Программирование на языке C++

Вводный курс

Александр Морозов
gelu.speculum@gmail.com

ИТМО, весенний семестр 2021

Содержание

О чём этот курс?

- Основные элементы языка C++
- Некоторые инструменты для разработки программ на C++
- Базовые навыки программирования

Почему C++?

- Язык сочетает черты низкоуровневого и высокоуровневого
- Позволяет как использовать сложные абстракции, так и прибегать к низкоуровневым оптимизациям и ручному управлению ресурсами
- Zero overhead abstractions
- Значительно более высокоуровневый, чем C, но в то же время может быть настолько же эффективным
- Один из самых распространенных прикладных языков

Место C++ в современном мире

- Графические оболочки (MS Windows UI, Aqua, KDE)
- Офисные пакеты (MS Office, OpenOffice)
- Графические редакторы и среды 3D моделирования (Photoshop, Maya)
- Компьютерные игры (CryEngine, Frostbite, Gamebryo, id Tech 4-, Source, Unreal Engine)
- CAD (Autodesk, Catia, FreeCAD)
- Браузеры и Javascript движки (Chrome, Firefox, V8, SpiderMonkey)
- Базы данных (MongoDB, частично MariaDB, MS SQL, Oracle, SAP DB, ScyllaDB)
- Системы информационного поиска, интернет поисковики (Google, Яндекс)
- Финтех (Bloomberg, Morgan Stanley, Itiviti)

Формат курса

- Лекции
- Дополнительные вебинары
- Небольшие примеры-иллюстрации к лекциям
- Задачи по мотивам примеров
- Большие задачи
- Соревнование по скорости для 2-й большой задачи
- Сдача задач через code review на github.com
- Экзамен

Некоторая литература

- Bjarne Stroustrup: Programming: Principles and Practice Using C++, 2014
  Программирование Принципы и практика использования C++
- Bjarne Stroustrup: The C++ Programming Language (Special edition), 2000
  Язык программирования C++ Специальное издание
- Bjarne Stroustrup: The Design and Evolution of C++, 1994
  Дизайн и эволюция C++
- Stanley Lippman: C++ Primer (5th Edition), 2012
  Язык программирования C++. Базовый курс
- Herb Sutter: Exceptional C++, 1999; More Exceptional C++, 2001
  Решение сложных задач на C++
- Nicolai Josuttis

Содержание

Первые языки программирования

Листинг 1: Машинный код

```
1   41 54
2   55
3   bf 00 22 60 00
4   53
5   48 83 ec 50
6   64 48 8b 04 25 28 00
7   00 00
8   48 89 44 24 48
9   31 c0
10  48 8d 44 24 10
11  48 89 e6
12  48 c7 44 24 08 00 00
13  00 00
14  c6 44 24 10 00
15  48 c7 44 24 28 00 00
16  00 00
17  48 89 04 24
18  48 8d 44 24 30
19  c6 44 24 30 00
20  48 89 44 24 20
21  e8 4c ff ff ff
22  48 8b 5c 24 08
23  48 8b 6c 24 28
24  48 39 eb
25  0f 87 fb 00 00 00
26  73 13
27  48 8b 44 24 20
```

Листинг 2: Ассемблер

```
1   push %r12
2   push %rbp
3   mov $0x602200,%edi
4   push %rbx
5   sub $0x50,%rsp
6   mov %fs:0x28,%rax
7
8   mov %rax,0x48(%rsp)
9   xor %eax,%eax
10  lea 0x10(%rsp),%rax
11  mov %rsp,%rsi
12  movq $0x0,0x8(%rsp)
13
14  movb $0x0,0x10(%rsp)
15  movq $0x0,0x28(%rsp)
16
17  mov %rax,(%rsp)
18  lea 0x30(%rsp),%rax
19  movb $0x0,0x30(%rsp)
20  mov %rax,0x20(%rsp)
21  callq 400bc0
22  mov 0x8(%rsp),%rbx
23  mov 0x28(%rsp),%rbp
24  cmp %rbp,%rbx
25  ja 400d82 <main+0x162>
26  jae 400c9c <main+0x7c>
27  mov 0x20(%rsp),%rax
```

Предшественники C++

- FORTRAN: язык математических вычислений, 1954
- Simula: объектно-ориентированный язык, 1965
- C: эффективный процедурный язык, 1972

Классификация языков программирования

- Компилируемые / интерпретируемые
- Императивные / декларативные
- Поддержка различных парадигм: ООП, функциональные, логические
- Статическая типизация / динамическая типизация
- Сильная типизация / слабая типизация
- Энергичные / ленивые

Появление C++: цели создателя

Бьярне Страуструп занимался моделированием распределенных аспектов операционных систем и ему нужен был язык:

- ООП
- пользовательские абстракции
- сильная типизация
- эффективность
- отсутствие «необоснованной стоимости» возможностей
- простота реализации (использование уже существующих инструментов)
- отсутствие излишних ограничений на стиль программирования

Краткая история C++

1979 - C with classes (расширение языка C классами, наследованием, более сильной типизацией, встраиваемыми функциями).
1983 - C++ (перегрузка функций и операторов, виртуальные функции, ссылки, типобезопасное управление памятью).
1985 - The C++ Programming Language (первое описание языка).
1989 - C++ 2.0 (множественное наследование, абстрактные классы, статические члены классов).
1990 - The Annotated C++ Reference Manual (шаблоны, исключения, пространства имен).
1992 - STL (обобщенная реализация различных структур данных и типовых алгоритмов).
1998 - C++98, первый ISO стандарт языка.

Краткая история C++, продолжение

1999 - Boost.
2003 - C++03, второй ISO стандарт, незначительные изменения.
2011 - C++11, новый стандарт, большие изменения и модернизация.
2013 - 4-е издание The C++ Programming Language.
2014 - C++14, дальнейшее развитие нового стандарта.
2017 - C++17, текущий *устоявшийся* стандарт языка.
2020 - C++20, текущий *опубликованный* стандарт языка.

Содержание

Абстрактная машина vs Настоящее железо

- Архитектура фон Неймана: общая память, АЛУ, УУ
- Последовательное исполнение
- Линейная непрерывная память
- Числа определяются в человеческих понятиях (целые, рациональные и т.п.)

- Особенности конкретной архитектуры, например, big-ending vs little-endian; битность процессора
- Целые и дробные числа представляются по-разному
- Много уровней памяти: регистры, кеш нескольких уровней, RAM
- Реальный и защищенный режим
- Виртуальная память
- Прерывания, системные вызовы, переключение контекста
- Параллелизм: внутри ядра процессора, между несколькими ядрами, между процессорами; разные гарантии синхронизации на разных архитектурах
- Инструкции различной сложности: RISC, CISC, векторные инструкции, сопроцессоры (“математический”, GPU)

Абстракции до C++11

- Последовательное исполнение в рамках observable behaviour
- Параллелизм отсутствует
- Целые числа подчиняются определенным правилам, без подробностей реализации
- Беззнаковые целые числа немного ближе к аппаратным подробностям
- Дробные числа имеют определенную точность и диапазон
- Исключения заданы лишь высокоуровневым поведением, без особенностей реализации
- Есть понятие упаковки сложных объектов в памяти, padding, требования к размещению в памяти заданы обтекаемо, в стандарте нет возможностей на это влиять

Что изменилось в C++11

- Появилась модель памяти, учитывающая параллельное исполнение
- Управление многопоточностью добавлено в стандартную библиотеку
- Конструкции для высокоуровневого управления параллелизмом

Гарантии языка и undefined behaviour

- Некоторые вещи язык гарантирует - вне зависимости от платформы, компилятора и иных внешних факторов. Например, `sizeof(int)<= sizeof(long)`.
- *Observable behaviour*: компилятор может менять программу, если её внешнее поведение не меняется.
- *Implementation defined behaviour*: поведение программы может различаться в зависимости от реализации компилятора (но это должно быть задокументировано).
- *Unspecified behaviour*: поведение зависит от реализации, но это не требуется документировать; каждый возможный вариант поведение должен быть корректным.
- *Undefined behaviour*: стандарт не накладывает никаких ограничений на поведение в этом случае.

Содержание

Пример программы на C++

```
 1  #include <iostream>
 2  #include <cstdlib>
 3  #include <algorithm>
 4  #include <string>
 5
 6  int main()
 7  {
 8      std::string direct, complementary;
 9      std::cin >> direct;
10      complementary.resize(direct.size());
11      std::transform(direct.begin(), direct.end(),
12                     complementary.begin(),
13                     [] (char x) {
14                         switch (x) {
15                             case 'A': return 'T';
16                             case 'T': return 'A';
17                             case 'C': return 'G';
18                             case 'G': return 'C';
19                             default: throw "Bad input";
20                         }
21                     });
22      std::cout << complementary << std::endl;
23      return 0;
24  }
```

Сложности грамматики C++

```
1       // variables 'x', 'y', 'z'
2       int(x), y, *const z;
3
4       // expression '(int(x)), (y), (new int))'
5       int(x), y, new int;
6
7       // ??
8       B b(A());
```

Эволюционные сложности C++

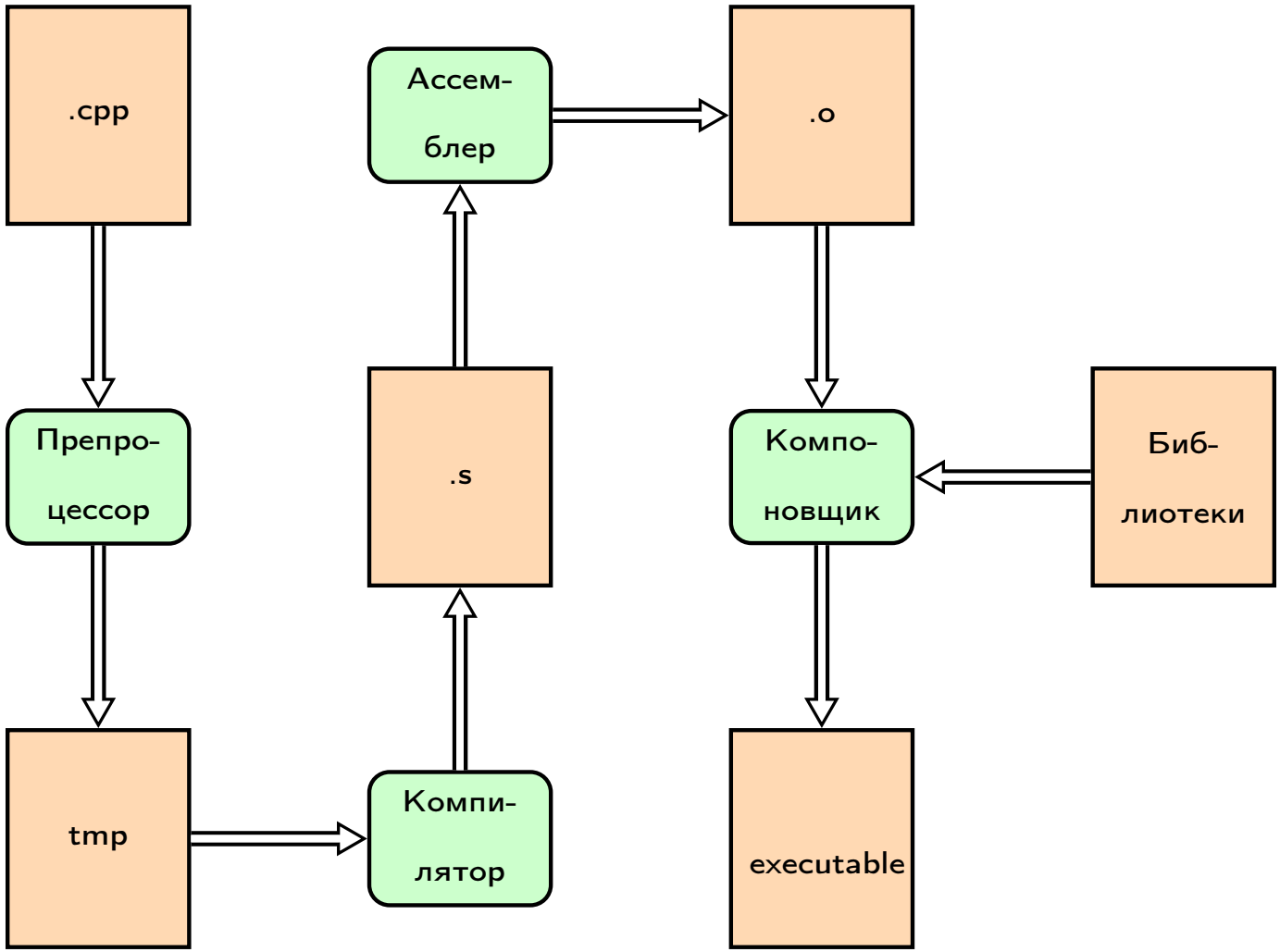
```
1       static vs static vs static
2
3       struct vs class vs typename
```

Сложности правил языка

- 9 различных видов инициализации переменных
- 21 правило упорядочивания исполнения
- 13 правил выбора лучшего кандидата при перегрузке функций
- и ещё больше веселья в шаблонах

Содержание

3 этапа трансляции C++



Трансляция C++ на примере gcc

Трансляция программы из одного файла:

```
g++ -std=c++17 -o prog prog.cpp
```

Результат препроцессора:

```
g++ -std=c++17 -E prog.cpp
```

Ассемблерный код:

```
g++ -std=c++17 -S prog.cpp
```

Объектный файл:

```
g++ -std=c++17 -c prog.cpp
```

Дизассемблирование:

```
objdump -dS prog
```

Объектные файлы

Объектные файлы обычно состоят из различных секций.
Например: заголовок, секция кода, секция данных, отладочная информация.
Сущности ссылаются по именам, адреса в памяти не назначены.
Mangling: имена сущностей из текста программы не всегда могут быть перенесены в имена в объектном файле. Для C обычно соответствие точное (хотя некоторые реализации добавляют к имени дополнительную информацию). В C++ структура имен более сложная и они приводятся к уникальным строковым именам по определенному алгоритму (зависит от реализации).
Например, имя `Space::Outer::Inner::code` может быть преобразовано в `_ZN5Space5Outer5Inner4codeE`.

Online компиляторы

- Coliru https://coliru.stacked-crooked.com/
- Wandbox https://wandbox.org/
- Godbolt https://godbolt.org/
- CPP Insights https://cppinsights.io/

Содержание

Практические задания

- Маленькие задачи – 4 варианта по мотивам примера из лекции
- Большие задачи
  - 4 набора по 3 задачи
  - наборы в целом сбалансированы

Дедлайны

- Маленькие задачи: март – апрель
- Большие задачи: апрель – начало июня
- Финальный дедлайн – экзамен, в первой половине сессии

Работа над задачей

- 1-й дедлайн – дедлайн оформления
- 2-й дедлайн – дедлайн приёмки
- 2 недели между двумя дедлайнами
- code review – несколько итераций
- работа **строго индивидуальная**
- по одной из больших задач – собеседование
- 14 “поздних” дней

Соревнование по скорости

- параллельно с процессом ревью
- 2 прогона, можно внести изменения после 1-го
- места распределяются по перцентилям
- множитель оценки $1 \ldots 3$

Оценивание задач

- базовая стоимость маленьких задач – 5-10 баллов
- базовая стоимость больших задач – 15-40 баллов
- суммарная базовая стоимость 3 больших задач – 75 баллов
- штрафные баллы начиная со второго ревью
- соревнование по скорости – коэффициент масштабирования оценки за задачу

Экзамен и финальная оценка

- допуск к экзамену – 50 баллов
- без прихода на экзамен:
  $E \geq 65$; $D \geq 85$; $C \geq 100$; $B \geq 125$; $A \geq 150$
- первая часть экзамена – блиц-опрос; $+1$ к финальной оценке
- вторая часть экзамена (по желанию) – 2 теоретических вопроса
- результат второй части экзамена – $-2 \ldots +2$ к финальной оценке