

객체지향프로그래밍 LAB #09

<기초문제>

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

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

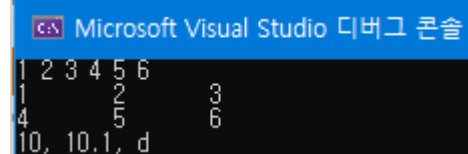
int main() {
    /*구현*/ //2 x 3 2차원 vector 선언

    for (/*구현*/) { // matrix 각각에 대해 cin을 받을 수 있도록 for문 채움
        for (int& elem : v) {
            cin >> elem;
        }
    }

    for (/*구현*/) { // matrix를 auto로 받도록 for문 채움
        for (auto elem : v) {
            cout << elem << "Wt";
        }
        cout << endl;
    }

    auto a = 10;
    auto b = 10.1;
    auto c = 'd';
    cout << a << ", " << b << ", " << c << endl;

    return 0;
}
```



```
Microsoft Visual Studio 디버그 콘솔
1 2 3 4 5 6
1 2 3
4 5 6
10, 10.1, d
```

2. 아래의 프로그램을 작성하시오. (*구현* 부분을 채울 것)

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

int main() {
    int ary[3] = { 1, 2, 3 };

    // 수업시간에 배운 내용을 토대로 배열 내 성분들의 주소값과 값을 출력
    cout << /*구현*/ << ", " << /*구현*/ << ", " << ary[0] << endl;
    cout << /*구현*/ << ", " << /*구현*/ << ", " << ary[1] << endl;
    cout << /*구현*/ << ", " << /*구현*/ << ", " << ary[2] << endl;
    return 0;
}
```

```
Microsoft Visual Studio 디버그 콘솔
006FFC54, 1, 1
006FFC58, 2, 2
006FFC5C, 3, 3
```

3. 아래의 프로그램을 작성하시오. (*구현* 부분을 채울 것)

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

void print(const int* ar, const int length) {
    /*구현*/ //print(array, 길이) 입력 시 array 내용물 출력하는 함수
}

int sum(int* begin, int* end) {
    int* curr = begin;
    int result = 0;
    while (curr != end) {
        result += *curr;
        /*구현*/ // curr의 위치를 한칸 뒤로 보냄
    }
    return result;
}

int main() {
    int ary[] = { 10, 20, 30, 40, 50 };
    print(ary, 5);

    int* begin, * end;

    begin = ary;
    /*구현*/ // end 위치 초기화
    cout << sum(begin, end) << endl;

    return 0;
}
```

```
Microsoft Visual Studio 디버그 콘솔
10    20    30    40    50
150
```

4. 아래의 프로그램을 작성하시오. (/*구현*/ 부분을 채울 것)

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

int main() {
    int size;
    cout << "Size: ";
    cin >> size;
    const int len = 10;
    int staticArr[len]; // 정적 배열을 만드는 문법

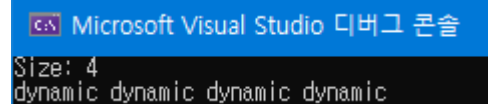
    // new, delete
    double* dynamicArr;
    /*구현*/ // 동적 배열 할당

    for (int i = 0; i < size; i++) {
        cout << "dynamicWt";
    }
    cout << endl;

    // 동적 배열과 정적 배열 사용법은 완전히 동일

    /*구현*/ // 할당 해제

    return 0;
}
```



```
Microsoft Visual Studio 디버그 콘솔
Size: 4
dynamic dynamic dynamic dynamic
```

5. 아래의 프로그램을 작성하시오. (/*구현*/ 부분을 채울 것)

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

int main() {
    int matrix[2][3] = { {1, 2, 3}, {4, 5, 6} };

    // matrix 각각의 성분을 초기화
    /*구현*/

    for (int row = 0; row < 2; row++) {
        for (int col = 0; col < 3; col++) {
            cout << /*구현*/ << "Wt"; // 각각 출력
        }
        cout << endl;
    }

    return 0;
}
```



6. 아래의 프로그램을 작성하시오. (/구현/ 부분을 채울 것)

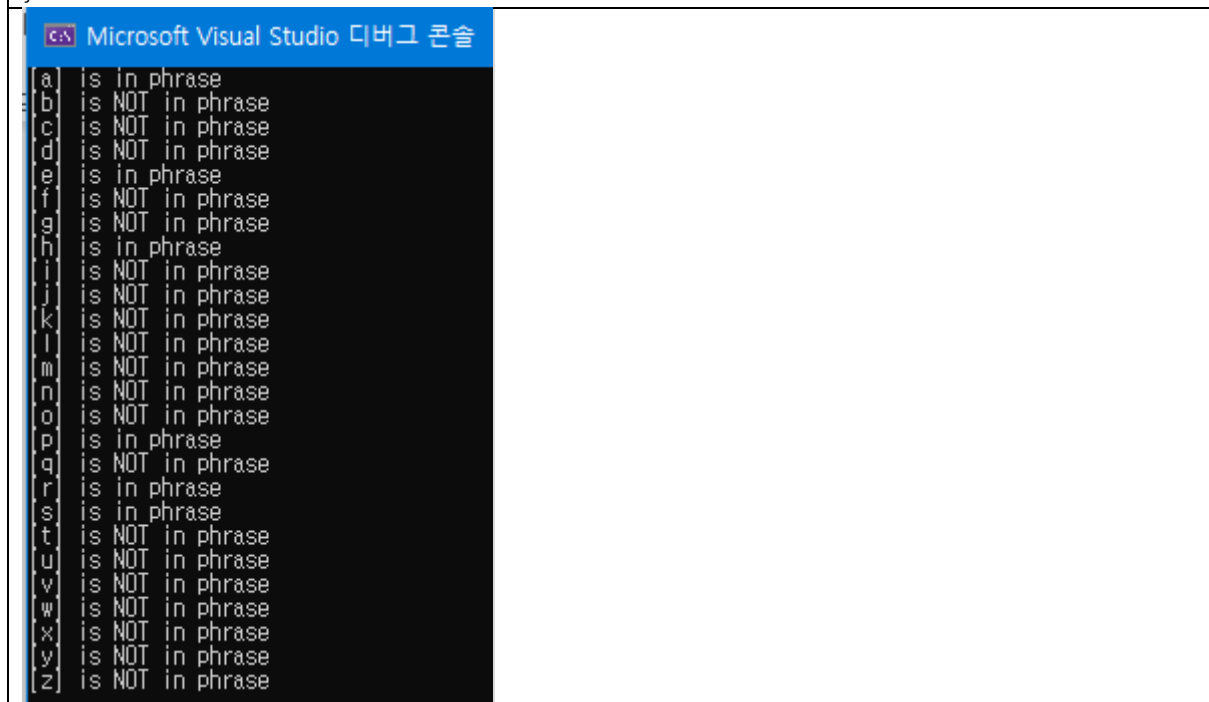
```
# include <iostream>
# include <vector>
using namespace std;

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

int main() {
    //          012345(6)
    const char* ch = "phrase";// ch[]
    // phrase(0)==NULL

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

    return 0;
}
}
```



<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;
}

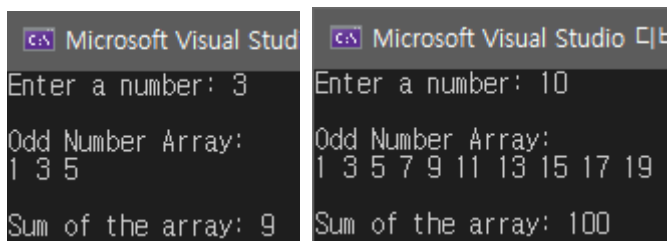
int main() {
    int n;
    cout << "Enter a number: ";
    cin >> n;

    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 */ }

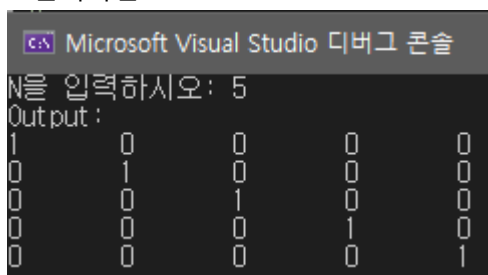
int main() {
    int n = 0;
    cout << "N을 입력하시오: ";
    cin >> n;
    if (n < 1) {
        cout << "n행렬을 생성할 수 없습니다.n" << endl;
        exit(EXIT_FAILURE);
    }

    int** table = buildTable(n);
    make_identity_matrix(table, n);
    printTable(table, n);

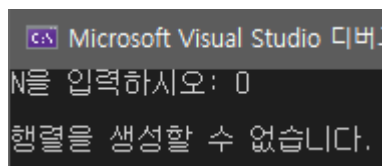
    for (int i = 0; i < n; i++)
        delete[] table[i];
    delete[] table;

    return 0;
}
```

4-출력화면:



```
Microsoft Visual Studio 디버그 콘솔
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
```



```
Microsoft Visual Studio 디버그 콘솔
N을 입력하시오: 0
행렬을 생성할 수 없습니다.
```