

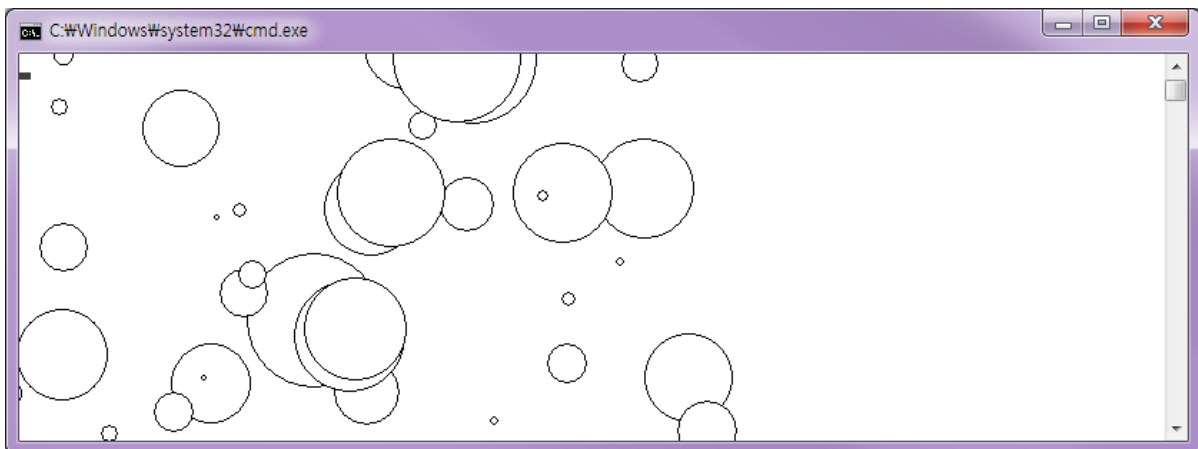
제6장 객체 배열과 벡터

1. 객체 배열을 이해한다.
2. 벡터(vector) 클래스를 사용할 수 있다.

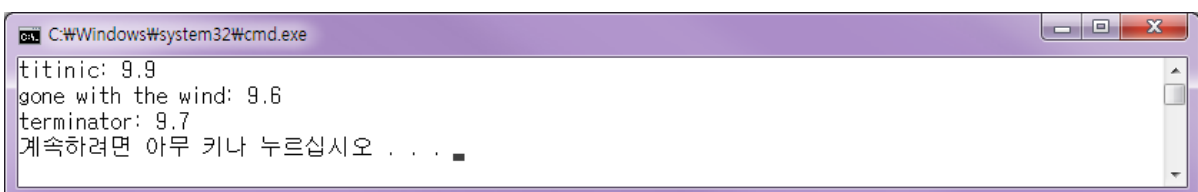
1

이번 장에서 만들어볼 프로그램

10개의 원을 저장하는 배열을 선언하고 'c' 키로 그리고 'q' 키로 종료



벡터를 이용하여 영화 정보를 저장했다가 출력하는 프로그램



2

6.2 객체 배열

- 원을 나타내는 객체를 여러 개 생성하여서 화면에 그려보는 프로그램을 작성하고자 한다.

```
Circle c1;  
Circle c2;  
Circle c3;  
...
```

3

객체 배열

문법 6.1

객체 배열 선언

```
클래스_이름      배열_이름[배열_크기];
```

```
Circle objArray[3];    objArray[0].calcArea();
```

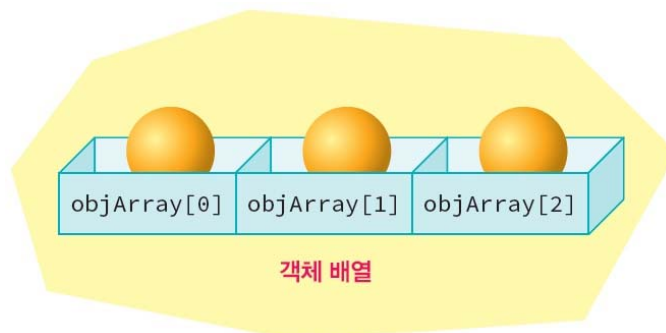


그림 6.1 객체 배열

4

예제

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

class Circle
{
public:
    int x, y;
    int radius;
    Circle() : x{ 0 }, y{ 0 }, radius{ 0 } { }
    Circle(int x, int y, int r) : x{ x }, y{ y }, radius{ r } { }
    void print() {
        cout << "반지름: " << radius << " @" << x << ", " << y
        << endl;
    }
};
```

5

예제

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

class Pizza {
public:
    Pizza(int s) : size(s) {}
    int size;           // 단위: 인치
};

Pizza createPizza() {
    Pizza p(10);
    return p;
}

int main() {
    Pizza pizza = createPizza();
    cout << pizza.size << "인치 피자" << endl;

    return 0;
}
```

6

```

int main(void)
{
    Circle objArray[10];

    for (Circle& c: objArray) {
        c.x = rand()%500;
        c.y = rand()%300;
        c.radius = rand()%100;
    }
    for (Circle c: objArray)
        c.print();

    return 0;
}

```



```

C:\Windows\system32\cmd.exe
반지름: 34 @(41, 167)
반지름: 24 @(0, 269)
반지름: 62 @(478, 258)
반지름: 45 @(464, 5)
반지름: 61 @(281, 27)
반지름: 42 @(491, 295)
반지름: 91 @(327, 36)
반지름: 53 @(104, 2)
반지름: 21 @(292, 82)
반지름: 95 @(216, 218)
계속하려면 아무 키나 누르십시오 . . .

```

객체 배열의 초기화

```

Circle objArray[10] = {
    Circle(100, 100, 30),
    Circle(100, 200, 50),
    Circle(100, 300, 80),
    ...
};

```

문제

- 다음 클래스에 대하여 다음 질문에 답하라

```
class Test {  
    int x;  
public:  
    Test() { x=0; cout << x << " "; }  
    Test(int x) : x{x} {}  
};
```

1. `Test a[2];` 문장이 실행될 때의 출력을 쓰시오.
2. `Test b[2] = { Test(1), Test(2) };` 문장이 실행될 때의 출력을 쓰시오.
3. 배열 `b`의 모든 요소를 출력하는 범위기반 루프를 작성하고 실행하시오. 필요하면 클래스 `Test`에 멤버 함수를 추가하라.

Lab: 책들을 저장해보자.

- 여러 권의 책을 저장할 수 있는 객체 배열 `books`를 생성하여 보자.



Solution

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

class Book
{
    string title;
    int price;
public:
    Book(string name, int price) : title{ name }, price{ price } {}
    void print() {
        cout << "제목:" << title << ",      가격:" << price << endl;
    }
};
```

11

Solution

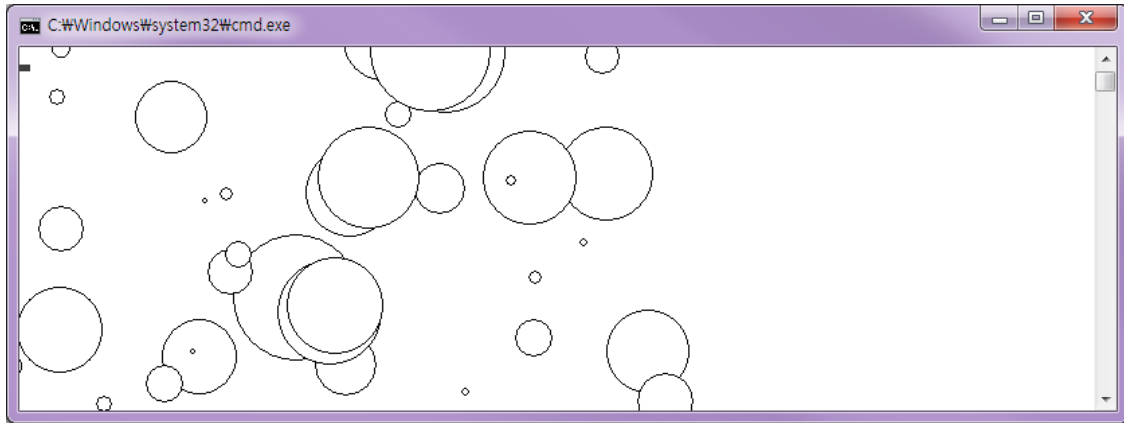
```
int main(void)
{
    Book books[2] = {
        Book("어서와 C++", 25000),
        Book("어서와 C ", 22000)
    };

    cout << "소장하고 있는 책 정보" << endl;
    cout << "=====" << endl;
    for (Book& b : books)
        b.print();
    cout << "=====" << endl;
    return 0;
}
```

12

Lab: 원들을 저장해보자.

- 10개의 원을 저장할 수 있는 배열을 선언하고 사용자가 키 'c'를 누르면 각각의 원의 위치와 반지름을 난수로 초기화한 후에 화면에 그린다. 사용자가 키 'q'를 누르면 프로그램을 종료한다.



13

Solution

```
#include <windows.h>
#include <conio.h>
#include <iostream>
using namespace std;
class Circle {
public:
    int x, y;
    int radius;
    Circle() : x{ 0 }, y{ 0 }, radius{ 0 } { }
    Circle(int x, int y, int r) : x{ x }, y{ y }, radius{ r } { }
    void draw()
    {
        int r = radius/2;
        HDC hdc = GetWindowDC(GetForegroundWindow());
        Ellipse(hdc, x-r, y-r, x+r, y+r);
    }
};
```

14

Solution

```
int main(void)
{
    Circle objArray[10];

    while(true){
        for (Circle& c: objArray) {
            c.x = rand()%500;
            c.y = rand()%300;
            c.radius = rand()%100;
            c.draw();
        }
        char ch = getch();
        if (ch == 'q') break;
    }
    return 0;
}
```

15

실습

- 사용자가 'm' 키를 누르면 배열 안의 모든 원을 오른쪽으로 100만큼 이동시켜서 그리시오.

6.3 벡터

- 벡터(**vector**)는 동적 배열이다.
- 컴파일 시간에 배열의 크기를 미리 결정할 필요가 없다.

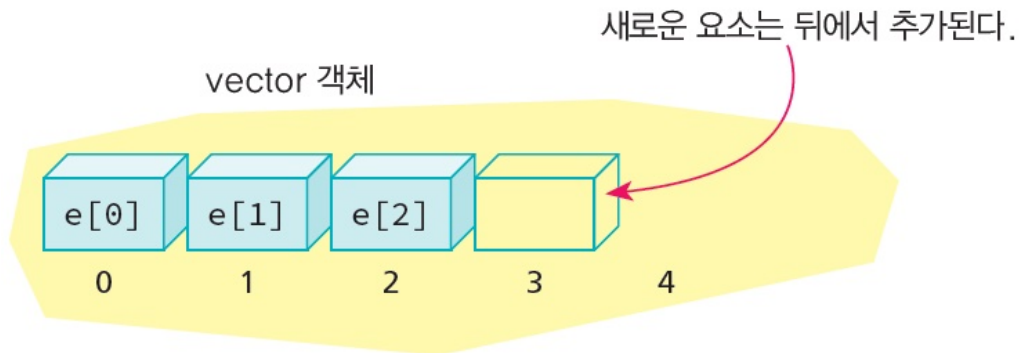


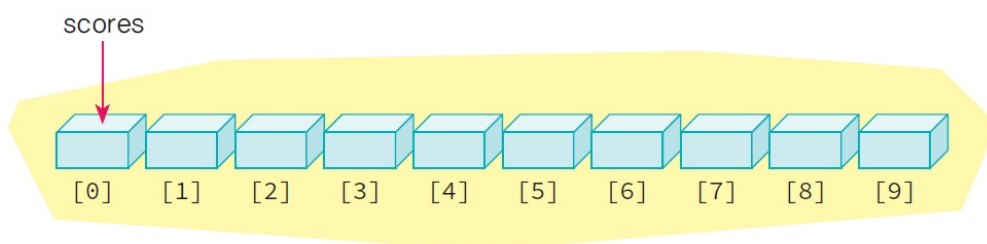
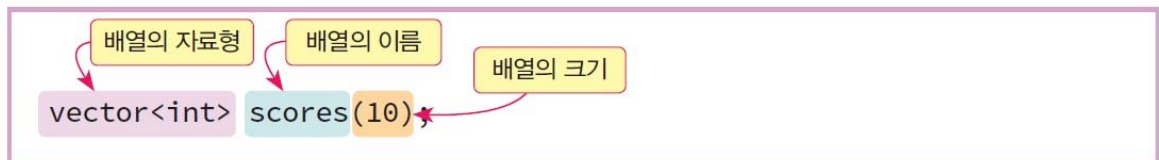
그림 6.2 벡터

17

벡터의 선언

문법 6.1

벡터 선언



18

벡터의 사용

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

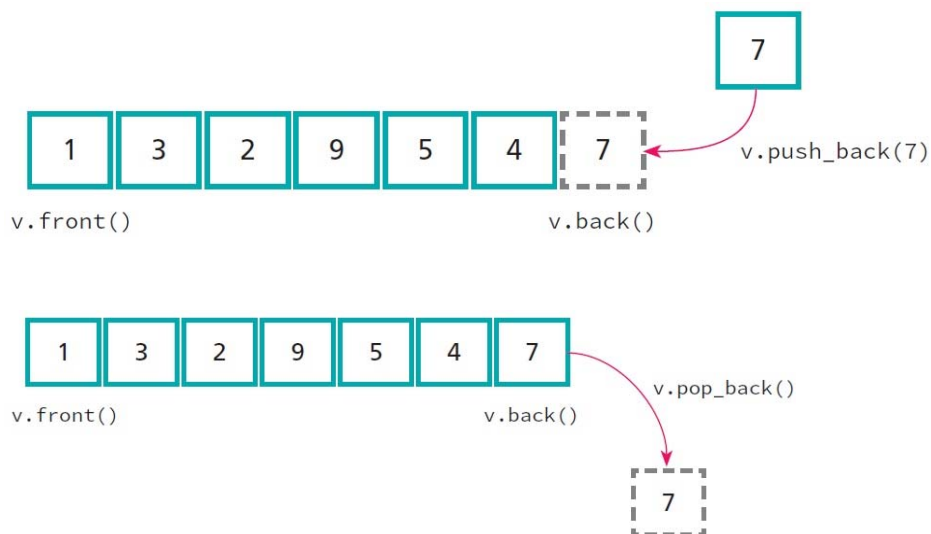
int main(void)
{
    vector<int> fibonacci { 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89 };

    for (auto& number : fibonacci)
        cout << number << ' ';

    cout << endl;
    return 0;
}
```



push_back()과 void pop_back()



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

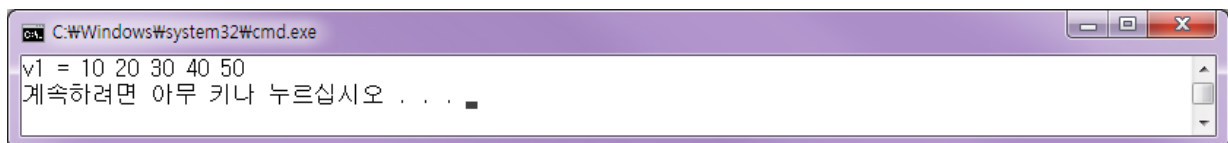
int main(void)
{
    vector<int> v1;

    v1.push_back(10);
    v1.push_back(20);
    v1.push_back(30);
    v1.push_back(40);
    v1.push_back(50);

    cout << "v1 = ";
    for (auto& e : v1) {
        cout << e << " ";
    }
    cout << endl;
    return 0;
}
```

21

실행결과



22

```

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

int main(void) {
    vector<int> v;
    for (int i = 0; i < 10; ++i) {
        v.push_back(i);
    }

    cout << "현재의 v = ";
    for (auto& e : v)
        cout << e << " ";
    cout << endl;

    cout << "삭제 요소 = ";
    // 벡터가 공백이 될 때까지 pop_back() 호출
    while (v.empty() != true) {
        cout << v.back() << " ";
        v.pop_back();
    }
    cout << endl;
}

```

23

실행결과

```

C:\Windows\system32\cmd.exe
현재의 v = 0 1 2 3 4 5 6 7 8 9
삭제 요소 = 9 8 7 6 5 4 3 2 1 0
계속하려면 아무 키나 누르십시오 . . .

```

24

벡터에서 요소의 위치

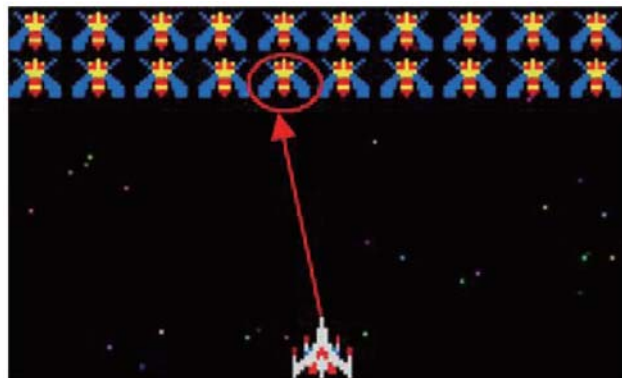
- 벡터에서 요소의 위치는 반복자(iterator)를 이용하여 표시한다.



```
for (auto p = v.begin(); p != v.end(); ++p)
    cout << *p << endl;
```

25

벡터의 중간에서 삭제



```
v.erase(v.begin()+i);
```

26

벡터와 연산자

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

int main(void)
{
    vector<int> v1{ 1, 2, 3, 4, 5 };
    vector<int> v2(v1);

    if (v1 == v2) {
        cout << "2개의 벡터가 일치합니다. " << endl;
    }
    return 0;
}
```



27

문제

- 사용자로부터 정수의 개수를 입력 받고, 정수의 개수만큼 사용자로부터 정수를 입력 받아서 벡터에 저장한다. 벡터에 저장된 정수 중에서 최대값과 최소값을 찾는 프로그램을 작성하시오. 범위 기반 루프를 사용한다.
 - 개수를 입력 받아서 `vector<int> vec(size);` 와 같이 벡터를 생성한다. `push_back()` 을 사용하여 정수를 벡터에 추가한다.

벡터에 문자열 저장

```
#include <iostream>
#include <vector>
#include <string>
using namespace std;
int main(void)
{
    vector<string> vec;           // 벡터를 생성한다.
    vec.push_back("MILK");       // 벡터의 끝에 자료를 저장한다.
    vec.push_back("BREAD");
    vec.push_back("BUTTER");
    for (auto e : vec) {
        cout << " " << e;
    }
    cout << endl;
    return 0;
}
```

29

벡터에 Circle 객체를 저장

```
#include <iostream>
#include <vector>
using namespace std;
class Circle
{
public:
    int x, y;
    int radius;
    Circle() : x{ 0 }, y{ 0 }, radius{ 0 } { }
    Circle(int x, int y, int r) : x{ x }, y{ y }, radius{ r } { }
    void print() {
        cout << "반지름: " << radius << " @" << x << ", " << y << ")" << endl;
    }
};
```

30

벡터에 Circle 객체를 저장

```
int main(void)
{
    vector<Circle> objArray;

    for (int i = 0; i < 10; i++) {
        Circle obj{ rand0%300, rand0%300, rand0%100 };
        objArray.push_back(obj);
    }
    for (Circle c : objArray)
        c.print();

    return 0;
}
```

31

실행결과



```
C:\Windows\system32\cmd.exe
반지름: 34 @(41, 167)
반지름: 24 @(100, 269)
반지름: 62 @(78, 258)
반지름: 45 @(164, 5)
반지름: 61 @(181, 27)
반지름: 42 @(191, 295)
반지름: 91 @(27, 36)
반지름: 53 @(204, 2)
반지름: 21 @(292, 82)
반지름: 95 @(116, 218)
계속하려면 아무 키나 누르십시오 . . .
```

32

벡터와 알고리즘

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <string>
using namespace std;
class Person {
private: string name;
        int age;
public:  Person::Person(string n, int a)    {
        name = n;
        age = a;
    }
    string get_name() { return name; }
    int get_age()     { return age; }
    void print()      {
        cout << name << " " << age << endl;
    }
};
```

33

벡터와 알고리즘

```
bool compare(Person &p, Person &q)
{
    return p.get_age() < q.get_age();
}
int main()
{
    vector<Person> list;
    list.push_back(Person("Kim", 30));
    list.push_back(Person("Park", 22));
    list.push_back(Person("Lee", 26));

    sort(list.begin(), list.end(), compare);
    for (auto& e : list) {
        e.print();
    }
    return 0;
}
```

34

실행결과



```
C:\Windows\system32\cmd.exe
Park 22
Lee 26
Kim 30
계속하려면 아무 키나 누르십시오 . . .
```

35

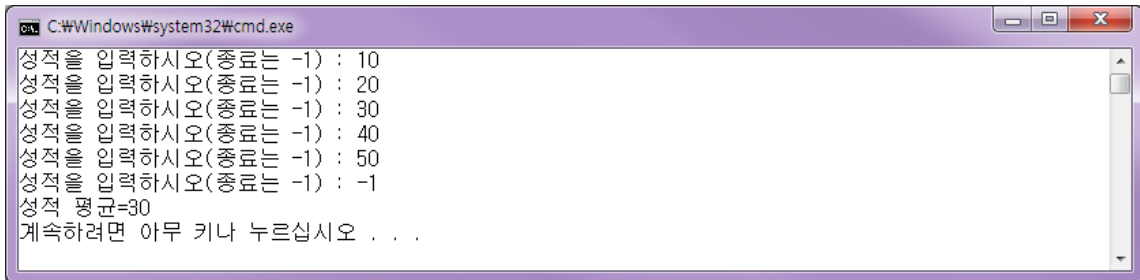
문제

- 사용자로부터 문자열의 개수를 입력 받는다. 그 개수만큼 문자열을 입력 받아서 벡터에 저장하는 프로그램을 작성하라. 벡터에 저장된 문자열을 알파벳순으로 정렬하여 화면에 출력하라. **sort()** 함수와 **string** 클래스의 연산자 < 과 > 을 사용한다.
 - `#include <algorithm>`을 추가하고 `sort(vec.begin(), vec.end(), compare);`를 사용한다.

36

Lab: 성적 평균 계산하기

- 학생들의 평균 성적을 계산하는 예제에서 학생이 몇 명인지 알 수 없다고 하자. 동적 배열인 벡터를 이용하여서 작성해보자.



```
C:\Windows\system32\cmd.exe
성적을 입력하시오(종료는 -1) : 10
성적을 입력하시오(종료는 -1) : 20
성적을 입력하시오(종료는 -1) : 30
성적을 입력하시오(종료는 -1) : 40
성적을 입력하시오(종료는 -1) : 50
성적을 입력하시오(종료는 -1) : -1
성적 평균=30
계속하려면 아무 키나 누르십시오 . . .
```

37

```
#include <iostream>
#include <vector>
using namespace std;
int main() {
    vector<int> scores;          // int 동적 배열을 생성한다.
    int i, sum = 0;
    while (true) {
        int score;
        cout << "성적을 입력하시오(종료는 -1) : ";
        cin >> score;
        if (score == -1) break;
        scores.push_back(score);
    }
    for (auto& value : scores) {
        sum += value;
    }
    double avg = (double)sum / scores.size();
    cout << "성적 평균=" << avg << endl;

    return 0;
}
```

38

Lab: 영화 정보 저장

- 벡터를 이용하여 영화에 대한 정보를 저장했다가 출력하는 프로그램을 작성해보자.



39

Solution

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

class Movie
{
private:
    string title;
    double rating;
public:
    Movie(string t = "", double r = 0.0) { title = t; rating = r; }
    void print_movie() { cout << title << ": " << rating << endl; }
};
```

40

Solution

```
int main(void)
{
    vector<Movie> movies;

    movies.push_back(Movie("titinic", 9.9));
    movies.push_back(Movie("gone with the wind", 9.6));
    movies.push_back(Movie("terminator", 9.7));

    for (auto& e : movies)
        e.print_movie();

    return 0;
}
```

41

6.4 array 클래스

- **vector**는 생성과 소멸을 하는데 상당한 시간이 소요된다. 따라서 **vector**의 장점이 많지만 성능 때문에 기존의 배열을 사용하는 경우도 많다.
- 이 문제를 해결하기 위해 **C++11**에서는 **std::array**를 새롭게 제시하였다. **array** 클래스를 사용하면 벡터의 장점과 기존 배열의 성능을 동시에 누릴 수 있다.

42

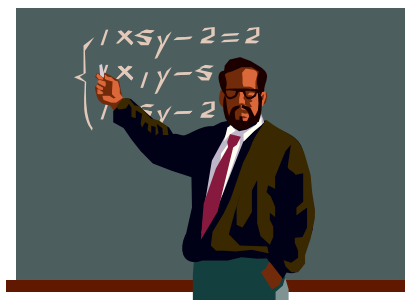
예제

```
#include <iostream>
#include <array>
using namespace std;
int main()
{
    array<int, 3> list{ 1, 2, 3 };
    for (int i = 0; i<list.size(); ++i)
        ++list[i];
    for (auto& elem : list)
        cout << elem << " ";
    cout << endl;
    return 0;
}
```



43

Q & A



44