УДК 004.624

Концепция универсального языка программирования

Кизянов Антон Олегович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Глаголев Владимир Александрович
Приамурский государственный университет имени Шолом-Алейхема
к.г.н., доцент кафедры информационных систем, математики и правовой информатики

Аннотация
В данной статье описано разнообразие языков программирования со сферой их применения. Собрана информация по рейтингам самых популярных языков программирования за последние 10 лет с их тенденцией во времени. Приведено обоснование невозможности создания универсального языка программирования с последующим его повсеместным использованием.
Ключевые слова: Универсальный язык программирования, Python, C, C++, Java, JavaScript

Concept of a universal programming language

Kizyanov Anton Olegovich
Sholom-Aleichem Priamursky State University
Student

Glagolev Vladimir Alexandrovich
Sholom-Aleichem Priamursky State University
Candidate of Geographical Sciences, Associate Professor of the Department of Information Systems, Mathematics and Legal Informatics

Abstract
This article describes a variety of programming languages with their scope. Collected information on the ratings of the most popular programming languages over the past 10 years with their trend over time. The substantiation of the impossibility of creating a universal programming language with its subsequent widespread use is given.
Keywords: Universal programming language, Python, C, C ++, Java, JavaScript

При использовании одного языка программирования в проекте часто требуется возможность широкого применения данного языка, так как при расширении функционала расширяются и сферы работы языка, что может затруднить дальнейшее его применение. В таких случаях приходится

прибегать к другим языкам программирования, которые могут в полной мере решить проблему расширения существующего проекта.

Главным аспектом при выборе языка программирования выступают скорость разработки и скорость выполнения кода, а сфера работы ограничивается начальными интересами проекта и не поднимается до тех пор, пока такой необходимости не потребуется.

Цель исследования – провести поиск и анализ языков программирования, сформулировать рейтинг популярных языков программирования и сферы их применения за последние 10 лет, сформулировать концепцию универсального языка программирования.

Ранее этим вопросом интересовалась редакция techrocks [1], они приводили мнения программистов о существовании универсального языка программирования. В статье приводились доводы в пользу специализированного инструмента для каждой отдельной задачи, как например изготовление финального продукта из ствола дерева, что существуют специальные инструменты для каждого из этапов обработки древесины для доведения к конечному продукту. Нуралы Аружан в статье «Лучшие языки программирования для изучения в 2021 году» [2] приводит рейтинг лучших языков программирования для изучения в 2021 году на основании их рентабельности на рынке и излечиваемости. Для каждого языка программирования он приводит сильные стороны языка и в какой сфере он востребован на данный момент. Сайт itanddigital.ru [3] приводит историю языков программирования и их эволюцию. Начиная с 1843 года с машинного алгоритма «Ады Лавлейс» идет история языков программирования. Каждый последующий язык привносил новые функции и возможности, в зависимости от потребностей и наследовал функционал своих предшественников, тем самым впитываю в себя проверенные функции и улучшая их. Но каждый язык программирования создавался под конкретную задачу своего времени.

Первая эра программирования была построена непосредственно в машинных кодах, а основным носителем информации были перфокарты и перфоленты. Программисты обязаны были знать архитектуру машины досконально. Программы были достаточно простыми, что обуславливалось, во-первых, весьма ограниченными возможностями этих машин, и, во-вторых, большой сложностью разработки и, главное, отладки программ непосредственно на машинном языке. Вместе с тем такой способ разработки давал программисту просто невероятную власть над системой. Становилось возможным использование таких хитроумных алгоритмов и способов организации программ, какие и не снились современным разработчикам.

Первым значительным шагом представляется переход к языку ассемблера. Не очень заметный, казалось бы, шаг переход к символическому кодированию машинных команд имел на самом деле огромное значение. Программисту не надо было больше вникать в хитроумные способы кодирования команд на аппаратном уровне. Более того, зачастую одинаковые по сути команды кодировались совершенно различным образом в

зависимости от своих параметров. Появилась также возможность использования макросов и меток, что также упрощало создание, модификацию и отладку программ. Появилось даже некое подобие переносимости — существовала возможность разработки целого семейства машин со сходной системой команд и некоего общего ассемблера для них, при этом не было нужды обеспечивать двоичную совместимость.

Вместе с тем, переход к новому языку таил в себе и некоторые отрицательные стороны. Становилось почти невозможным использование всяческих хитроумных приемов сродни тем, что упомянуты выше. Кроме того, здесь впервые в истории развития программирования появились два представления программы: в исходных текстах и в откомпилированном виде. Сначала, пока ассемблеры только транслировали мнемоники в машинные коды, одно легко переводилось в другое и обратно, но затем, по мере появления таких возможностей, как метки и макросы, дизассемблирование становилось все более и более трудным делом. К концу ассемблерной эры возможность автоматической трансляции в обе стороны была утеряна окончательно. В связи с этим было разработано большое количество специальных программ-дизассемблеров, осуществляющих обратное преобразования, однако в большинстве случаев они с трудом могут разделить код и данные. Кроме того, вся логическая информация (имена переменных, меток и т.п.) теряется безвозвратно. В случае же задачи о декомпиляции языков высокого уровня примеры удовлетворительного решения проблемы и вовсе единичны.

В 1954 году в недрах корпорации IBM группой разработчиков во главе с Джоном Бэкусом был создан язык программирования Fortran. Это первый язык программирования высокого уровня. Впервые программист мог по-настоящему абстрагироваться от особенностей машинной архитектуры. Ключевой идеей, отличающей новый язык от ассемблера, была концепция подпрограмм.

В 1960 году был создан язык программирования Cobol. Он задумывался как язык для создания коммерческих приложений, и он стал таковым. На Коболе написаны тысячи прикладных коммерческих систем. Отличительной особенностью языка является возможность эффективной работы с большими массивами данных, что характерно именно коммерческих приложений.

В 1960 году командой во главе с Петером Науром был создан язык программирования Algol. Этот язык дал начало целому семейству алгол-подобных языков (важнейший представитель — Pascal).

В 1963 году в Дартмурском колледже был создан язык программирования BASIC. Язык задумывался в первую очередь как средство обучения и как первый изучаемый язык программирования. Он предполагался легко интерпретируемым и компилируемым. Надо сказать, что BASIC действительно стал языком, на котором учатся программировать.

В 1970 году Никлаусом Виртом был создал язык программирования Pascal для обучения программированию. Язык замечателен тем, что это

первый широко распространенный язык для структурного программирования. Впервые оператор безусловного перехода перестал играть основополагающую роль при управлении порядком выполнения операторов.

В 1986 году Бьярн Страуструп создал первую версию языка C++, добавив в язык C объектно-ориентированные черты, взятые из Simula, и исправив некоторые ошибки и неудачные решения языка. C++ продолжает совершенствоваться и в настоящее время. Язык стал основой для разработки современных больших и сложных проектов. У него имеются, однако же, и слабые стороны, вытекающие из требований эффективности.

Большинство компьютерных архитектур и языков программирования ориентированы на последовательное выполнение операторов программы. В настоящее время, однако же, существуют программно-аппаратные комплексы, позволяющие организовать параллельное выполнение различных частей одного и того же вычислительного процесса. Для программирования таких систем необходима специальная поддержка со стороны средств программирования, в частности, языков программирования. Некоторые языки общего назначения содержат в себе элементы поддержки параллелизма, однако же программирование истинно параллельных систем требует подчас специальных приемов.

В сети интернет существует множество рейтингов языков программирования. Все они строятся на разных критериях, некоторые берут в основу требования работодателей к притендентам за различные года, некотрые сравнивают частоту упоминания языков программирования в поисковой выдаче поисковых сервисов, некоторые сравнивают активность побличных репозиториев разработки языков программирования и т.д.

Рейтинг TIOBE берет свою историю в 2003 году, он ведет свой рейтинг по результатам поисковых запросов, содержащих название языка. Для формирования индекса используется поиск в нескольких наиболее посещаемых порталах: Google, Wikipedia, YouTube, Baidu, Yahoo!, Bing, Amazon. Расчёт индекса происходит ежемесячно. Текущая информация предоставляется бесплатно.

На рисунке 1 изображен сводный график популярности языков программирования в процентном соотношении з апоследние 10 лет. По данным за последние 10 лет можно сделать вывод, что такие языки как C, C++, Java, C# постоянно находились в рейтинге каждый год, они формировали костяк написанного ПО за эти года, а такие языки как F#, R, Groovy, PL/SQL, Delphi понемногу покидают рейтинг, что говорит, что в будущем они совсем выйдут из него как такие языки как SAS, RPG (OS/400), Transact-SQL, Ada, Logo, F# и другие. Тенденция ведет к вытеснению специализированных языков к более универсальным.

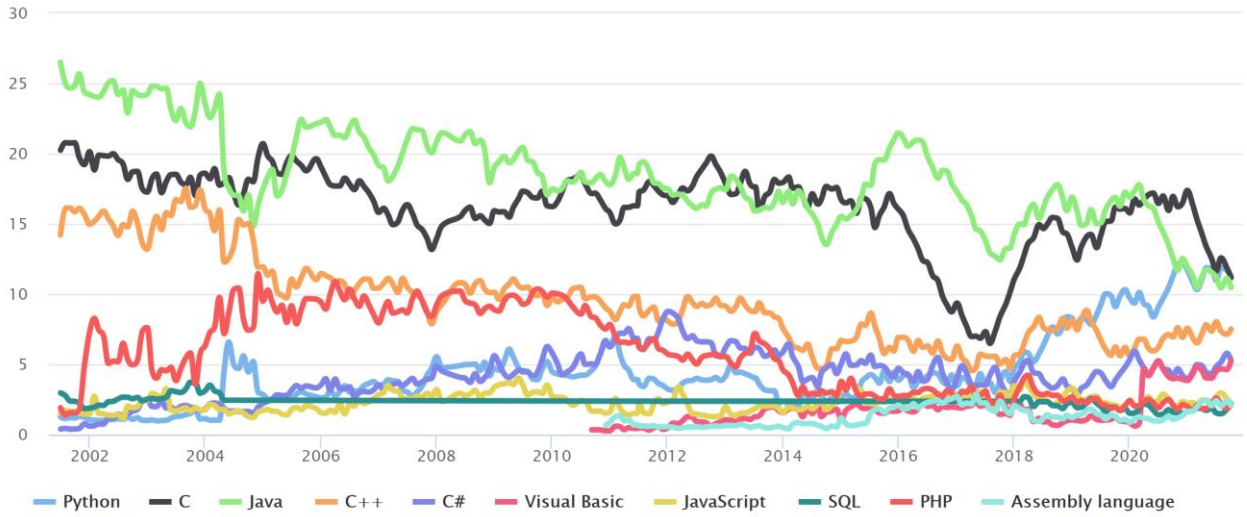

Рис. 1 Сводный график рейтинга языков программирования за 10 лет

Сферы применения языков за последние 10 лет

В программировании разделяют следующие сферы применения языков программирования: Веб программирование, Разработка игр, Мобильная разработка, Разработка ПО.

Веб программирование

Данная сфера считается самой простой для входа. В ней подразделяют 2 стороны Front-End и Back-End. Это две составляющие для любого проекта в веб сфере. Front-end включает в себя использование языка программирования JavaScript.Back-end включает уже больше языков программирования, таких как Python, PHP, Ruby, C++, C#, JavaScript и других.

Разработка игр

Разрабатывать игры можно самому, а можно при использовании готовых движков. Если разрабатывать игры самому, то нужен низкоуровневый язык программирования, что будет выполнятся максимально быстро. Тут явным лидером являются языки C и C++. В то же время, можно встретить разработку игр еще на Java.

Если говорить про игровые движки, то выбор побольше. Для крупных 3D проектов выбирают крупные движки: Unreal Engine, Unity или же CryEngine. Но их использование не избавляет от написания кода самостоятельно, для Unreal Engine и CryEngine потребуется C++, для Unity C#.

Мобильная разработка

В сфере мобильной разработки разделяют 2 пути, разработка под устройства Android и iOS устройства. Для первых потребуется знания языка Java или более современный Kotlin, под второй знание Objective-C или более современный Swift.

Разработка ПО

Данная сфера самая обширная, так как в ней находятся все остальные не вошедшие в первые 3 сферы, от написания простых утилит для компьютера до разработки ПО для управления атомной электростанции.

В эту категорию попадают языки программирования как C, C++, C#, Java, Python. Они попали из-за их универсальности, что требуется при использовании их на различных операционных системах таких как Windows, Linux, Mac и других.

В последнее время сюда еще относят сферу нейронных сетей и анализа данных, в чем универсальные языки также вытесняют более профильные языки программирования, например, Python со временем вытеснил язык R и MATLAB в этой сфере.

Концепция универсального языка программирования

Универсальный язык должен достигать как высокого, так и низкого уровня: он должен максимально использовать абстракции, но, когда нужно производительность, программист должен быть в состоянии достичь необходимой глубины в аппаратном обеспечении и манипулировать битами, если это необходимо. Такие языки существуют, и они называются «трансцендентными».

Некоторые из этих языков, такие как Common Lisp, Scheme и Smalltalk начинают как высокоуровневые, но при этом имеют механизмы (начиная от опционального ввода и заканчивая представлением для аппаратных команд низкого уровня с использованием встроенного ассемблера), позволяющие с легкостью достичь низкого уровня. Другие же, например, Forth, начинают как низкоуровневые, но могут с легкостью переходить на высокий уровень. Такие языки в прямом смысле позволяют создавать мини-языки, идеально вписывающиеся в оригинальный синтаксис исходного языка, что заметно упрощает решение сложных задач.

Для создания операционных систем с нуля использовались некоторые языки более высокого уровня. C – вездесущий язык, созданный в то время, когда ресурсов было не так много, но он занял золотую середину между простотой, низкоуровневостью и высокоуровневостью и стал очень популярен, несмотря на то, ему не хватает гибкости.

Итак, если эти языки способны на все, почему их не использует все больше людей? К сожалению, причина заключается в их простоте. Для ее достижения им приходится жертвовать математической приоритетностью, что многих сбивает с толку.

- Lisp использует префиксную нотацию — (+ 1 2 3) и получает 6.
- Forth использует постфиксную нотацию — «1 2 + 3 +». Получает 6.
- Smalltalk оценивает все слева направо, поэтому из «1 + 2 * 3» получает 9

И хотя, как большинство поначалу находят это странным, но все не так запутанно. Это всего лишь дело привычки. Сравните эти языки с любым

языком на основе Algol (C, Pascal, PHP, Python и т. д.), в котором присутствует иерархия выполнения математических действий, которая, как это ни парадоксально, имеет тонкие отличия от одного языка к другому. Все эти правила слишком многочисленны и труднозапоминаемы. Легче просто заключить все в круглые скобки, чтобы не пришлось беспокоиться о причудливых правилах приоритетности.

Компьютер в данном случае – это чистый холст памяти, ждущий нанесения красок в виде скрипта, приложения или утилита. В зависимости от того, что программист хочет получить в итоге, и от того, с чего начинает, он выбирает язык программирования, каждый из которых имеет свою специфику. Некоторые языки просты и приятны в обращении, как пила или разводной ключ. К таким языкам относятся Basic и JavaScript. Для достижения высокой скорости и точности программирования существуют языки C и C++. Это утверждение может вызвать массу споров, но высокая скорость и точность в этом контексте связаны с операционными системами, алгоритмами и другими критическими для скорости задачами программирования. Мистер Тьюринг научил нас тому, что и операционную систему можно написать на Perl или Javascript, но можно и из дуба наждачной бумагой стол вышлифовать, если, конечно, хватит наждачной бумаги, времени и терпения. О практичности речь не идет.

Вывод

Области применения языков программирования очень разнообразны: интегральные схемы, веб, системное ПО и т.п. Уникальный супер-язык — это универсальность, а, как известно, универсальность ведёт к потере каких-то «фишек», что не всегда приемлемо. Кроме того, принципы ООП очень хороши, но не являются «серебряной пулей». Сейчас быстрыми темпами всё большую популярность набирает Функциональное программирование, которое отлично подходит для параллельного/конкурентного программирования, особенно в эпоху производительности, большого объёма данных и остановки роста частот, на которых работают процессоры. Сейчас целенаправленное создание супер-языка не имеет смысла из-за отсутствия целевой аудитории. На данный момент уже есть огромное количество специалистов, умеющих работать в существующих языках. Массово всем изучать, переучиваться на новый язык экономически нецелесообразно. Кроме специалистов, есть много наработок, библиотек и инструментов, которые нужно будет переписывать на новый язык, что тоже вряд ли будет хорошим решением. Сейчас просматривается плавная эволюция: одни языки сменяются другими, появляются улучшенные варианты (примеры — Java в Kotlin или Scala, JavaScript в TypeScript), делаются попытки объединить языки.

Библиографический список

1. Почему не существует универсального языка программирования? //

Techrocks.ru | программирование, стартапы, новости IT URL: https://techrocks.ru/2018/07/11/universal-programming-language/ (дата обращения 10.12.2021)
2. Лучшие языки программирования для изучения в 2021 году // Open Access | by Zeba Academy URL: https://open.zeba.academy/luchshie-yazyki-programmirovaniya/ (дата обращения 10.12.2021)
3. История языков программирования // Кадровое агентство. Подбор ит специалистов. Услуги. Подбор IT и digital персонала в Москве URL: https://itanddigital.ru/historycoding (дата обращения 10.12.2021)
4. Универсальный язык программирования для Android, iphone и windows phone // Oh! Android URL: http://www.ohandroid.com/90054.html (дата обращения 10.12.2021)
5. История языков // Главная | МОУ «Осельковская ООШ» URL: https://oselkschool.ru/book/export/html/74 (дата обращения 10.12.2021)