Лекция: Статические и динамические массивы в C++

Массивы — это один из самых важных инструментов для работы с последовательностями данных в C++. Массивы могут быть как статическими, так и динамическими, и понимание различий между ними важно для эффективного использования памяти и ресурсов программы.

1. Статические массивы (Static Arrays)

a) Что такое статический массив?

Статический массив — это массив, размер которого задаётся на этапе компиляции и не может изменяться во время выполнения программы. Память под статические массивы выделяется в стеке.

b) Объявление статического массива

Размер статического массива должен быть известен заранее, на этапе компиляции.

- **Синтаксис:**

```
тип имя_массива[размер];
```

- **Пример:**

```
int numbers[5];        // массив из 5 целых чисел
int values[3] = {1, 2, 3};  // массив с инициализацией
```

c) Характеристики статических массивов

- **Фиксированный размер:** После объявления размер массива не может быть изменён.
- **Быстрая работа:** Память для массива выделяется в стеке, что делает операции с ним быстрыми.
- **Ограниченность по размеру:** Поскольку память в стеке ограничена, размер статических массивов не может быть слишком большим.

d) Пример работы со статическим массивом

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

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

    for (int i = 0; i < 5; i++) {
        cout << arr[i] << " ";
    }
```

```
    return 0;
}
```

e) Недостатки статических массивов

1. **Фиксированный размер:** Если заранее неизвестно, сколько элементов будет в массиве, использование статических массивов становится неэффективным.
2. **Ограниченное использование памяти:** Статические массивы хранятся в стеке, размер которого ограничен, поэтому для хранения больших массивов их использование не всегда возможно.

2. Динамические массивы (Dynamic Arrays)

a) Что такое динамический массив?

Динамический массив — это массив, размер которого может быть определён во время выполнения программы. Память для динамических массивов выделяется в куче (heap), что позволяет работать с большими объёмами данных.

b) Создание динамического массива

Для создания динамического массива в C++ используется оператор `new`, который выделяет память в куче.

- **Синтаксис:**

```
тип* имя_массива = new тип[размер];
```

- **Пример:**

```
int* arr = new int[5];  // динамический массив из 5 элементов
```

c) Работа с динамическими массивами

Доступ к элементам динамического массива осуществляется так же, как и в статическом, через индексы.

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

int main() {
    int size;
    cout << "Введите размер массива: ";
    cin >> size;

    int* arr = new int[size];  // выделение памяти для массива
```

```
    for (int i = 0; i < size; i++) {
        arr[i] = i + 1;  // заполнение массива
    }

    for (int i = 0; i < size; i++) {
        cout << arr[i] << " ";  // вывод массива
    }

    delete[] arr;  // освобождение памяти
    return 0;
}
```

d) Освобождение памяти

Для динамических массивов необходимо вручную освобождать память после завершения работы с массивом с помощью оператора `delete[]`. Если не освободить память, произойдёт утечка памяти.

```
delete[] имя_массива;
```

e) Преимущества динамических массивов

1. **Гибкость в размере:** Размер массива можно задавать в момент выполнения программы, что делает динамические массивы более гибкими, особенно в тех случаях, когда количество элементов неизвестно заранее.
2. **Использование памяти в куче:** Динамические массивы используют кучу, а не стек, что позволяет создавать массивы большого размера.

f) Недостатки динамических массивов

1. **Необходимость ручного управления памятью:** Программист должен вручную выделять и освобождать память. Неправильное управление памятью может привести к утечкам памяти или к сегментационным ошибкам.
2. **Медленная работа:** Операции с памятью в куче, как правило, медленнее, чем с памятью в стеке.

3. Сравнение статических и динамических массивов

Характеристика	Статический массив	Динамический массив
Размер	Определён на этапе компиляции	Определяется во время выполнения
Память	Выделяется в стеке	Выделяется в куче
Управление памятью	Управляется автоматически	Необходимо вручное освобождение
Скорость доступа	Быстрый доступ	Медленнее из-за работы с кучей

Характеристика	Статический массив	Динамический массив
Размер массива	Ограничен размером стека	Может быть большим, так как хранится в куче

4. Векторы: Альтернатива динамическим массивам

Для упрощения работы с динамическими массивами в C++ существует контейнер `std::vector`, который является частью стандартной библиотеки STL (Standard Template Library). Векторы автоматически управляют памятью, увеличивая её по мере необходимости.

- **Объявление и использование вектора:**

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

int main() {
    vector<int> vec;  // создание пустого вектора

    vec.push_back(10);  // добавление элемента в конец
    vec.push_back(20);

    for (int i = 0; i < vec.size(); i++) {
        cout << vec[i] << " ";
    }

    return 0;
}
```

Преимущества использования векторов:

1. **Автоматическое управление памятью:** Не нужно вручную выделять и освобождать память.
2. **Изменяемый размер:** Вектор автоматически увеличивает свой размер по мере добавления элементов.
3. **Удобные методы:** Вектор предоставляет множество методов для работы с элементами, такие как `push_back()`, `size()`, `clear()` и другие.

5. Пример использования статического и динамического массива

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

int main() {
    // Статический массив
    int staticArray[3] = {1, 2, 3};
    cout << "Статический массив: ";
    for (int i = 0; i < 3; i++) {
```

```
        cout << staticArray[i] << " ";
    }
    cout << endl;

    // Динамический массив
    int size;
    cout << "Введите размер динамического массива: ";
    cin >> size;

    int* dynamicArray = new int[size];  // динамический массив

    for (int i = 0; i < size; i++) {
        dynamicArray[i] = i + 1;  // заполнение массива
    }

    cout << "Динамический массив: ";
    for (int i = 0; i < size; i++) {
        cout << dynamicArray[i] << " ";  // вывод массива
    }
    cout << endl;

    delete[] dynamicArray;  // освобождение памяти

    return 0;
}
```

Заключение

Статические и динамические массивы — это важные инструменты для работы с данными в C++. Статические массивы эффективны и быстры, но ограничены по размеру. Динамические массивы гибки в размере, но требуют ручного управления памятью. В зависимости от задач, программисты выбирают подходящий тип массивов, а для ещё большей гибкости и удобства используют векторы из стандартной библиотеки C++.