У C ++ вказівники та масиви тісно пов'язані. Зазвичай компілятор перетворює масив у вказівник. За допомогою вказівника можна маніпулювати елементами масиву, як і за допомогою індексів.

Ім'я масиву це адреса його першого елемента. Відповідно через операцію розіменування ми можемо отримати значення за цією адресою:

|  |  |
| --- | --- |
|  | int a[] = {1, 2, 3, 4, 5};  std::cout << "a[0] = " << \*a << std::endl;    // a[0] = 1 |

Додаючи до адреси першого елемента деяке число, ми можемо отримати певний елемент масиву. Наприклад, в циклі пройдемося по всім елементам масиву:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | #include <iostream>   using namespace std;  int main()  {      const int n = 5;      int a[n] = {1, 2, 3, 4, 5};        for(int i=0; i < n; i++)      {          cout << "a[" << i << "]: address=" << a+i << "\tvalue=" << \*(a+i) <<endl;      }        return 0;  } |

Тобто, адреса другого елементу представлятиме вираз a + 1, а його значення - \*(a + 1).

Для додавання та віднімання діють ті ж правила, що і в операціях з вказівниками. Додавання одиниці означає додати до адреси значення, яке дорівнює розміру типу масиву. Так, у даному випадку масив має тип **int**, розмір якого, як правило, становить 4 байти, тому додати одиницю до адреси означає збільшення адреси на 4. Додаючи до адреси 2, ми збільшуємо значення адреси на 4 \* 2 = 8. І так далі.

Програма виведе на консоль наступний результат:

a[0]: address=[ якась адреса] value=1

a[1]: address=[ якась адреса] value=2

a[2]: address=[ якась адреса] value=3

a[3]: address=[ якась адреса] value=4

a[4]: address=[ якась адреса] value=5

Але при цьому ім'я масиву це не стандартний вказівник, і ми не можемо змінити його адресу, наприклад:

int a[5] = {1, 2, 3, 4, 5};

a++;            // так зробити не можна

int b = 8;

a = &b;         // так зробити не можна

але вказівник на перший елемент або на будь-який інший елемент масиву ми можемо зробити так:

int a[5] = {1, 2, 3, 4, 5};  
int \*c = a; //или int \*c = &a[0];

Оскільки двовимірний масив зберігається в пам'яті послідовно (по рядах), то так само можна створити вказівник на початок двовимірного масиву і перебирати всі його елементи, знаючи розмір масиву або адресу останнього елемента:

int a[2][2] = {{1, 2} , {3, 4}};

int \*\*c = (int\*\*)a //или int \*c = &a[0][0];

Але якщо треба працювати з невідомою кількістю змінних, то не знаючи кінець або кількість змінних краще використовувати масив.