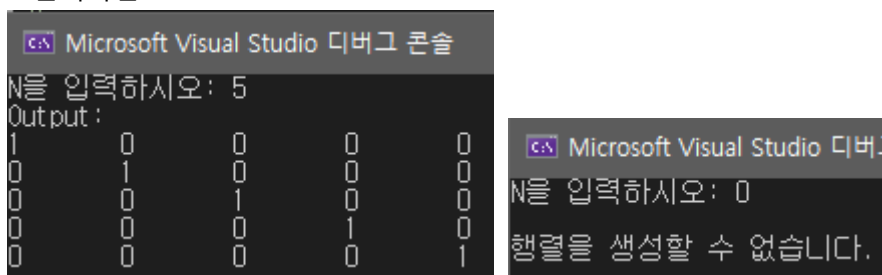
int main() {
    int n = 0;
    cout << "N을 입력하시오: ";
    cin >> n;
    if (n < 1) {
        cout << "n행렬을 생성할 수 없습니다.n" << endl;
        exit(EXIT_FAILURE);
    }

    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;
}
```

4-출력화면:



5. (recursion 연습) 임의의 크기를 가지는 벡터를 키보드에서 입력받은 뒤 그 벡터의 원소들에 대한 모든 permutation 조합을 출력하는 프로그램을 작성하라. 시작코드는 아래와 같으며, permute() 함수를 작성하면 됨. 반드시 recursion을 사용할것.

====시작코드=====

```
#include <iostream>
#include <vector>

/*
 * print
 * Prints the contents of a vector of integers
 * a is the vector to print; a is not modified
 */
void print(const std::vector<int>& a) {
    int n = a.size();
    std::cout << "{";
    if (n > 0) {
        std::cout << a[0];    // Print the first element
        for (int i = 1; i < n; i++)
            std::cout << ',' << a[i];    // Print the rest
        }
    std::cout << "}";
}

/*
 * Prints all the permutations of vector a in the
 * index range begin...end, inclusive. The function's
 * behavior is undefined if begin or end
 * represents an index outside of the bounds of vector a.
 */
void permute(std::vector<int>& a, int begin, int end) {
}

/*
 * Tests the permutation functions
 */
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);
}
```

```
}
```

====시작코드 끝====

실행결과

```
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}
```