

Класи. Об'єкти та методи

Класи

У світі об'єктно-орієнтованого програмування типи даних можуть не тільки містити дані, але і функції, які будуть працювати з цими даними. Для визначення такого типу даних в C ++ використовується ключове слово **class**. Використання ключового слова `class` визначає новий користувацький тип даних - клас.

У C ++ класи дуже схожі на структури, за винятком того, що вони забезпечують набагато більшу потужність і гнучкість. Фактично структура та клас по функціоналу ідентичні:

```
1 struct DateStruct
2 {
3     int day;
4     int month;
5     int year;
6 };
7
8 class DateClass
9 {
10 public:
11     int m_day;
12     int m_month;
13     int m_year;
14 };
```

Оголошення класу не призводить до виділення будь-якої пам'яті. Для використання класу потрібно оголосити змінну цього типу класу:

```
1 DateClass today { 12, 11, 2018 }; // объявляем переменную класса DateClass
```

Методи класів

Крім зберігання даних, класи також можуть містити і функції! Функції, що оголошені всередині класу, називаються **функціями-членами** або **методами**. Методи можуть бути визначені всередині або поза класом.

Клас Date з методом виведення дати:

```
1 class DateClass
2 {
3 public:
4     int m_day;
5     int m_month;
6     int m_year;
7
8     void print() // определяем функцию-член
9     {
10         std::cout << m_day << "/" << m_month << "/" << m_year;
11     }
12 };
```

До членів (змінних і функцій) класу доступ здійснюється через оператор вибору членів (.):

```
1 #include <iostream>
2
3 class DateClass
4 {
5 public:
6     int m_day;
7     int m_month;
8     int m_year;
9
10    void print()
11    {
12        std::cout << m_day << "/" << m_month << "/" << m_year;
13    }
14 };
15
16 int main()
17 {
18     DateClass today { 12, 11, 2018 };
19
20     today.m_day = 18; // используем оператор выбора членов для выбора переменной-члена объекта today
21     today.print(); // используем оператор выбора членов для вызова метода объекта today класса Date
22
23     return 0;
24 }
```

Результат виконання програми:

18/11/2018

Методи класу працюють так: всі виклики функцій-членів повинні бути пов'язані з об'єктом класу. Коли викликаємо `today.print()`, то повідомляємо компілятору викликати метод `print()` об'єкта `today`.

Розглянемо визначення методу `print ()` ще раз:

```
1 void print() // определяем функцию-член
2 {
3     std::cout << m_day << "/" << m_month << "/" << m_year;
4 }
```

На що фактично посилаються `m_day`, `m_month` і `m_year`? Вони посилаються на зв'язаний об'єкт `today`.

Тому, при виклику `today.print()`, компілятор інтерпретує:

```
m_day як today.m_day;
m_month як today.m_month;
m_year як today.m_year.
```

Використання префікса `m_` (англ. «M» = «**m**embers») для змінних-членів допомагає розрізнити змінні-члени від параметрів функції або локальних

змінних всередині методів класу. Це корисно з кількох причин:

- коли ми бачимо змінну з префіксом `m_`, то ми розуміємо, що працюємо зі змінною-членом класу.
- на відміну від параметрів функції або локальних змінних, оголошених всередині функції, змінні-члени оголошуються у визначенні класу. Отже, якщо ми хочемо знати, як оголошена змінна з префіксом `m_`, то ми розуміємо, що шукати потрібно в визначенні класу, а не всередині функції.

Зазвичай програмісти пишуть імена класів з великої літери. **Правило: Пишіть імена класів з великої літери.**

Ще один приклад програми з використанням класу:

```
1  #include <iostream>
2  #include <string>
3
4  class Employee
5  {
6  public:
7      std::string m_name;
8      int m_id;
9      double m_wage;
10
11     // Метод вивода інформації о работнике на экран
12     void print()
13     {
14         std::cout << "Name: " << m_name <<
15             "\nId: " << m_id <<
16             "\nWage: $" << m_wage << '\n';
17     }
18 };
19
20 int main()
21 {
22     // Объявляем двух работников
23     Employee john { "John", 5, 30.00 };
24     Employee max { "Max", 6, 32.75 };
25
26     // Выводим информацию о работниках на экран
27     john.print();
28     std::cout<<std::endl;
29     max.print();
30
31     return 0;
32 }
```

Результат виконання програми:

Name: John
Id: 5
Wage: \$30

Name: Max
Id: 6
Wage: \$32.75

Завдання №1

Створіть клас *Numbers*, який містить два цілих числа. Цей клас повинен мати дві змінні-члени для зберігання цих двох цілих чисел. Ви також повинні створити два методи:

- метод *set ()*, який дозволить присвоювати значення змінним;
- метод *print ()*, який буде виводити значення змінних.

Виконання наступної функції *main ()*:

```
1 int main()
2 {
3     Numbers n1;
4     n1.set(3, 3); // инициализируем объект n1 значениями 3 и 3
5
6     Numbers n2{ 4, 4 }; // инициализируем объект n2 значениями 4 и 4
7
8     n1.print();
9     n2.print();
10
11     return 0;
12 }
```

Програма видає такий результат:

```
Numbers(3, 3)
Numbers(4, 4)
```