Историческая справка: особенности языка C и его экосистемы

Луцив Дмитрий Вадимович
Кафедра системного программирования СПбГУ

Содержание

Dennis Ritchie: The Development of the C Language

https://www.bell-labs.com/usr/dmr/www/chist.html

- Сеттинг
- Корни в других языках
- Использование сейчас

Мы периодически будем обращаться к этому тексту

Мотивация

Поколения ЭВМ

1. 1940е–1950е. Электромагнитные реле и радиолампы: 10^5 вт, большие машинные залы; доступны военным и работающим на них физикам
2. 1950е–1960е. Полупроводники (транзисторы, диоды): 10^4 вт, несколько стоек; доступны крупным учреждениям, банкам
3. 1960е–1970е. Интегральные схемы: $10^2 - 10^3$ вт, одна или несколько стоек; доступны небольшим учреждениям и лабораториям
4. 1970е–1980е–н.в. Микропроцессоры в одной интегральной схеме: $10 - 10^2$ вт, небольшой корпус, доступны мелким организациям, позже — физическим лицам

Сеттинг

Подход 1960-х вцелом

- Мэйнфреймы типа IBM/360 или GE-645
- Языки программирования типа PL/I
- ОС типа OS/360 или Multics
- Пакетное управленеи заданиями в духе JCL
- Нет универсальных интерактивных оболочек

Всё тяжёлое, толстое и сложное

Сеттинг

Подход 1960-х вцелом

- Мэйнфреймы типа IBM/360 или GE-645
- Языки программирования типа PL/I
- ОС типа OS/360 или Multics
- Пакетное управленеи заданиями в духе JCL
- Нет универсальных интерактивных оболочек

Всё тяжёлое, толстое и сложное

Изменение подхода 1970-х

- Более простые *и дешёвые* миникомпьютеры типа DEC PDP-7
- Более вольное их использование
- Зоопарк архитектур и семейств

Появление Unix

Киллер-фичи Unix:

- Древовидная иерархическая файловая система
- Агностический подход к данным в файлах: файлы стали просто последовательностями байтов, а раньше программисты работали с форматированными датасетами, это было часто быстрее, но сложнее
- Интерактивная пользовательская командная оболочка

Появление Unix

Киллер-фичи Unix:

- Древовидная иерархическая файловая система
- Агностический подход к данным в файлах: файлы стали просто последовательностями байтов, а раньше программисты работали с форматированными датасетами, это было часто быстрее, но сложнее
- Интерактивная пользовательская командная оболочка
- Ну и ещё там всякое, о чём позже...

До и после Unix

- Multics умела и предоставляла многие упомянутые возможности, но была слишком сложной, а пользователи мечтали о минимализме
- Подходы Unix актуален и по прошествии 50 лет: его аспекты позже проявлялись в дизайне, например DOS и Windows

Милый документальный фильм от AT&T:

- https://youtu.be/tc4ROCJYbm0

Ранняя история

Популярные языки 1960-х и ранее

- Fortran: один из первых, высокоуровневый, быстрый, вычислительный
- COBOL: для деловых задач
- PL/I: общего назначения, мощный и сложный, подходил для системного программирования
- Языки ассемблера для разных архитектур, не переносимые

Популярные языки 1960-х и ранее

- Fortran: один из первых, высокоуровневый, быстрый, вычислительный
- COBOL: для деловых задач
- PL/I: общего назначения, мощный и сложный, подходил для системного программирования
- Языки ассемблера для разных архитектур, не переносимые
- *Все перечисленные — не только языки ассемблера* — не очень-то переносимый
- Не структуррые языки, что ухудшало качество кода — «спагетти-код» тяжело сопровождать

Появление C

- Компилируемый
- Структурный
- Хорош в системном программировании
- Достаточно простой
- Переносимый

Что такое структурное программирование?

Программа состоит из:

- Блоков с последовательностью операторов
- Циклов
- Ветвления (`if` — `else` — ...)
- Перечисленное можно комбинировать

Что такое структурное программирование?

Программа состоит из:

- Блоков с последовательностью операторов
- Циклов
- Ветвления (`if` — `else` — ...)
- Перечисленное можно комбинировать

Теорема Böhm-Jecopini: перечисленное достаточно для того, чтобы выразить любой алгоритм в смысле полноты по Тюрингу

Что такое структурное программирование?

Программа состоит из:

- Блоков с последовательностью операторов
- Циклов
- Ветвления (`if` — `else` — ...)
- Перечисленное можно комбинировать

Теорема Böhm-Jecopini: перечисленное достаточно для того, чтобы выразить любой алгоритм в смысле полноты по Тюрингу

Вдобавок:

- `goto` есть, но не приветствуется
- Процедуры (функции)!
- Лексическая область видимости переменных (а не как в Basic или Python!)

Что ещё глянуть?

- Notes on Structured Programming. By Prof. Dr. Edsger W. Dijkstra — T. H. Report 70-WSK-03 Second Edition April 1970
- Dijkstra: EWD 215: A Case against the GO TO Statement (PDF).

Предки C

- 1960: Algol-60
- 1963: CPL
- 1967: BCPL
- 1969: B
- 1972: C

C и Unix

C and Unix evolved together

- C был переносим (не пользовался специфической функциональностью разных архитектур)
- C был достаточно прост, чтобы быстро реализовывать порождение кода для новых архитектур
- C хорошо подходил для системного программирования

C and Unix evolved together

- C был переносим (не пользовался специфической функциональностью разных архитектур)
- C был достаточно прост, чтобы быстро реализовывать порождение кода для новых архитектур
- C хорошо подходил для системного программирования

Изначально Unix написан на языке ассемблера, но в начале 1970-х его в значительной степени переписали на C, и это до сих пор одна их причин его популярности! Сейчас Unix-подобные системы на серверах, сетевом оборудовании, ПК, мобильных устройствах и т.д.

Актуальная история

1980s–1990s

- Дешёвые ПК
- Интернет

1990s–2000s–н.в.

- Много мобильных и встроенных архитектур
- Параллельные архитектуры

Вопросы

EDU.DLUCIV.NAME