

Обзор основных языков программирования.

1. Основные языки программирования.

1957-1959	Fortran, LISP, COBOL	Авторы	Назначение	Использовался
	Старейшие языки программирования. Высокоуровневые, созданы для научных, математических и бизнес вычислений.	Джон Бекус Джон Маккарти Грейс Хоппер («бабушка Кобола»)	Разработка ПО для научных и инженерных вычислений, для обработки списков, бизнеса.	В NASA, кредитных картах и банкоматах
1970	Pascal	Автор	Назначение	Использовался
	Высокоуровневый для обучения структурному программированию и структурированию данных.	Никлаус Вирт	Обучение программированию	Skype
1972	C	Автор	Назначение	Использовался
	Основан на языке «В». Низкоуровневый, общего назначения. Его синтаксис стал основой для C#, Java, Perl, PHP, Python и др.	Деннис Ритчи	Кроссплатформенное программирование, системное программирование, программирование для UNIX, разработка игр	UNIX (первые веб-серверы и веб-клиенты)
1983	C++	Автор	Назначение	Использовался
	Первоначальное название «Си с классами» («++» – оператор инкремента в C). Среднеуровневый, объектно-ориентированный.	Бьёрн Страуструп	Коммерческая разработка приложений, встроенного ПО, клиент-серверных приложений, видеоигр	Adobe, Google, Chrome, Mozilla, Firefox, IE
1983	Objective-C	Автор	Назначение	Использовался
	Объектно-ориентированное расширение C. Высокоуровневый, общего назначения	Брэд Кокс и Том Лав	Программирование в Apple	Apple OS X и iOS

1987	Perl	Автор	Назначение	Использовался
	Высокоуровневый, общего назначения. Для обработки отчетов в UNIX	Ларри Уолл	Разработка общих шлюзовых интерфейсов, приложений для баз данных, систем администрирования, интернет-программирования, визуального программирования.	IMDb, Amazon, Priceline, Ticketmaster
1991	Python	Автор	Назначение	Использовался
	Высокоуровневый, общего назначения. Создан для поддержки различных стилей программирования.	Гвидо ван Россум	Веб-приложения, разработка ПО, защита информации.	Google, Yahoo, Spotify
1993	Ruby	Автор	Назначение	Использовался
	Высокоуровневый, общего назначения. Язык с простым синтаксисом, влияние на который оказали Perl, LISP, Smalltalk.	Юкиhiro Мацумото	Разработка веб-приложений	Twitter, Hulu, Groupon
1995	Java	Автор	Назначение	Использовался
	Высокоуровневый, общего назначения. Был создан для интерактивного ТВ-проекта. Кроссплатформенный.	Джеймс Гослиг	Веб-программирование, разработка веб-приложений и ПО, графического интерфейса пользователя. Разработка нативных и встраиваемых приложений	В системе и приложениях Android
1995	PHP	Автор	Назначение	Использовался
	Общего назначения, с открытым исходным кодом. Для создания динамических веб-страниц.	Расмус Лердорф	Создание и поддержка динамических веб-страниц, разработка на стороне веб-сервера.	Facebook, Вконтакте, Википедии, Digg, WordPress, Joomla
1995	JavaScript	Автор	Назначение	Использовался
	Высокоуровневый язык. Создан для расширения функциональности веб-страниц.	Брендан Эйх	Динамическая веб-разработка, использование в браузерах, PDF-документах,	Gmail, Adobe, Photoshop, Mozilla, Firefox

2. Хронология современных языков программирования

Год	Название	Разработчики, компания	Предшественник(и)
2000	C#	Андерс Хейлсберг, Microsoft (ECMA)	Cи, C++, Java, Delphi
2001	Visual Basic .NET	Microsoft	Visual Basic
2002	Скретч визуальная событийно-ориентированная среда программирования для обучения	команда программистов Массачусетского технологического института	Logo, Smalltalk, Squeak, E-Toys, HyperCard, AgentSheets, StarLogo, Tweak, BYOB
2003	Factor динамически типизированный конкатенативный язык программирования (стековый язык программирования)	Слава Пестов	Joy, Forth, Лисп
2003	Scala функциональный и объектно-ориентированный ЯП	Мартин Одерский	Smalltalk, Java, Haskell, Standard ML, OCaml
2005	F# поддерживает функциональное, императивное (процедурное) и объектно-ориентированное программирование	Дон Сайм, Microsoft Research	Objective Caml, C#, Haskell
2009	Go компилируемый многопоточный язык программирования	Google	C, Oberon, Limbo
2009	CoffeeScript Надстройка над JavaScript для написания серверных и приложений, работающих в браузере поверх Node.js добавляет синтаксический сахар в духе Ruby, Python, Haskell и Erlang	Джереми Ашкенас	JavaScript, Ruby, Python
2010	Chapel Каскадный высокопроизводительный язык с поддержкой распараллеливания	Brad Chamberlain, Cray Inc.	HPF, ZPL

2010	Rust компилируемый ЯП общего назначения с поддержкой парадигм функционального и процедурного программирования, параллелизма пригоден для системного программирования Mozilla, Dropbox	Грэйдон Хор, Mozilla Research	Alef, C++, Camlp4, Common Lisp, Erlang, Haskell, Hermes, Limbo, Napier, Napier88, Newsqueak, NIL, Sather, OCaml, Standard ML, Cyclone, Scheme
2011	Elm функциональный язык для декларативного создания графических веб-интерфейсов, используя функционально- реактивный стиль программирования	Evan Czaplicki	Haskell, Standard ML, OCaml, F#
2011	Kotlin объектно-ориентированный язык	JetBrains	Java, Scala, Groovy, C#, Gosu
2012	TypeScript является надстройкой над JavaScript отличается явным статическим определением типов, поддержка классов	Андерс Хейлсберг, Microsoft	JavaScript
2014	Swift Компилируемый объектно-ориентированный язык для разработчиков iOS и macOS	Apple	C, Objective-C
2015	Perl 6 Компилятор Perl 6 преобразует текст, написанный на языке Perl 6, в байт-код, который в дальнейшем исполняется на виртуальной машине. Такой же подход применяется в технологиях Java и .NET Framework.	Ларри Уолл	Haskell, JavaScript, Perl 5, Ruby, Smalltalk
2019	Bosque язык с открытым исходным кодом. Цель - повышение качества ПО и повышение производительности труда разработчиков.	Марк Мэппон, Microsoft	NodeJS/ JavaScript, TypeScript, ML

2021	Microsoft Power Fx универсальный, декларативный и функциональный язык программирования со строгой типизацией	Виджей Миталь, Робин Абрахам, Шон Катценбергер, Дэрил Рубин, Microsoft	Pascal, Mathematica, Miranda.
2022	Carbon , Google Carbon поддерживает базовую переносимость с C++, код на Carbon может быть интегрирован в код C++	Чендлер Каррут и команда разработчиков Google	преемник C++

3. Новые языки программирования

а) Язык Bosque — новый язык программирования от Microsoft

В середине апреля 2019 года Microsoft представила новый язык программирования, который получил название Bosque (разработчик Марк Баррон (Mark Barron)). Он распространяется с открытым исходным кодом и предназначен для того, чтобы написанный код был простым и понятным как для человека, так и для компьютера.

Новый язык, чьё название с испанского переводится как «лес», призван быть как можно более простым для понимания и помочь избежать сложностей при разработке и написании кода. Однако отмечается, что язык экспериментальный и пока не готов к широкому использованию.

Автор описывает этот язык как попытку выйти за рамки модели структурного программирования, ставшей популярной в 1970-х. Парадигма структурного программирования, в которой управление потоком выполнения осуществляется с помощью циклов, условных операторов и подпрограмм, стала популярной после публикации в 1968 году статьи компьютерного учёного Эдсгера Дейкстры «Go To Statement Considered Harmful». Маррон считает, что мы можем добиться большего, избавившись от таких источников сложности, как *циклы, изменяемое состояние и ссылочное равенство*. Результатом раскрытия этой идеи Маррона и является Bosque, представляющий парадигму программирования, которую Маррон в своей статье назвал «*регуляризованным программированием*». Спецификация Bosque, синтаксический анализатор, средство проверки типов, эталонный интерпретатор и поддержка IDE выпущены под лицензией MIT и доступны на GitHub.

В основу **Bosque** легли типы и синтаксис **TypeScript**, а семантика позаимствована из **ML** и **Node/JavaScript**.

Главная миссия дизайна языка — чтобы он был прост и понятен как для человека, так и для компьютера. Подробно об особенностях нового языка программирования можно прочитать в [документации](#) Microsoft.

1) Все значения в Bosque являются *неизменяемыми* (immutable).

Но при этом можно объявить изменяемую переменную ключевым словом **var**!

2) В языке нет циклов *for*, *while* и т.д. Вместо этого есть коллекции и конвейеры (пайплайны). Другими словами, вместо циклов нужно использовать *map*, *filter* и т.д. Используются функциональные объекты (*Functors*), которые выполняют роль циклов и могут повысить качество работы ПО.

3) Строки можно делать разных типов. Т.е., например, можно сделать строку-имя или строку-zipcode, и для *type*-чекера это будут две разные строки. Если вы в аргументе функции ожидаете *zipcode*, а вам по ошибке туда положат имя, то компилятор это не проглотит. Синтаксис такой: `String[Zipcode]`.

4) Вызов функций можно делать с указанием названия аргументов из сигнатуры функции, например: `myfunc (x=1, y=2)`

5) В стандартной библиотеке есть различные коллекции, и с коллекциями можно работать по-разному. Можно просто по цепочке вызывать *map*, потом *filter* и т.д., а можно работать через конвейеры.

Примеры

Сложение двух чисел

```
function add2(x: Int, y: Int): Int {  
    return x + y;  
}
```

```
add2(2, 3)           //5  
add2(x=2, y=3)       //5  
add2(y=2, 5)         //7
```

b) Microsoft представила язык программирования Power Fx

Power Fx — это язык с малым объемом кода, который будет использоваться во всей платформе **Microsoft Power Platform**. Это универсальный, декларативный и функциональный язык программирования со строгой типизацией, основан на синтаксисе функций **Excel**.

Это новое название языка формул для приложений на основе холста в Power Apps (платформа разработки бизнес-приложений) - это набор приложений, служб и соединителей, а также платформа данных, которая предоставляет среду разработки для эффективного создания пользовательских приложений для бизнеса.

Девиз языка — «Вы можете создать приложение так же легко, как электронную таблицу». Суть применения low-code как раз состоит в том, чтобы снизить порог входа до уровня продвинутого пользователя Excel.

Power Fx будет доступен как программное обеспечение с открытым исходным кодом. В настоящее время он интегрирован в приложения на основе холста.

Программирование в стиле электронной таблицы.

Например, `m * a` в большинстве языков означает умножение `m` и `a`. Результат выражения может быть помещен в некоторую переменную, использован в качестве аргумента процедуры/функции или вложен в большее выражение.

В **Power Fx** выражение описывает вычисление, которое связывает выражение с идентификатором и `m` или `a` автоматически обновляется до нового значения. Вот почему язык называют **Power Fx языком формул**. Изменения происходят всегда в реальном времени (реактивное программирование).



c) Язык Carbon — новый язык программирования от Google

В июле 2022 года инженер Google Чендлер Каррут впервые представил язык **Carbon** — экспериментальный язык программирования общего назначения, созданный компанией Google, как «*преемник* C++». Презентация прошла на конференции Cpp North в Торонто (Канада). Чендлер Каррут называет Carbon не заменой, но преемником C++ (разработанный в 1982 году и выпущенный в 1985 году).

Программисты на C++, желающие полностью перейти на Carbon, получают в свое распоряжение инструментарий для автоматической транслитерации библиотек C++ в код на новом языке Google. Обратная миграция тоже возможна – в дальнейшем эти библиотеки могут использоваться в существующем проекте на C++.

Все необходимые разработчику инструменты Carbon размещены на принадлежащем Microsoft портале GitHub и распространяются по лицензии Apache 2.0. Компилятор кода Carbon написан при помощи LLVM (Low Level Virtual Machine) – специальной программной инфраструктуры для создания компиляторов. Также в нем использовались наработки из Clang – компилятора для C, C++, Objective-C и Objective-C++.

Программа «Hello, World!», написанная на языке Carbon:

```
package Sample api;

fn Main() -> i32 {
    Print("Hello, World!");
    return 0;
}
```

В настоящее время Carbon находится на экспериментальной стадии. Дорожная карта:

Выпуск основной рабочей версии 0.1 к концу 2022 г.

версии 0.2 в 2023 г.

Релиз 1.0 в 2024–2025 гг.

```
// C++:
#include <math.h>
#include <iostream>
#include <span>
#include <vector>

struct Circle {
    float r;
};

void PrintTotalArea(std::span<Circle> circles) {
    float area = 0;
    for (const Circle& c : circles) {
        area += M_PI * c.r * c.r;
    }
    std::cout << "Total area:" << area << "\n";
}

auto main(int argc, char** argv) -> int {
    std::vector<Circle> circles = {{1.0}, {2.0}};
    // Implicitly converts `vector` to `span`.
    PrintTotalArea(circles);
    return 0;
}
```

```
// Carbon:
package Geometry api;
import Math;

class Circle {
    var r: f32;
};

fn PrintTotalArea(circles: Slice(Circle)) {
    var area: f32 = 0;
    for (c: Circle in circles) {
        area += Math.Pi * c.r * c.r;
    }
    Print("Total area: {0}", area);
}

fn Main() -> i32 {
    // A dynamically sized array, like `std::vector`.
    var circles: Array(Circle) = ({.r = 1.0},
                                   {.r = 2.0});
    // Implicitly converts `Array` to `Slice`.
    PrintTotalArea(circles);
    return 0;
}
```

«Конечно, синтаксис Carbon будет проще, чем синтаксис C++. Но я не считаю это важным. Для меня любой синтаксис нормальный, потому что при изучении языка программирования синтаксис я осваиваю в последнюю очередь. Моя позиция такова: если вы решили изучить язык программирования, то синтаксис – последнее, что вы должны учить. Сначала освоите философию. Потому что синтаксис – это как писать на русском или на английском. Вы уже умеете буквы писать? Значит, научитесь. Здесь то же самое.

Клавиши нажимать все умеют, разберитесь лучше сначала с системой типов в языке», – мнение эксперта

d) Mojo: убийца Python и будущее AI (англ. artificial intelligence)

Mojo – специализированный язык программирования, ориентированный на разработку в сфере AI. Он был представлен 2 мая 2023 года компанией Modular. В этом проекте участвует большое количество гуру специалистов ИИ, главные из них:

- Крис Лэттнер – сооснователь и директор Modular (в прошлом один из ключевых разработчиков языка Swift, компилятора Clang, а также технологий LLVM и MLIR, работал в Google, Tesla и Apple);
- Тим Дэвис – сооснователь и руководитель продукта, внёс большой вклад в инфраструктуру искусственного интеллекта Google в Google Brain и Core Systems.

На данный момент Mojo поддерживается на Ubuntu Linux и macOS, но вскоре обещают добавить поддержку и на Windows. Более подробно с требованиями вы можете ознакомиться на [официальном сайте Mojo](#).

Разрабатывая системы ИИ на Python, инженеры, так или иначе, подключают модули, написанные на более производительных языках. Такой подход усложняет отладку, обучение и развертывание приложений. Mojo использует MLIR (Multi-Level Intermediate Representation), фреймворк для разработки компиляторов, и значительно обходит Python в вычислительной скорости. Например, для выполнения алгоритма Мандельброта первому нужно 0,03 секунды, а второму – 17 минут.

У Mojo есть несколько целей:

- работать, имея полную совместимость с экосистемой Python;
- дать разработчикам возможность развертывать код в ускорителях;
- низкоуровневый контроль для обеспечения прогнозируемой производительности;
- обеспечить отсутствие фрагментации экосистемы.

По сути Mojo – это язык программирования, который поддерживает метапрограммирование на этапе компиляции. Кроме того, он поддерживает такие функции, как кэширование во время компиляции, методы адаптивной компиляции и т.д. Другие языки программирования не обладают такими функциями.

Достоинства Mojo: совместимость с библиотеками Python, встроенная автонастройка, которая самостоятельно подбирает значения для параметров исходя из имеющегося оборудования, а также функцию *struct*, которая упорядочивает атрибуты в памяти для использования в структурах данных.

Mojo может стать наиболее важным достижением в программировании за последнее время.

Mojo – новый язык, и он выглядит довольно многообещающим. Он решает некоторые проблемы, возникающие у разработчиков в области ИИ. Но, сегодня уже существуют другие решения, которые также могут повысить скорость Python, например, Jax, Codon и Julia – язык, ориентированный на исследование данных.

Таким образом, может случить одно из двух. Первое: количество функций может значительно вырасти, и сообщество примет Mojo. Второе: он станет языком программирования узкой направленности, который будет использовать библиотеки Python. Заменит ли Mojo язык программирования Python, покажет лишь время.

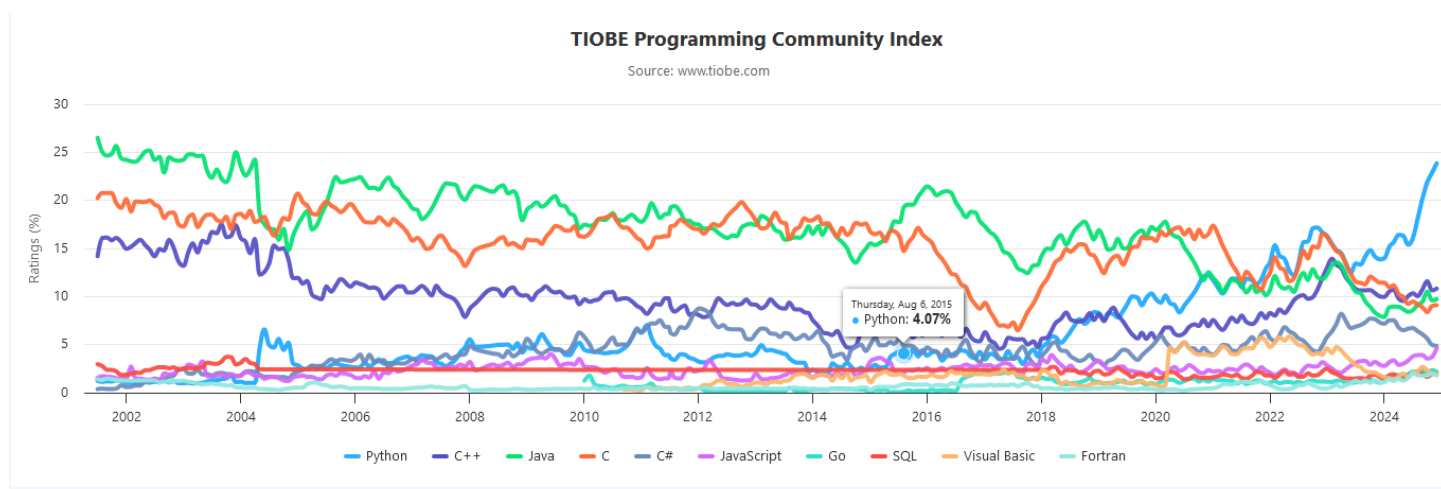
4. TIOBE определила кандидатов на «язык года»

(TIOBE Index for December 2024)













Индекс TIOBE – индекс, который оценивает популярность языков программирования, основываясь на количестве поисковых запросов, содержащих название языка. Расчет индекса происходит ежемесячно.

Каждый год, начиная с 2003, авторами TIOBE выбирается язык года (Programming Language of the Year):

- 2023 C#
- 2022 [C++](#)
- 2021 [Python](#)
- 2020 [Python](#)
- 2019 [C](#)
- 2018 [Python](#)
- 2017 [C](#)
- 2016 [Go](#)
- 2015 [Java](#)
- 2014 [Javascript](#)
- 2013 [Transact-SQL](#)
- 2012 [Objective-C](#)
- 2011 [Objective-C](#)
- 2010 [Python](#)
- 2009 [Go](#)
- 2008 [C](#)
- 2007 [Python](#)
- 2006 [Ruby](#)
- 2005 [Java](#)
- 2004 [PHP](#)
- 2003 [C++](#)



Четвёрка лидеров осталась неизменной: **Python, C++, Java и C.**

Dec 2024	Dec 2023	Change	Programming Language		Ratings	Change
1	1			Python	23.84%	+9.98%
2	3	▲		C++	10.82%	+0.81%
3	4	▲		Java	9.72%	+1.73%
4	2	▼		C	9.10%	-2.34%
5	5			C#	4.87%	-2.43%
6	6			JavaScript	4.61%	+1.72%
7	13	▲▲		Go	2.17%	+1.14%
8	9	▲		SQL	1.99%	+0.37%
9	8	▼		Visual Basic	1.96%	+0.14%
10	12	▲		Fortran	1.79%	+0.72%
11	16	▲▲		Delphi/Object Pascal	1.44%	+0.52%
12	7	▼▼		PHP	1.39%	-0.62%
13	11	▼		Scratch	1.33%	+0.26%
14	18	▲▲		Rust	1.29%	+0.48%
15	14	▼		MATLAB	1.09%	+0.16%
16	20	▲▲		R	1.05%	+0.33%
17	10	▼▼		Assembly language	1.04%	-0.07%
18	19	▲		Ruby	1.03%	+0.26%
19	23	▲▲		COBOL	0.98%	+0.30%
20	17	▼		Swift	0.98%	+0.16%