Модуль 12

- Шаблоны функций
- Шаблоны классов
- Специализации шаблонов
- Variadic templates

Шаблоны функций

```
int max(int a, int b)
{
    return (a > b) ? a : b;
}


double max(double a, double b)
{
    return (a > b) ? a : b;
}
```

Для каждого типа перегружать функцию?

Шаблоны функций

Шаблоны функций — это функции, которые служат образцом для создания других подобных функций.

Главная идея — создание функций без указания точного типа(ов) некоторых или всех переменных. Для этого мы определяем функцию, указывая **тип параметра шаблона**, который используется вместо любого типа данных.

```
T max(T a, T b)
{
    return (a > b) ? a : b;
}
```

Шаблоны функций

```
template <typename T>
T max(T a, T b)
{
    return (a > b) ? a : b;
}
```

Шаблоны функций

!

Когда компилятор встречает вызов шаблона функции, он копирует шаблон функции и заменяет типы параметров шаблона функции фактическими (передаваемыми) типами данных.

Если будет создан шаблон функции, но не будет вызван, то экземпляр этого шаблона создан не будут.

Шаблоны функций

Недостатки:

- ➔ Некоторые старые компиляторы могут не поддерживать шаблоны функций или поддерживать, но с ограничениями. Однако сейчас это уже не такая проблема, как раньше.
- ➔ Шаблоны функций часто выдают сумасшедшие сообщения об ошибках, которые намного сложнее расшифровать, чем ошибки обычных функций.
- ➔ Шаблоны функций могут увеличить время компиляции и размер кода, так как один шаблон может быть «реализован» и перекомпилирован в нескольких файлах.

Шаблоны класса

Классы, как и функции, могут быть параметризованы типами или константами. Такие классы называются **шаблонными**. Примерами шаблонов классов являются все контейнеры стандартной библиотеки.

Шаблон не является ни классом, ни функцией — это трафарет, используемый для создания классов или функций. Таким образом, шаблоны работают не так, как обычные функции или классы.

Шаблоны класса

При работе с **обычными** классами необходимо помещать определение класса в заголовочный файл, а определения методов этого класса в отдельный файл .cpp с аналогичным именем.

Для шаблонных классов весь код должен быть в заголовочном файле.

Шаблоны класса

Параметр non-type

Параметр non-type в шаблоне — это специальный параметр шаблона, который заменяется не типом данных, а конкретным значением. Этим значением может быть:

- ➔ целочисленное значение или перечисление;
- ➔ указатель или ссылка на объект класса;
- ➔ указатель или ссылка на функцию;
- ➔ указатель или ссылка на метод класса;
- ➔ std::nullptr_t.

Шаблоны класса

Параметр non-type

Std::array из стандартной библиотеки использует такой подход при выделении памяти.

`std::array<int, 5>`, `int` является параметром типа, а `5` — параметром non-type в шаблоне класса!

Явная специализация шаблона

При создании экземпляра шаблона функции для определенного типа данных компилятор копирует шаблон функции и заменяет параметр типа шаблона функции на фактический (передаваемый) тип данных. Это означает, что все экземпляры функции имеют одну реализацию, но разные типы данных. Хотя в большинстве случаев это именно то, что требуется, иногда может понадобиться, чтобы реализация шаблона функции для одного типа данных отличалась от реализации шаблона функции для другого типа данных. Для этого используется **специализация шаблона функции**.

```
template <>
void Repository<double>::print()
{
    std::cout << std::scientific << m_value << '\n';
}
```

Явная специализация шаблона

Специализация шаблона класса (или ***«явная специализация шаблона класса»***) позволяет специализировать шаблон класса для работы с определенным типом данных (или сразу с несколькими типами данных, если есть несколько параметров шаблона).

Явная специализация шаблона

Частичная специализация шаблона позволяет выполнить специализацию шаблона класса (но не функции!), где некоторые (но не все) параметры шаблона явно определены.

Частичная специализация шаблона может использоваться **только с классами**, но не с отдельными функциями (для функций используется только полная специализация шаблона)

variadic templates

Вариативные (или переменные) шаблоны (variadic templates) - это возможность определения шаблонов с переменным числом аргументов. Это позволяет создавать функции и классы, которые могут принимать произвольное количество аргументов различных типов.