Шаблон функции

- Иногда не важен тип, с которым работает функция.
  - Если с ним можно проводить некоторые операции.
    - `int max(int a, int b) { return a > b ? a : b; }`
    - `double max(double a, double b) { return a > b ? a : b; }`
- Шаблон функции (шаблонная функция)
  - Определение:

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

  - Вызов:
    - `max(1, 2);`
    - `max(1.0, 2.0);`
  - Воплощение (инстанциация, instantiation).

Специализация шаблонов

Особое определение шаблона для сочетания типов-параметров

- max("Dmitry", "Kozliuk");
- ```
  const char* max(const char* a, const char* b) {
      return a > b ? a : b;
  }
  ```
  - Сравнение указателей не сравнивает символы строк, функция работает некорректно.
  - Нужно определить особую версию для T = **const char***
- ```
  template<>  // Параметры известны, но это всё равно шаблон.
  const char* max(const char* a, const char* b) {
      return strcmp(a, b) > 0 ? a : b;
  }
  ```

Шаблонные псевдонимы типов и шаблонные константы

- **using** Sample = vector<**double**>**;**
  - **typedef** vector<**double**> Sample**;**
- **template**<**typename** T>
  using PoolVector = vector<T**,** Pool>**;**
  - Второй тип-параметр у vector — распределитель памяти.
  - PoolVector<**int**> xs**;** // vector<**int,** Pool> xs;
- **template**<**typename** T>
  T **const** PI = 3.14159265358979**;**
  - **double** precise = PI**;** // 3.14159265358979
  - **float** estimate = PI**;** // 3.141593
  - Не требуются лишние приведения типов.

Физическое разделение кода и шаблоны

- Единица трансляции компилируется один раз.
- При инстанцировании (подстановке типов) шаблона создается новая версия кода.
- Один шаблон порождает несколько версий кода.
- Невозможно компилировать код шаблона один раз, вынося его в файл *.cpp.

```
#pragma once

template < typename T >
class LinkedList
{
public:
    void push_back (T value) {
        // …
    }
    void pop_back ();
    // …
};

templte < typename T >
void LinkedList < T > :: pop_back() {
    // …
}
```