

Алгоритмизация и программирование

Лекция 1

О преподавателях

Чабанов Владимир Викторович, старший преподаватель Кафедры компьютерной инженерии и моделирования Физико-технического института.

Кафедра: 310А

E-mail: chabanov.vv@cfuv.ru

VK: <https://vk.com/id444710087>

Тимофеева София Владимировна, старший преподаватель Кафедры компьютерной инженерии и моделирования Физико-технического института.

Кафедра: 310А

E-mail: timofeeva.sv@cfuv.ru

VK: <https://vk.com/id701465528>

Материалы курса

- Курс на мудле: <https://moodle.cfuv.ru/course/view.php?id=22885>;
- Материалы на GitHub: https://github.com/VladimirChabanov/alg_and_prog_03.03.03;

Практика

Крайне желательно приносить с собой ноутбук.

Практические и контрольные задания размещены в системе [Яндекс.Контест](#).

Доступ к практическим заданиям: [заполните форму](#);

Работа на лекции

Работа на лекции оценивается от 0 до 5 баллов.

Оценка определяется исходя из количества пройденных тестов. Не верный/полный ответ на тест тоже засчитывается как пройденный.

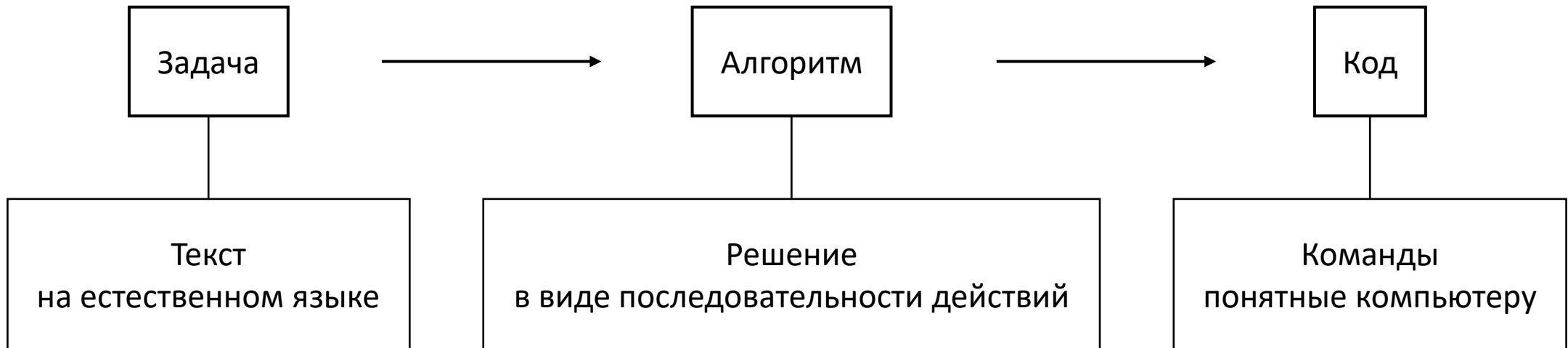
Тест нужно пройти в течение 5 минут после публикации ссылки.

О чём предмет

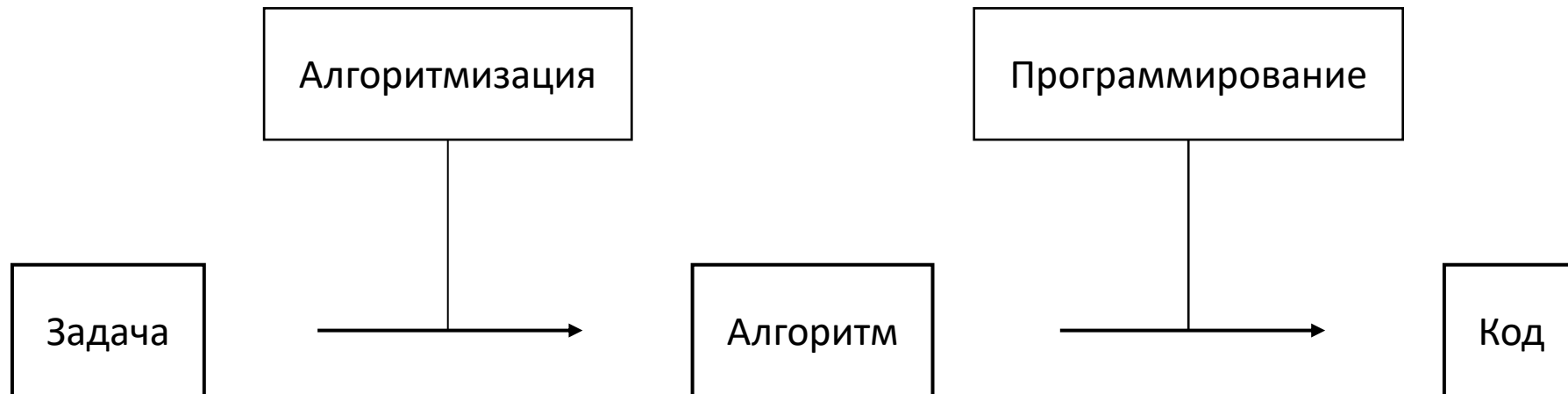
О предмете



О предмете



О предмете



На каком языке будем писать

C++



C++ - компилируемый, статически типизированный язык программирования общего назначения.

Основной принцип: zero-overhead

Создан: в начале 80-х (появление: 1983; выпуск: 1985)

Автор: Бьёрн Страуструп

Где писать код

Онлайн-компиляторы

[Wandbox](#)

- доступно большое количество языков (не только C++);

[Compiler Explorer](#)

- доступно большое количество языков (не только C++);
- для C++ доступно множество различных компиляторов в том числе экспериментальных;
- позволяет посмотреть ассемблерный код и сравнить его для разных вариантов сборки;
- есть встроенная поддержка некоторых популярных библиотек;

[OnlineGDB](#)

- можно запустить дебагер.

Локально

Visual Studio

- доступно большое количество языков (не только C++);
- "всё включено" (компилятор, отладчик, профилировщик);
- есть community версия;

Что такое код/программа на C++

Что такое код?

```
#include <iostream>

int main(){
    std::cout<<"Hello World";
    return 0;
}
```

Код – это текст, который написан в соответствии с "правилами" языка – стандартом языка.

Код должен быть сохранён в файл с определённым расширением (для C++: .cpp .h .hpp, ...);


- файл с расширением .cpp – файл с исходным кодом (Source Code File);
- файл с расширением .h – заголовочный файл (Header file);

Стандарт

Официальный сайт Standard C++ Foundation: <https://isocpp.org/>

Buy this standard

Format	Language
✓ PDF	English

CHF 198  Buy

Стандарт – это платный документ.

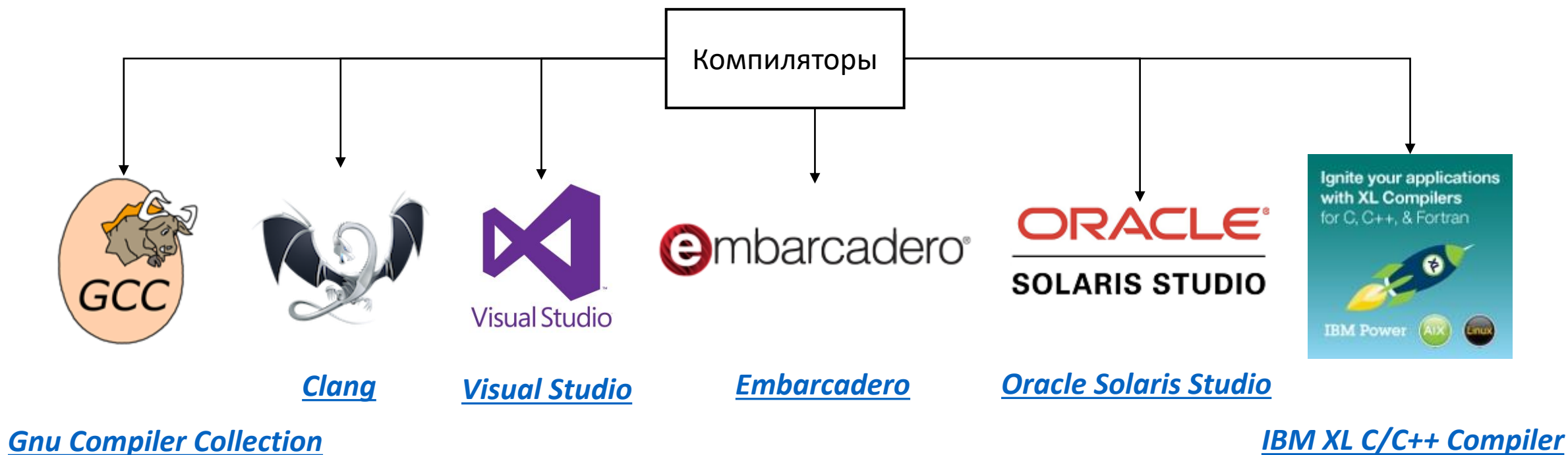
[Черновик стандарта](#) практически не отличаются от самого стандарта.

Стандарт – не учебник по языку, он больше похож на справочник.

Компиляторы

Стандарт – это текстовый документ и он не сможет преобразовать код в исполняемый файл.

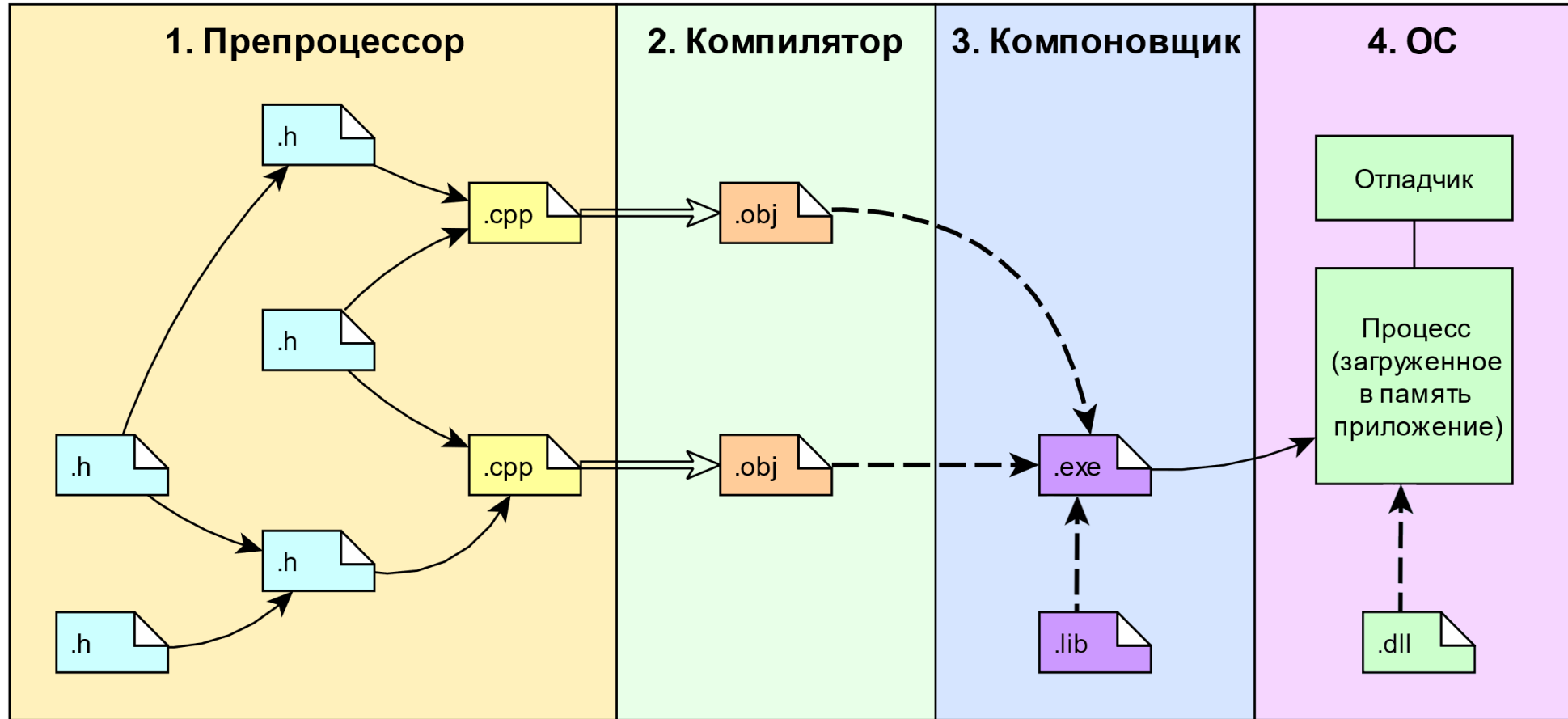
Компилятор – программа, переводящая написанный на языке программирования текст в набор машинных кодов.



Программа

Программа на C++ – это набор текстовых файлов (сpp и h), с исходным кодом. Для получения исполняемой программы (exe) эти файлы передаются компилятору.

Этапы компиляции (трансляции)



Состав языка

Алфавит

- Прописные и строчные латинские буквы:

a b c d e f g h i j k l m n o p q r s t u v w x y z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

- Арабские цифры:

0 1 2 3 4 5 6 7 8 9

- Специальные символы:

_ { } [] # () < > % : ; . ? * + - / ^ & | ~ ! = , \ " '

- Пробельные символы: пробел, табуляция, символ переход на новую строку;

Лексемы

Из символов формируются лексемы языка:

- Ключевые слова
- Знаки операций
- Разделители
- Литералы
- Пользовательские идентификаторы

Границы лексем определяются другими лексемами, такими, как разделители или знаки операций.

Ключевые слова

<code>alignas</code>	<code>constinit</code>	<code>false</code>	<code>public</code>	<code>true</code>
<code>alignof</code>	<code>const_cast</code>	<code>float</code>	<code>register</code>	<code>try</code>
<code>asm</code>	<code>continue</code>	<code>for</code>	<code>reinterpret_cast</code>	<code>typedef</code>
<code>auto</code>	<code>co_await</code>	<code>friend</code>	<code>requires</code>	<code>typeid</code>
<code>bool</code>	<code>co_return</code>	<code>goto</code>	<code>return</code>	<code>typename</code>
<code>break</code>	<code>co_yield</code>	<code>if</code>	<code>short</code>	<code>union</code>
<code>case</code>	<code>decltype</code>	<code>inline</code>	<code>signed</code>	<code>unsigned</code>
<code>catch</code>	<code>default</code>	<code>int</code>	<code>sizeof</code>	<code>using</code>
<code>char</code>	<code>delete</code>	<code>long</code>	<code>static</code>	<code>virtual</code>
<code>char8_t</code>	<code>do</code>	<code>mutable</code>	<code>static_assert</code>	<code>void</code>
<code>char16_t</code>	<code>double</code>	<code>namespace</code>	<code>static_cast</code>	<code>volatile</code>
<code>char32_t</code>	<code>dynamic_cast</code>	<code>new</code>	<code>struct</code>	<code>wchar_t</code>
<code>class</code>	<code>else</code>	<code>noexcept</code>	<code>switch</code>	<code>while</code>
<code>concept</code>	<code>enum</code>	<code>nullptr</code>	<code>template</code>	
<code>const</code>	<code>explicit</code>	<code>operator</code>	<code>this</code>	
<code>constexpr</code>	<code>export</code>	<code>private</code>	<code>thread_local</code>	
<code>constexpr</code>	<code>extern</code>	<code>protected</code>	<code>throw</code>	

<code>and</code>	<code>and_eq</code>	<code>bitand</code>	<code>bitor</code>	<code>compl</code>	<code>not</code>
<code>not_eq</code>	<code>or</code>	<code>or_eq</code>	<code>xor</code>	<code>xor_eq</code>	

Знаки операций

- Оператор начала директивы препроцессора:

%: %::

- Оператор пунктуации и знаки операций:

{ } [] ()

<: :> <% %> ; : ...

? :: . .* -> ->* ~

! + - * / % ^ & |

= += -= *= /= %= ^= &= |=

== != < > <= >= <=> && ||

<< >> <<= >>= ++ -- ,

and or xor not bitand bitor compl

and_eq or_eq xor_eq not_eq

Литералы

Существует несколько видов литералов:

- Целочисленный литерал
- Литерал чисел с плавающей точкой
- Символьный литерал
- Строковый литерал
- Литерал булевого типа
- Литерал типа указатель
- Литерал определённый пользователем

Пользовательские идентификаторы

Идентификатор – это имя программной сущности. В идентификаторе можно использовать латинские символы, цифры и символ подчёркивания. Прописные и строчные символы различаются, например: `name`, `Name`, `NAME` – три различных имени.

В качестве первого символа идентификатора запрещено использовать цифру. Кроме того, внутри имени не допускается использование пробела. Идентификатор не должен совпадать с ключевым словом.

Не рекомендуется начинать имена с символа подчеркивания или двух, т.к. они могут быть использованы в следующих версиях языка.

По стандарту, длина идентификатора не ограничена, но некоторые компиляторы могут налагать свои ограничения.

Выражения

Лексемы могут быть скомбинированы в выражения:

- Выражение;
- Полное выражение;

Комментарии

В C++ есть 2 вида комментариев:

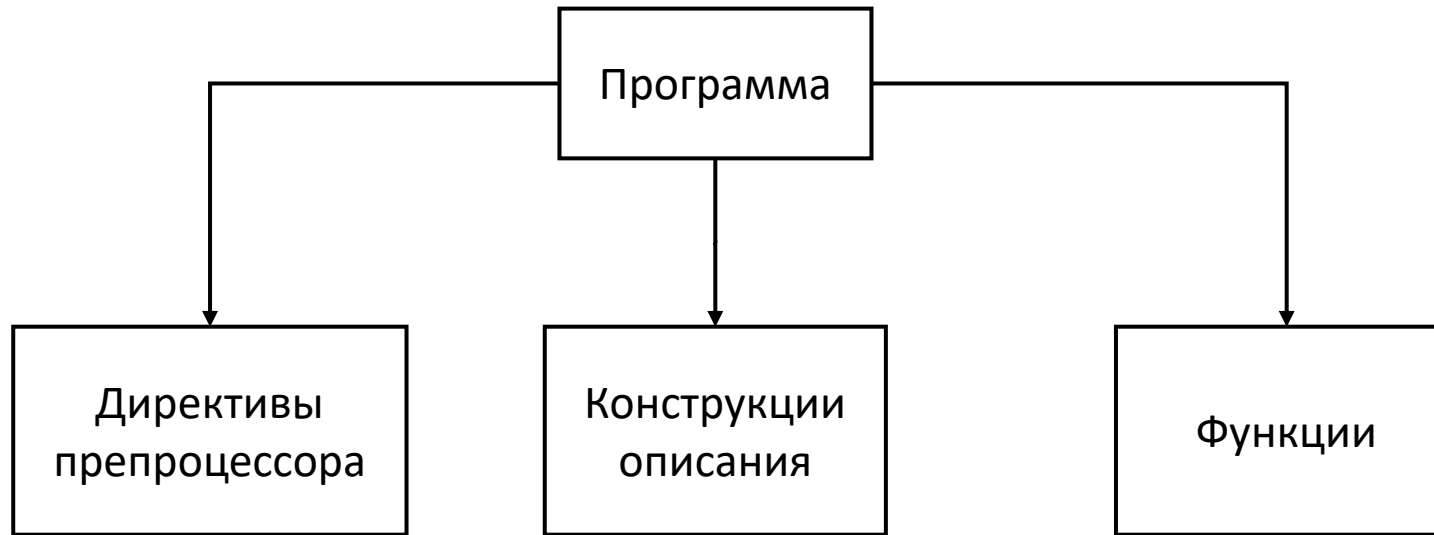
- Однострочные. Начинаются с `//` и заканчиваются перед символом конца строки;
- Многострочные. Начинаются `/*` и заканчиваются `*/`

Комментарии полностью удаляются из кода на стадии препроцессинга.

<https://wandbox.org/permlink/g919ArA0C3dqefZm>

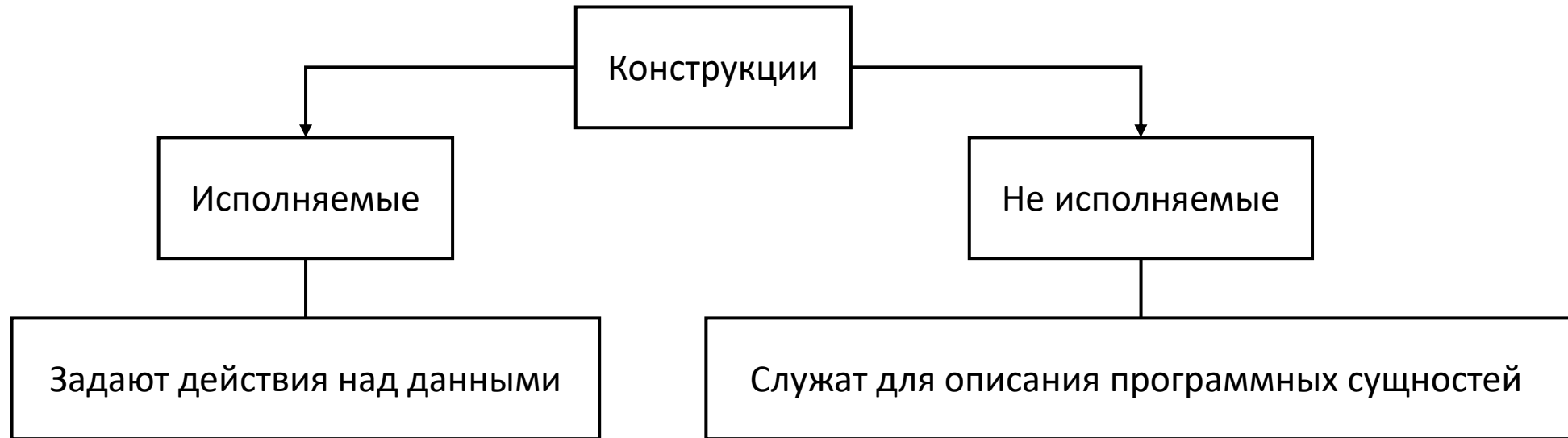
Структура программы

Базовая структура



<https://wandbox.org/permlink/HwXb9ql6TuUac2f1>

Конструкции



Точка входа

Одна из функций должна называться `main`. Эта функция является точкой входа в программу, т.е. исполнение программы начинается с первой инструкции функции `main` и завершается после выполнения последней.

В минимальном варианте программа может содержать только функцию `main` и больше ничего.

Формы main

Функция main должна иметь одну из следующих форм:

```
int main(){}  

```

```
int main(int argc, char *argv[]){}  

```

```
int main(int argc, char* argv[], char* envp[]){}  

```