|  |
| --- |
| РГПУ им. Герцена, ИКНиТО, ИВТ1 |
| История языков программирования: СИ |
| Часть 1 |

|  |
| --- |
| Иванов Дмитрий Владимирович  05.11.2017 |

Оглавление

[Введение 2](#_Toc498435653)

[Unix 3](#_Toc498435654)

[Прародители Си 4](#_Toc498435655)

[Как создавался Си 6](#_Toc498435656)

# Введение

Популярность языка программирования Си трудно переоценить, особенно вспоминая его былые заслуги. Наверное, каждый разработчик, как минимум, знает о его существовании, и, как максимум, пробовал на нем программировать. Си является предшественником таких языков, как:

* C++,
* Objective-C,
* C#,
* Java.

Компания [Microsoft](https://ru.wikipedia.org/wiki/Microsoft) для разработки родного языка к своей платформе .Net [[1]](#footnote-1)выбрала именно Си-подобный синтаксис. Более того, на Си написано множество операционных систем.

Конечно, Си не идеален: создатели языка – Кен Томпсон и Деннис Ритчи – долгое время дорабатывали его. Стандартизация Си продолжается до сих пор. Он существует более 45 лет и активно используется.

С ним часто ассоциируют не один, а два языка программирования — C/C++. Однако ниже речь пойдет именно о «чистом» Си.

 Язык Си восходит корнями к языку ALGOL (расшифровывается как ALGorithmic Language), который был создан в 1958 году совместно с комитетом Европейских и Американских учёных в сфере компьютерных наук на встрече в Швейцарской высшей технической школе Цюриха. Язык был ответом на некоторые недостатки языка FORTRAN[[2]](#footnote-2) и попыткой их исправить. Кроме того, разработка Си тесно связана с созданием операционной системы UNIX, над которой также работали Кен Томпсон и Деннис Ритчи.

**Деннис Ритчи**

# Unix

Проект МАС (Multiple Access Computer, Machine-Aided Cognition, Man and Computer) начался как чисто исследовательский в MIT в 1963 году.  
В рамках проекта МАС была разработана операционная система CTSS (Compatible Time-Sharing System). Во второй половине 60-х было создано несколько других систем с разделением времени, например:

1. BBN,
2. DTSS,
3. JOSS,
4. SDC
5. Multiplexed Information and Computing Service (MULTICS).

Multics – совместная разработка MIT, Bell Telephone Laboratories (BTL) и General Electric (GE) по созданию ОС с разделением времени для компьютера GE-645. Последний компьютер под управлением Multics выключили 31 октября 2000 года.

Однако BTL отошел от этого проекта еще в начале 1969 года. Некоторые его сотрудники (Кен Томпсон, Деннис Ритчи, Стью Фельдман, Дуг МакИлрой, Боб Моррис, Джо Оссанна) захотели продолжить работу самостоятельно. Томпсон работал над игрой Space Travel на GE-635. Ее написали сначала для Multics, а потом переписали на Фортране под GECOS на GE-635. Игра моделировала тела Солнечной системы, а игроку надо было посадить корабль куда-нибудь на планету или спутник.

Ни софт, ни железо этого компьютера не годились для такой игры. Томпсон искал альтернативу, и переписал игру под бесхозный PDP-7. Память была объемом 8К 18-битных слов, и еще был процессор векторного дисплея для вывода красивой для того времени графики.

Томпсон и Ритчи полностью вели разработку на кросс-ассемблере на GE и переносили код на перфолентах. Томпсону это активно не нравилось, и он начал писать ОС для PDP-7, начиная с файловой системы. Так появилась UNIX.

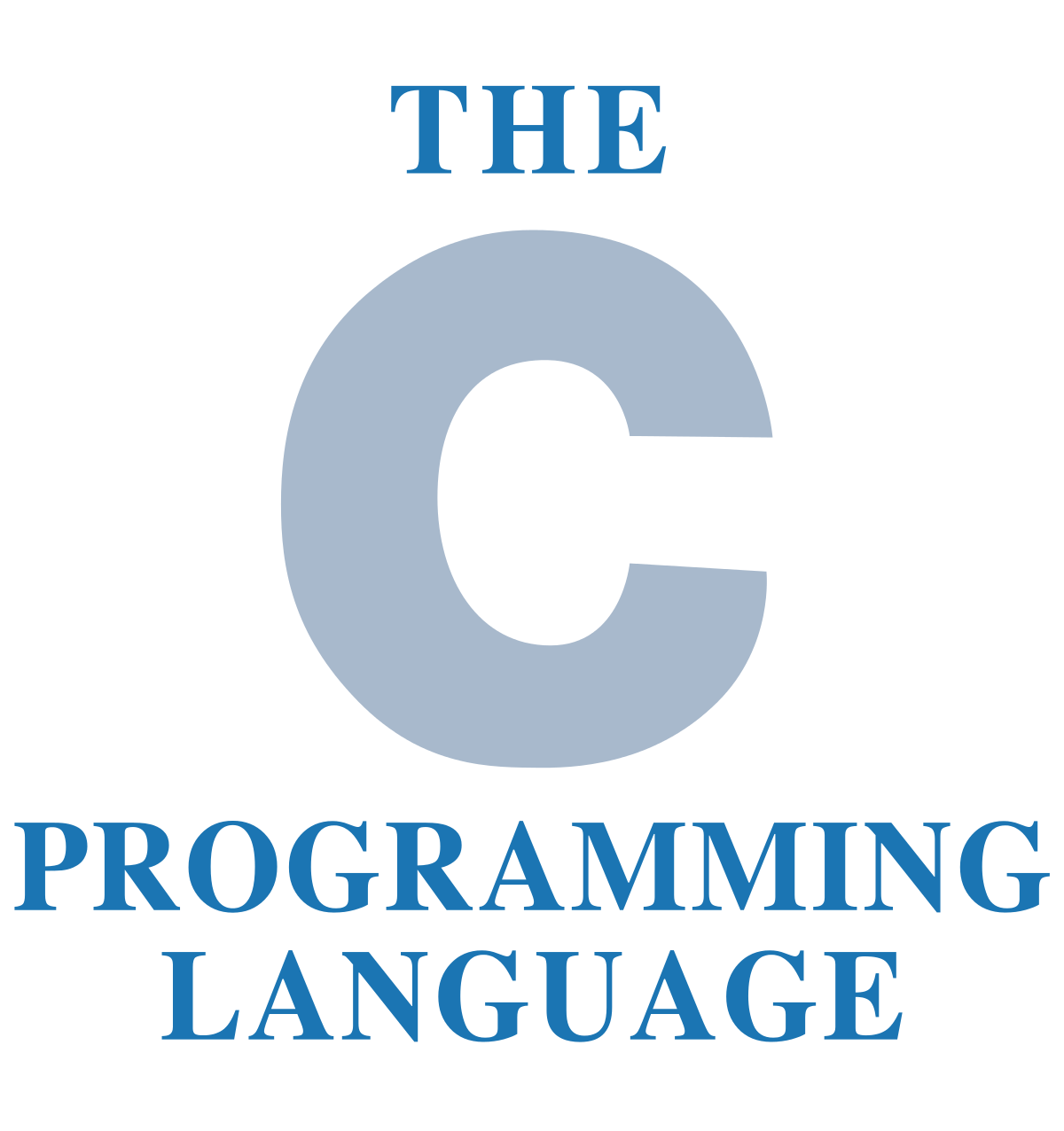
Томпсон хотел создать комфортабельное вычислительное окружение, сконструированное в соответствии с его дизайном, используя любые доступные средства. Его замыслы, что очевидно оглядываясь назад, вбирали в себя многие инновации Multics, включая понятие процесса как основы управления, древовидную файловую систему, интерпретатор команд в качестве пользовательской программы, упрощённое представление текстовых файлов и обобщённый доступ к устройствам.

PDP-7 UNIX также положил начало высокоуровневому языку B, который создавался под влиянием языка BCPL. Деннис Ритчи сказал, что В — это Си без типов. BCPL помещался в 8 Кб памяти и был тщательно переработан Томпсоном. В постепенно вырос в С.

К 1973 году язык Си стал достаточно силён, и большая часть ядра UNIX, первоначально написанная на ассемблере PDP-11/20, была переписана на Си. Это было одно из самых первых ядер операционных систем, написанное на языке, отличном от ассемблера. Получается, что Си – это «сопутствующий продукт», полученный во время создания операционной системы UNIX.

# 

# Прародители Си



Вдохновлённые языком ALGOL-60, Математическая лаборатория Кембриджского Университета совместно с Компьютерным отделом Лондонского университета создали в 1963 году язык CPL (Combined Programming Language). Язык CPL посчитали сложным, и в ответ на это Мартином Ричардсоном был создан в 1966 году язык BCPL, основное предназначение которого заключалось в написании компиляторов. Сейчас он практически не используется, но в своё время из-за хорошей портируемости он играл важную роль.

BCPL использовался в начале 1970-х в нескольких интересных проектах, в числе которых — операционная система OS6 и частично в зарождающихся разработках Xerox PARC.

BCPL послужил предком для языка Би (B), разработанного в 1969 в уже знакомой всем AT&T Bell Telephone Laboratories, не менее знакомыми Кеном Томпсоном и Деннисом Ритчи.

Как и остальные операционные системы того времени, UNIX был написан на ассемблере. Отладка программ на ассемблере настоящая мука. Томпсон решил, что для дальнейшей разработки ОС необходим язык высокого уровня и придумал небольшой язык B. За основу Томпсон взял язык BCPL. Язык B можно рассматривать как C без типов.

Во многих [деталях](http://givi.olnd.ru/chist/) BCPL, B и C различаются синтаксически, но в основном они похожи. Программы состоят из последовательности глобальных деклараций и объявлений функций (процедур). В BCPL процедуры могут быть вложенными, но не могут ссылаться на нестатические объекты, определённые в содержащих их процедурах. B и C избегают такого ограничения, вводя более строгое: вложенных процедур нет вообще. Каждый из языков (за исключением самых древних версий B) поддерживает раздельную компиляцию и предоставляет средства для включения текста из именованных файлов.

В противоположность повсеместному изменению синтаксиса, которое происходило во время создания B, основная семантика BCPL — его структура типов и правила вычисления выражений — осталась нетронутой. Оба языка — безтиповые, вернее имеют единственный тип данных — «слово» или «ячейка», набор битов фиксированной длины. Память в этих языках — массив таких ячеек, а смысл содержимого ячейки зависит от операции, которая к ней применяется. Например, оператор "+" просто складывает свои операнды при помощи машинной инструкции add, и другие арифметические операции также безразличны к смыслу своих операндов.

Ни BCPL, ни B, ни C не выделяют в языке символьные данные; они считают строки векторами целых чисел и дополняют общие правила несколькими соглашениями. И в BCPL, и в B строковый литерал означает адрес статической области инициализированный символами строки упакованными в ячейки.

# Как создавался Си

В 1970 Bell Labs приобрела для проекта компьютер PDP-11. Так как B был готов к работе на PDP-11, Томпсон переписал часть UNIX на B.

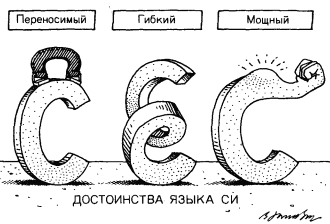
Но модель B и BCPL подразумевала издержки при работе с указателями: правила языка, определяя указатель как индекс в массиве слов, делали указатели индексами слов. Каждое обращение к указателю при исполнении генерировало масштабирование указателя в адрес байта, который ожидал процессор.

Поэтому становилось ясно, что для того, чтобы справиться с символами и байтовой адресацией, а также подготовиться к грядущей аппаратной поддержке вычислений с плавающей точкой, нужна типизация.

В 1971 году Ритчи начал создавать расширенную версию B. Сначала он назвал её NB (New B), но когда язык стал сильно отличаться от B, название сменили на C. Вот что, [писал](http://givi.olnd.ru/chist/#embryonic_c) об этом сам Ритчи: “Я хотел, чтобы структура не только характеризовала абстрактный объект, но и описывала набор бит, который мог быть прочитан из каталога. Где компилятор смог бы спрятать указатель, на name, которого требует семантика? Даже если бы структуры были бы задуманы более абстрактными, и место для указателей могло бы быть спрятано где-нибудь, как бы я решил техническую проблему корректной инициализации этих указателей при выделении памяти для сложного объекта, возможно структуры содержащей массивы, которые содержат структуры, и так до произвольной глубины?”

Решение состояло в решительном скачке в эволюционной цепочке между безтиповым BCPL и типизированным C. Он исключал материализацию указателя в хранилище, а вместо этого порождал его создание, когда имя массива упоминалось в выражении. Правило, которое сохранилось и в сегодняшнем C, состоит в том, что значения–массивы, когда они упоминаются в выражении, конвертируются в указатели на первый из объектов, составляющих этот массив.

Второе нововведение, которое наиболее ясно отличает C от его предшественников, — вот эта более полная структура типов и особенно её выразительность в синтаксисе деклараций. NB предлагал основные типы int и char совместно с массивами из них и указателями на них, но никаких других способов скомпоновать их.

Требовалось обобщение: для объекта любого типа должно быть возможным описать новый объект, который объединяет несколько таких объектов в массив, получает его из функции или является указателем на него.  
  
Для любого объекта такого составного типа, уже был способ указать на объект, который является его частью: индексировать массив, вызвать функцию, использовать с указателем оператор косвенного обращения. Аналогичное рассуждение приводило к синтаксису объявления имён, который отражает синтаксис выражения, где эти имена используются.

[Изображение](https://habrastorage.org/getpro/habr/post_images/2b2/5c3/4d0/2b25c34d0f1bc0ace16e20af4876adfb.gif) из книги «Язык Си»: M. Уэйт, С. Прата, Д. Мартин

Формула нахождения площади круга:

|  |  |  |
| --- | --- | --- |
| Июнь 2016 | Июнь 2015 | Язык программирования |
| 1 | 1 | Java |
| 2 | 2 | Си |
| 3 | 3 | С++ |
| 4 | 6 | Python |
| 5 | 4 | C# |

Первые пять позиций индекса TIOBE, который измеряет рост популярности языков программирования

1. .NET Framework — [программная платформа](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0#кроссплатформенное_программное_обеспечение), выпущенная компанией [Microsoft](https://ru.wikipedia.org/wiki/Microsoft" \o "Microsoft) в [2002 году](https://ru.wikipedia.org/wiki/2002_%D0%B3%D0%BE%D0%B4). Основой платформы является общеязыковая среда исполнения [Common Language Runtime (CLR)](https://ru.wikipedia.org/wiki/Common_Language_Runtime" \o "Common Language Runtime), которая подходит для разных языков программирования. Функциональные возможности CLR доступны в любых языках программирования, использующих эту среду. [↑](#footnote-ref-1)
2. FORTRAN — первый [язык программирования высокого уровня](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%B2%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D0%B3%D0%BE_%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D1%8F), получивший практическое применение, имеющий [транслятор](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D1%81%D0%BB%D1%8F%D1%82%D0%BE%D1%80) и испытавший дальнейшее развитие. Создан в период с 1954 по 1957 год группой программистов под руководством [Джона Бэкуса](https://ru.wikipedia.org/wiki/%D0%91%D1%8D%D0%BA%D1%83%D1%81,_%D0%94%D0%B6%D0%BE%D0%BD) в корпорации [IBM](https://ru.wikipedia.org/wiki/IBM). [↑](#footnote-ref-2)