Компьютерные инструменты в образовании, 2024
№ 1: 85–95
УДК: 004.43
http://cte.eltech.ru
doi:10.32603/2071-2340-2024-1-85-95

практика программирования

ЯЗЫК PYTHON В ВЫСШЕМ ОБРАЗОВАНИИ

Павлов Д. А.[1], канд. физ.-мат. наук, доцент, ✉ dapavlov@etu.ru

[1]Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина), ул. Профессора Попова, 5, корп. 3, 197022, Санкт-Петербург, Россия

Аннотация

В статье приводятся оценки популярности языков программирования в различных сферах и уделяется особенное внимание языку Питон, набравшему популярность в образовательной среде. Приводится перечень широко известных (и не свойственных другим языкам) недостатков Питона как языка и экосистемы. Полезность Питона в карьере признаётся преувеличенной, а в образовательном процессе — скорее отрицательной.

Ключевые слова: *Python, программирование, образование.*

Цитирование: Павлов Д. А. Язык Python в высшем образовании // Компьютерные инструменты в образовании. 2024. № 1. С. 85–95. doi:10.32603/2071-2340-2024-1-85-95

1. ВВЕДЕНИЕ

Наступил 2024. Питон — самый популярный язык в мире по рейтингу TIOBE [1]. Самый известный программный продукт ушедшего года — ChatGPT — написан на Питоне. Питон легко выучить, и программисты на Питоне делают головокружительную карьеру в разработке. Питон — универсальный язык, на котором можно писать любые программы.

Эти факты могут создать ложное впечатление, что новые учебные курсы по программированию должны быть посвящены Питону и все новые программы нужно также писать на Питоне.

Между тем указанные факты никаких подобных следствий не имеют, для выбора Питона для написания программ нет оснований (за исключением узкой ниши машинного обучения), а изучение Питона может, скорее, навредить (в особенности в качестве первого языка программирования). Разберём популярные заблуждения в деталях.

Все нижеизложенные цифры — свежие (даны по состоянию на 2023) и вряд ли заметно изменятся в пользу Питона в будущем, так как язык существует более 32 лет, он старше своих нынешних конкурентов (кроме Bash, Tcl и C/C++) и свой потенциал взрывного роста давно исчерпал.

2. РЕЙТИНГИ

TIOBE — рейтинг, составляемый на основе статистики поисковых запросов. Никаких данных, свидетельствующих о связи этой статистики с необходимостью учить язык или

писать на нём программы, не известно.

В рейтингах, составленных на основании опросов пользователей сайта «Stackoverflow», Питон уступает JS (а среди профессионалов — и SQL). В целом ориентироваться на подобные рейтинги бесполезно, так как их показатели, во-первых, не имеют связи с реальными количественными или качественными показателями, применяемыми в образовании или индустрии, а во-вторых, меняются чаще, чем образовательные учреждения могут менять свои учебные программы. (В вышеупомянутом TIOBE Питон лидировал в 2010, но опустился на 8-е место в 2013).

По количеству открытых репозиториев на Github Питон занимает второе место (после JS и перед Typescript). В исходных кодах пакетов современного дистрибутива Debian Bookworm [3] Питон находится на 3-м месте по общему количеству строк кода (6,2 %), после C и C++. В целом, никаких признаков доминирования Питона в индустрии не обнаруживается.

3. ПРОСТОТА

Синтаксис Питона прост, что обеспечивает быстрое усвоение языка на начальном этапе. Питон замечательно подходит для написания небольших программ и обучения основным понятиям программирования (выражения, управляющие конструкции, ввод-вывод, простейшие структуры данных), что актуально при обучении программированию детей младших и средних классов школы. Но для больших проектов Питон никогда не предназначался. Сам язык, его инфраструктура, сообщество — всё ориентировано в большей степени на небольшие скрипты и быстрое прототипирование. По мере роста программы (и программиста) пропадает очарование простого вхождения в язык и начинают проявляться недостатки.

3.1. Медлительность исполнения

Интерпретатор CPython (основная используемая реализация языка) — одна из самых медленных реализаций скриптового языка на свете. Программы на Питоне исполняются медленнее, чем аналогичные им на Си, в разы и десятки раз. В некоторой степени эта проблема может быть решена выводом интенсивных вычислений через FFI во внешние библиотеки, реализованные на Си (например, как сделано в numpy). Но такая схема работает лишь до тех пор, пока в вычислениях нет ветвлений. Ветвления, выраженные в коде на Питоне, не могут быть помещены в numpy. Примером вычислительной задачи, решение которой содержит ветвления, является поиск числа в отсортированном массиве методом бисекций. В numpy реализован этот алгоритм в функции «searchsorted», но создать аналогичную функцию средствами numpy пользователь не может, да и searchsorted по каким-то причинам работает медленно [4, 5].

3.2. Несовместимости

Большая часть библиотек разрабатывается в отдельности от языка. При одновременном использовании нескольких библиотек возникают несовместимости между библиотеками, разрешимые в лучшем случае инсталлированием рекомендованных версий (для упрощения этого процесса в Питоне созданы виртуальные окружения, содержащие копии всех библиотек, нужных программе), а в худшем — правками со стороны пользователя после часов отладки и чтения форумов. Проблема усугубляется тем, что минимали-

стичный синтаксис языка поощряет скрытое взаимодействие библиотеки с интерпретированной моделью пользовательской программы. Протоколов для такого рода взаимодействий не разработано, из-за чего и происходят наиболее сложные поломки. Примеры: pytorch и pyinstaller [6], numba и scipy [7]. Проектам на C/C++/Java/C# и на многих других языках такие поломки не свойственны.

3.3. Отсутствие тредов на уровне ОС

Номинально треды в Питоне есть, но без параллельного выполнения байт-кода. Следовательно, параллельный режим доступен лишь для операций ввода-вывода и вызовов внешних библиотек (FFU). Все остальные операции выполняются последовательно на одном ядре CPU. Причиной этого является глобальная блокировка интерпретатора (GIL), созданная в CPython с первых дней его существования. Необходимости в GIL не было: например, в интерпретаторе Tcl она отсутствует. Тем более, нет этой проблемы в языках без интерпретатора: C/C++/C#/Java/Go. Параллельное выполнение в Питоне можно получить в режиме мультипроцессинга, то есть с помощью не тредов, а отдельных процессов. Такое решение приводит к увеличению расхода памяти (в каждом из процессов сидит целый независимый интерпретатор) и к необходимости обмена данными через разделяемую память (а не просто память процесса, как было бы при обычных тредах).

3.4. Разнообразие и изменчивость

Начинающий питонист быстро обнаруживает, что одну и ту же вещь можно сделать множеством способов, половина из которых работала раньше, но в настоящий момент сломана, а в другой половине невозможно найти общественный консенсус.

Так, притчей во языцех является установка библиотек. Их можно ставить в системную инсталляцию Питона, или завести локальную инсталляцию Питона и ставить библиотеки в неё, или инсталлировать через PIP в локальную директорию site-packages, или через PIP в виртуальное окружение. Само виртуальное окружение можно сделать с помощью venv или virtualenv (это разные вещи с похожими названиями); также существуют pyenv, pyenv-virtualenv, virtualenvwrapper, pyenv-virtualenvwrapper, pipenv, conda, anaconda и miniconda. Рано или поздно разработчику приходится узнавать об этих средствах и о том, в каких отношениях они находятся между собой. Ни о какой простоте разработки (равно как и о самой разработке) при этом не идёт и речи.

Разнообразие проявляется и в интерфейсах библиотек, которые постоянно меняются. В сообществе питонистов не принято заботиться об обратной совместимости версий. Программа, которая не обновлялась 5 лет, скорее всего не будет работать со свежими версиями пакетов, от которых она зависела 5 лет назад. Она также может не заработать и с новыми версиями библиотек (после внесения нужных изменений в код, зависящий от них), потому что в библиотеках появились новые ошибки. (Стандарты тестирования Питоновских библиотек бесконечно далеки от стандартов, применяемых, например, в SQLite [9], написанной, между прочим, на Си образца 1989 г.)

Принципы разработки библиотек вовсе не идут вразрез с принципами разработки самого Питона. В 2008 г. вышел Python 3, несовместимый с Python 2 на уровне не только библиотек, но и синтаксиса, что доставило проблем разработчикам в течение последующих 11 лет. Известна история компании Dropbox, которая начала миграцию с Python 2 на 3 в 2015 и закончила в 2018 [8]; руководил процессом лично создатель языка Гвидо ван

Россум, работавший в компании. Через год после миграции он ушёл на пенсию, а ещё через 2 месяца поддержка Питона 2 на python.org была официально прекращена.

Эта и другие истории о печальных последствиях нарушения обратной совместимости не научили сообщество разработчиков ничему. Работавшие ранее вещи в Питоне 3 целенаправленно удаляют или ломают. Так, в грядущем выпуске Python 3.13 будут удалены 20 модулей [10] — больше, чем было удалено из Java за всё время её существования [11]. Запущенный процесс избавления от GIL сломает множество программ, и процесс восстановления займёт много лет. Программы на Питоне будут устаревать и ломаться всегда, это одна из культурных особенностей языка. Программистам на Питоне нужно бежать, чтобы просто оставаться на месте.

В языках C, C++, Java, C# и многих других обратной совместимости уделяется огромное внимание. Аналогичных принципов придерживаются разработчики программ на этих языках. Новые возможности добавляются, но старые не удаляются. Если новые возможности несовместимы со старым API, заводится новая версия API, но старая продолжает поддерживаться. Ломать совместимость без веских причин *нельзя*.

Вывод: если писать на Питоне программы из 100 строк действительно просто, то большие программы писать сложно, и уж точно не проще, чем на других языках, и результат, как правило, работает менее стабильно и более требователен к ресурсам.

4. УНИВЕРСАЛЬНОСТЬ

На Питоне можно написать любую программу. На любом языке можно написать любую программу. Люди пишут тетрисы на sed [12], 3D-графику на Powershell [13] и TLS на Javascript [14], а также занимаются машинным обучением на Форте [15]. Всё это позволяет гордиться человечеством, но не имеет никакого отношения к практическим вопросам о том, какие языки нужно учить и на каких языках нужно писать.

Питон популярен в скриптах (для которых предназначался изначально), в веб-фреймворках (которые сами по себе выросли из скриптов), в ИИ (хорошо подошёл под используемую модель вычислений) и в науке (что неудивительно, зная, как научные сотрудники относятся к программированию).

В других областях Питона почти нет. Знаете ли вы мобильные приложения на Питоне? Десктопные (кроме Dropbox, над которым, как сказано выше, работал создатель языка, и, кстати, там используется не совсем Питон, а его модификация с JIT под названием Pyston)? Для микроконтроллеров? Операционные системы? Базы данных? Системы виртуальной реальности? Всё это бывает, но в маргинальных количествах.

Мобильные приложения пишутся на Java/Kotlin/Objective C/Swift/JS. Десктопные — на C++/C#/Java/JS. В микроконтроллерах, ОС и БД царит Си. В системах виртуальной реальности — C#/C++/JS.

Но поговорим о питоновских нишах.

4.1. Скрипты

Скрипты бывают системные (для нужд ОС) или внутри больших приложений. Системные скрипты на Питоне не имеют выдающихся преимуществ перед скриптами на Bash, Perl, BAT, Powershell и Tcl. Питон вполне годится для системного скриптования, но по историческим причинам не занимает первое место в этой нише и вряд ли когда-либо займёт.

Скрипты в приложениях и видеоиграх? В Adobe преимущественно используется JS и его производные. В GIMP и Autocad — преимущественно Лисп. В Microsoft Office — преимущественно VBA. В World of Warcraft и Dota используется Lua. В Minecraft — JS. В играх Valve — Squirrel. Питон нередко встречается как опция (например, в Excel в 2023 появилась экспериментальная поддержка), но, как и в системных скриптах, никогда не станет лидером.

4.2. Веб-фреймворки

На Питоне написано множество скриптов, реализующих серверную часть веб-приложений. Для этих скриптов сделаны популярные фреймворки (Django, Flask). Получается достаточно хорошо, если не считать свойственных Питону проблем с быстродействием, памятью и отсутствием тредов.

Тем не менее Питон используется лишь в 1,4 % сайтов [16], а лидером серверной веб-разработки является PHP (76,7 %). Питонисты смотрят на PHP свысока и считают его отсталым языком, но большая часть этих претензий относится к PHP 5-й версии (2004), тогда как в PHP 7 (2015) язык был подвергнут серьёзной переработке, а в 8-й версии получил вдобавок JIT-компилятор и стал полноценным современным языком. С обратной совместимостью ситуация в PHP не лучше, чем в Питоне, но всё же PHP 8 используют 17,5 % сайтов [17] — в 12 раз больше, чем Питон.

Среди популярных сайтов доля Питона выше, чем в среднем, но далека от доминирующей.

- Google Search — в основном написан на C++, на Питоне только поисковый робот.
- Youtube — написан на Питоне, но собственно раздача видеоконтента является обязанностью специальной сети доставки контента (Media CDN); на каком языке она написана, не говорится, но вряд ли на Питоне.
- Facebook — написан преимущественно на Hack, произошедшем от PHP.
- Twitter — написан на Java, Scala, Ruby. Возможно есть и Питон, но в небольших количествах.
- Википедия — написана на PHP.
- Reddit — на Питоне, отдельные части на Go.
- VK — написан на KPHP, произошедшем от PHP.
- Поиск Яндекса — преимущественно на C++ и Java.

Излишне упоминать, что серверные скрипты на Питоне не являются полноценными веб-серверами и обычно работают в связке с последними. Веб-серверы (Apache, nginx) написаны на Си.

4.3. Машинное обучение

Практически все нейросетевые модели разрабатываются на Питоне. Он действительно прекрасно для них подходит. Модели представляют собой последовательности однородных математических операций с массивами чисел, что позволяет автоматически вычислять необходимые в машинном обучении градиенты (см. дифференцируемое программирование). Недолговечность программ на Питоне в этом случае также не является проблемой, так как жизненный цикл моделей короток. Не является проблемой и быстродействие, так как ветвления и сложные структуры данных (самые слабые места языка с точки зрения быстродействия) в нейросетевых моделях не используются, а математические алгоритмы, лежащие в основе питоновских библиотек для машинного

обучения, реализованы на Си или CUDA. Так, в Pytorch менее половины кода написано на Питоне, в Tensorflow — менее трети.

Чтобы профессионально заниматься нейросетями, Питон знать недостаточно. Требуется знание линейной алгебры, математической статистики, нелинейной оптимизации и численных методов для всего этого. Питон — не первая дисциплина, которую придётся учить, и не первый язык программирования.

Кратко пройдёмся по биографиям нескольких знаменитых разработчиков в этой области. Андрей Карпатый писал на C++ и C, и до сих пор изредка пишет [18, 19]. Грег Брокман начинал карьеру с изучения теории языков программирования [20]. Алексей Крижевский, Илья Суцкевер и Джеффри Хинтон, создавшие знаменитую нейросеть AlexNet [21], написали её на CUDA. Излишне добавлять, что все эти разработчики учились в хороших университетах США и Канады и стали тем, кем они стали, благодаря широте полученного ими образования и опыта, а не благодаря Питону, на котором они пишут сейчас.

4.4. Наука

Заблуждение о том, что Питон играет в науке какую-то значимую роль, имеет две причины:

- Многие люди считают нейросети инструментом науки, хотя в действительности нейросети в научных исследованиях используются сравнительно редко, 8 % в среднем и ещё меньше, если исключить информатику [22].
- Научные работники не склонны к написанию полноценных программ; им требуется инструмент, позволяющий с помощью минимального количества кода осуществлять операции с данными и визуализацию в интерактивном режиме. Исторически для этого применялись Matlab, Mathcad и Mathematica, но в 2015 г. появился Jupyter Notebook и немедленно завоевал популярность благодаря своим средствам программирования, при всех недостатках всё же превосходивших указанные аналоги; к тому же Jupyter, в отличие от своих аналогов, бесплатен.

В остальном (кроме машинного обучения и Jupyter) использование Питона в науке минимально, в основном для биндингов к библиотекам на C/C++, одноразовых скриптов и студенческих проектов. Более распространённые в науке языки — вышеупомянутые C/C++, а также Фортран; специально для научных вычислений был разработан язык Julia.

Никаких уникальных качеств, востребованных в науке, Питон не имеет. Программу для научного исследования можно написать хоть на Bash (пример: статья в Nature [23]). Целью любого исследования является интерпретация данных, помещение результатов в научный контекст, извлечение смысла. Язык программирования на этом фоне второстепенен.

5. ОБРАЗОВАНИЕ

В 2008 г. в Массачусетском технологическом институте был прекращён [24] знаменитый курс 6.001 «Структура и интерпретация компьютерных программ», основанный на Лиспе. Один из авторов курса — профессор Джеральд Сассман — по собственному желанию перешёл с преподавания Лиспа на преподавание Питона. (При этом ему и его коллегам Питон не нравился [25]) На первый взгляд, это может показаться заменой менее популярного языка на более популярный, то есть уступкой моде, но в действительности

дело обстояло более печальным образом. Джеральд Сассман не просто перестал преподавать Лисп, а перестал преподавать программирование в том виде, в котором преподавал его 28 лет. Он решил, что профессия программиста, который создаёт программы с нуля, более не востребована, и сменил курс 6.001 на 6.01 (программирование роботов на Питоне с помощью готовых библиотек).

Между тем новые программы всё ещё нужны и будут нужны всегда (см. Айзек Азимов, «Профессия», 1957 [26]). Начинать подготовку специалиста с решения задач путём манипуляции готовыми компонентами — значит, забивать его голову преходящими концепциями и лишать фундаментальных знаний. Сассман был совершенно прав в выборе языка для этой цели и неправ в выборе самой цели.

В Питоне принята низкая культура программирования (если не считать культурой любование краткостью синтаксиса, даже если этим синтаксисом написана программа, работающая квадратичное время вместо линейного). Питон держит начинающего программиста в неведении относительно истинной модели выполнения программ (на CPU или на GPU). Синтаксис Питона поощряет использование неэффективных структур данных. Программисты на Питоне часто пишут программы для себя, а не для других. (Более того, в последние годы чаще пишут Jupyter-блокноты, чем программы.) Питонисты считают, что уже выучили окончательный язык программирования, и это самое страшное: если Питон является первым из изучаемых языков, то он становится и единственным. Профессиональные знания не удерживаются в голове студента, профессиональные навыки не формируются.

6. КАРЬЕРА

Зарплаты разработчиков на Питоне не имеют заметного отрыва от зарплат разработчиков на других языках. Например, начинающие на Питоне получают немного меньше, чем такие же на C++/C#/Java, и немного больше, чем такие же на JS и PHP (данные Хабр.Карьеры [27]). По количеству вакансий Питон занимает 2-е место (20 %) после JS (30 %), третье место у Java (18 %, данные DevJobsScanner [28]). Но всё это не имеет большого значения. Для каждого индивида существенная разница в зарплате достигается не при переходе на другой язык, а при повышении уровня квалификации (что и видно по кратным разницам в зарплатах между начинающими, средними и старшими разработчиками). Нет причины учить именно Питон или начинать именно с Питона; нужно учиться хорошо писать программы.

Как говорилось выше, если Питон стоит в самом начале учебной программы, то высоки шансы на то, что студент научится только Питону и больше ничему. Допустим, такой студент устроился на работу. Он верит, что Питон — это залог успеха, активно пишет код на своём любимом языке, но по сути дела у него ничего не получается из-за низкой квалификации и он увольняется. Затем на его место берут нового работника, более квалифицированного, с просьбой довести начатое до конца. Новый работник принимает на себя всю тяжесть свежего легаси, исправляет ошибочные технологические решения, рано или поздно переписывает весь код, и от изначальной версии ничего не остаётся. Ничего, кроме Питона в качестве языка программирования. Того же результата без Питона можно было бы достичь быстрее и приятнее. Универсальность и популярность Питона — это самоисполняющееся пророчество.

7. ЗАКЛЮЧЕНИЕ

Видимая популярность технологии не обязательно означает необходимость модифицировать под неё учебную программу. Сравним Питон с PowerPoint, пользователей которого в мире не менее полумиллиарда. Powerpoint используется практически во всех отраслях науки и техники (и даже искусства). Презентации в Powerpoint показывают все люди — от школьников до миллиардеров, от микробиологов до астрофизиков, от стажёров до директоров. Это инструмент, поистине объединивший человечество (за исключением вооружённых сил США, в которых запретили презентации в 2010-х годах [29]). В некоторых вузах преподаются курсы по подготовке презентаций. Нужны ли эти курсы? Может быть. Нужно ли преподавать их прежде курсов, составляющих профессию? Не нужно. Эти рассуждения применимы и к Питону.

Список литературы

1. TIOBE Software BV, “TIOBE Index for January 2024,” 2024. [Online]. Available: https://www.tiobe.com/tiobe-index
2. Stack Exchange Inc., “2023 Developer Survey,” 2023. [Online]. Available: https://survey.stackoverflow.co/2023/#most-popular-technologies-language
3. The Debsources developers, “Statistics: bookworm,” 2023. [Online]. Available: https://sources.debian.org/stats/bookworm/
4. “Why am I not getting better performance with numpy.searchsorted over bisect.bisect_left on a list of datetime?,” 2019. [Forum]. Available: https://stackoverflow.com/q/54465277
5. “Find nearest value in numpy array,” 2022. [Forum]. Available: https://stackoverflow.com/a/41856629
6. PyTorch Foundation, “Packing app with Pyinstaller rises OSError: TorchScript requires source access in order to carry out compilation, make sure original .py files are available,” 2021. [Online]. Available: https://github.com/pytorch/pytorch/issues/54828
7. L. Radke, “Numba jit and Scipy,” 2020. [Forum]. Available: https://stackoverflow.com/questions/63236229/numba-jit-and-scipy
8. Max Bélanger and Damien DeVille, “How we rolled out one of the largest Python 3 migrations ever,” in *https://dropbox.tech*, 2018. [Online]. Available: urlhttps://dropbox.tech/application/how-we-rolled-out-one-of-the-largest-python-3-migrations-ever
9. SQLite Consortium “How SQLite Is Tested,” 2024. [Online]. Available: https://www.sqlite.org/testing.html
10. Python Software Foundation, “What’s New In Python 3.12,” 2023. [Online]. Available: https://https://docs.python.org/3/whatsnew/3.12.html
11. Oracle Corp., “Oracle JDK Migration Guide. 4 Removed APIs,” 2024. [Online]. Available: https://docs.oracle.com/en/java/javase/21/migrate/removed-apis.html
12. G. Quénot, “Sedtris,” 2023. [Online]. Available: https://github.com/sputnick-dev/sedtris
13. M. Kalske, “Posh3d_cube_ball,” 2021. [Online]. Available: https://github.com/mi4c/Posh3d_cube_ball
14. Digital Bazaar Inc., “Forge,” 2021. [Online]. Available: https://github.com/digitalbazaar/forge
15. Chochain, “TensorForth - lives in GPU, does linear algebra and machine learning,” 2023. [Online]. Available: https://github.com/chochain/tensorForth
16. Q-Success, “Usage statistics of server-side programming languages for websites,” 2023. [Online]. Available: https://w3techs.com/technologies/overview/programming_language
17. Q-Success, “Usage statistics of PHP Version 8 for websites,” 2023. [Online]. Available: https://w3techs.com/technologies/details/pl-php/8
18. A. Karpathy, “ScriptBots,” 2012. [Online]. Available: https://github.com/karpathy/scriptsbots
19. A. Karpathy, “llama2.c,” 2023. [Online]. Available: https://github.com/karpathy/llama2.c/
20. G. Brockman, “My path to OpenAI,” 2016. [Online]. Available: https://blog.gregbrockman.com/my-path-to-openai

21. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017, doi: 10.1145/3065386
22. R. Van Noorden and J. M. Perkel, “AI and science: what 1,600 researchers think,” *Nature*, vol. 621, no. 7980, pp. 672–675, 2023; doi:10.1038/d41586-023-02980-0
23. C. Gorgulla et al., “An open-source drug discovery platform enables ultra-large virtual screens,” *Nature*, vol. 580, no. 7805, pp. 663–668, 2020; doi: 10.1038/s41586-020-2117-z
24. E. Broder, “The End of an Era. The last lecture of 6.001,” 2008. [Online]. Available: https://mitadmissions.org/blogs/entry/the_end_of_an_era_1/
25. J. A. Ortega, comment, “Sussmaniana,” 2009. [Online]. Available: https://jaortega.wordpress.com/2009/03/29/sussmaniana/
26. *Азимов А.* Профессия / Девять завтра: сборник. Нью-Йорк: Doubleday, 1959. С. 11–68.
27. Habr career, “Salary forks in spring 2023: programming languages and frameworks,” 2023 (in Russian). [Online]. Available: https://habr.com/ru/companies/habr_career/articles/746038/
28. Logan dev, “Top 8 Most Demanded Programming Languages in 2023,” 2023. [Online]. Available: https://www.devjobsscanner.com/blog/top-8-most-demanded-programming-languages/
29. E. Bumiller, “We Have Met the Enemy and He Is PowerPoint,” *New York Times*, Section A, p. 1, 27 Apr. 2010. Available: https://www.nytimes.com/2010/04/27/world/27powerpoint.html

Поступила в редакцию 01.12.2023, окончательный вариант — 21.12.2023.

Павлов Дмитрий Алексеевич, кандидат физико-математических наук, доцент кафедры алгоритмической математики СПбГЭТУ «ЛЭТИ», ✉ dapavlov@etu.ru

Computer tools in education, 2024
№ 1: 85–95
http://cte.eltech.ru
doi:10.32603/2071-2340-2024-1-85-95

Should We Teach Python at Universities?

Pavlov D. A.[1], Cand. Sc., Associate Professor, ✉ dapavlov@etu.ru

[1]Saint Petersburg Electrotechnical University, 5, building 3, st. Professora Popova, 197022, Saint Petersburg, Russia

Abstract

Various measures of popularity of programming languages in different scopes are discussed in the paper, with Python being the main point of interest. Well-known drawbacks of Python as a language and as an ecosystem (not inherent to other popular languages) are listed. The usefullness of Python in carreer is considered overrated, while in education Python often does more harm than good.

Keywords: *Python, programming, education.*

Citation: D. A. Pavlov., “Should We Teach Python at Universities?,” *Computer tools in education*, no. 1, pp. 85–95, 2024 (in Russian); doi:10.32603/2071-2340-2024-1-85-95

References

1. TIOBE Software BV, “TIOBE Index for January 2024,” 2024. [Online]. Available: https://www.tiobe.com/tiobe-index
2. Stack Exchange Inc., “2023 Developer Survey,” 2023. [Online]. Available: https://survey.stackoverflow.co/2023/#most-popular-technologies-language
3. The Debsources developers, “Statistics: bookworm,” 2023. [Online]. Available: https://sources.debian.org/stats/bookworm/
4. “Why am I not getting better performance with numpy.searchsorted over bisect.bisect_left on a list of datetime?,” 2019. [Forum]. Available: https://stackoverflow.com/q/54465277
5. “Find nearest value in numpy array,” 2022. [Forum]. Available: https://stackoverflow.com/a/41856629
6. PyTorch Foundation, “Packing app with Pyinstaller rises OSError: TorchScript requires source access in order to carry out compilation, make sure original .py files are available,” 2021. [Online]. Available: https://github.com/pytorch/pytorch/issues/54828
7. L. Radke, “Numba jit and Scipy,” 2020. [Forum]. Available: https://stackoverflow.com/questions/63236229/numba-jit-and-scipy
8. Max Bélanger and Damien DeVille, “How we rolled out one of the largest Python 3 migrations ever,” in *https://dropbox.tech*, 2018. [Online]. Available: urlhttps://dropbox.tech/application/how-we-rolled-out-one-of-the-largest-python-3-migrations-ever
9. SQLite Consortium “How SQLite Is Tested,” 2024. [Online]. Available: https://www.sqlite.org/testing.html
10. Python Software Foundation, “What’s New In Python 3.12,” 2023. [Online]. Available: https://https://docs.python.org/3/whatsnew/3.12.html
11. Oracle Corp., “Oracle JDK Migration Guide. 4 Removed APIs,” 2024. [Online]. Available: https://docs.oracle.com/en/java/javase/21/migrate/removed-apis.html
12. G. Quénot, “Sedtris,” 2023. [Online]. Available: https://github.com/sputnick-dev/sedtris
13. M. Kalske, “Posh3d_cube_ball,” 2021. [Online]. Available: https://github.com/mi4c/Posh3d_cube_ball

14. Digital Bazaar Inc., “Forge,” 2021. [Online]. Available: https://github.com/digitalbazaar/forge
15. Chochain, “TensorForth - lives in GPU, does linear algebra and machine learning,” 2023. [Online]. Available: https://github.com/chochain/tensorForth
16. Q-Success, “Usage statistics of server-side programming languages for websites,” 2023. [Online]. Available: https://w3techs.com/technologies/overview/programming_language
17. Q-Success, “Usage statistics of PHP Version 8 for websites,” 2023. [Online]. Available: https://w3techs.com/technologies/details/pl-php/8
18. A. Karpathy, “ScriptBots,” 2012. [Online]. Available: https://github.com/karpathy/scriptsbots
19. A. Karpathy, “llama2.c,” 2023. [Online]. Available: https://github.com/karpathy/llama2.c/
20. G. Brockman, “My path to OpenAI,” 2016. [Online]. Available: https://blog.gregbrockman.com/my-path-to-openai
21. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017, doi: 10.1145/3065386
22. R. Van Noorden and J. M. Perkel, “AI and science: what 1,600 researchers think,” *Nature*, vol. 621, no. 7980, pp. 672–675, 2023; doi:10.1038/d41586-023-02980-0
23. C. Gorgulla et al., “An open-source drug discovery platform enables ultra-large virtual screens,” *Nature*, vol. 580, no. 7805, pp. 663–668, 2020; doi: 10.1038/s41586-020-2117-z
24. E. Broder, “The End of an Era. The last lecture of 6.001,” 2008. [Online]. Available: https://mitadmissions.org/blogs/entry/the_end_of_an_era_1/
25. J. A. Ortega, comment, “Sussmaniana,” 2009. [Online]. Available: https://jaortega.wordpress.com/2009/03/29/sussmaniana/
26. I. Asimov, “Profession” in *Nine tomorrows*, New York City, NY: Doubleday, 1959, pp. 11–68.
27. Habr career, “Salary forks in spring 2023: programming languages and frameworks,” 2023 (in Russian). [Online]. Available: https://habr.com/ru/companies/habr_career/articles/746038/
28. Logan dev, “Top 8 Most Demanded Programming Languages in 2023,” 2023. [Online]. Available: https://www.devjobsscanner.com/blog/top-8-most-demanded-programming-languages/
29. E. Bumiller, “We Have Met the Enemy and He Is PowerPoint,” *New York Times*, Section A, p. 1, 27 Apr. 2010. Available: https://www.nytimes.com/2010/04/27/world/27powerpoint.html

Received 01-12-2023, the final version — 21-12-2023.

Dmitry Pavlov, Candidate of Sciences (Phys.-Math.), Associate Professor of the Algorithmic Mathematics Department, Saint Petersburg Electrotechnical University, ✉ **dapavlov@etu.ru**