Годишник Телекомуникации 2019, том 6, с. 1-15

Yearbook Telecommunications 2019, vol. 6, p. 1-15

eISSN 2534-854X https://telecommunications.nbu.bg/bg/godishnik-telekomunikacii

DOI: https://doi.org/10.33919/YTelecomm.19.6.1

СРАВНИТЕЛЕН АНАЛИЗ НА PYTHON С ДРУГИ ПРОГРАМНИ ЕЗИЦИ

Филип Андонов

COMPARATIVE ANALYSIS OF PYTHON WITH OTHER PROGRAMMING LANGUAGES

Filip Andonov

Резюме: Целта на настоящата статия е да предложи и аргументира набор от критерии за оценка и сравнение на програмни езици, с чиято помощ се извършва сравнение на езика Python с негови конкуренти за различни цели/приложения. В резултат е разработена методология, основана на многокритериален анализ, за сравнение и избор на конкретен програмен език за дадено приложение. Като принос може да се отчете както предложената методология, така и факта, че тя е универсална по своя характер и може да се използва и в анализа на други езици, чрез други критерии и с други тегла.
Ключови думи: многокритериален анализ, програмни езици, разработване на софтуер

Abstract: The goal of this paper is to present a set of criteria for comparison of program languages and give arguments for their selection. With their help The Python language is compared with its competitors for different applications. As a result a methodology for the selection of a program language is developed, based on multi-criteria analysis. The contribution is the developed methodology and the fact that it is universal by nature and can be used for the selection of a different set of languages, criteria and weights.
Keywords: multicritera analysis, programming languages, software development

1. ВЪВЕДЕНИЕ

Ако искаме да разберем какво прави Python толкова популярен, трябва да го сравним с други широко разпространени езици и за да не бъдем голословни, ще разглеждаме качествата му като критерии за оценка на езика. Изборът с кои езици да го сравняваме неизбежно съдържа субективен компонент, но за да разполагаме с формално основание, нека разгледаме списъка с най-популярните езици за 2018 година според индекса TIOBE. Те са Java, C, Python, C++, C#, Visual Basic .NET, JavaScript, PHP, Objective-C и SQL. От тях SQL всъщност не е език за програмиране, а Objective-C е твърде специализиран (само за iOS и MacOS), така че можем да ги премахнем от списъка. Ако решим да ограничим списъка до 5 претендента, то PHP, въпреки широкото си разпространение, отпада то него. Visual Basic.NET е странен претендент – той се приближава доста до C#, но е леко опростен, поне откъм синтаксис. Свързан е тясно с архитектурата на Microsoft, но за по-сериозни проекти за тази цел обикновено се използва C#. Той пък от своя страна по общи характеристики се приближава много до Java, което не е изненада предвид факта, че е създаден именно като отговор на Java по времето, когато Sun Microsystems и Microsoft бяха основни конкуренти. Де факто обаче и C# се използва само за проекти, дълбоко обвързани с технологиите на Microsoft, поради което можем да го премахнем или обединим заедно с Java в списъка ни. Така финалната версия е: Python, Java, C++, C, JavaScript.

2. КРИТЕРИИ

2.1. ЛЕСЕН ЗА УЧЕНЕ

Свойствата, които бяха изброени в първа глава несъмнено го правят лесен за изучаване, както като първи език (използвайки го като инструмент за въвеждане на основни концепции), така и като език за хора с предишен опит. Тук трябва да отбележим, че във втория случай кривата на сложност отначало е по-стръмна поради някои особености на синтаксиса на Python, но ако обучаемият преодолее този етап, то (поне според опита на автора) връщането към език с не така удобен синтаксис изисква известно усилие на волята.

Лесният синтаксис на езика помага бързо да се навлезе в него и сравнително рано да се пишат приложения, които представляват интерес за учещия го. Това е особено важно в съвременния цифров свят, в който за вниманието на всеки негов гражданин се конкурират безчет сайтове, реклами, видеа, мобилни приложения, мобилни игри и т. н. Така ускореното постигане на ефектен резултат става важен критерий при изучаването на език, за да не се загуби интереса на изучаващия го.

Динамичната типизация в Python го прави подходящ както за първи език – тъй като позволява да се остави подробното въвеждане на понятието *тип* за по-късно и да се намали когнитивното натоварване на изучаващия го. За производствени цели динамичната типизация също е по-скоро улеснение и инструмент за предотвратяване на някои видове грешки, разбира се, за сметка на други.

Пълната обектно-ориентираност и автоматичното управление на паметта повдигат нивото на абстракция и спестяват много затруднения с адресната аритметика и ръчното освобождаване на памет, отново улеснявайки изучаването му.

Много от останалите „скриптови“ езици са специализирани за конкретна дейност и това се отразява на синтаксиса и граматиката на езика. Поради това Python като език с общо предназначение, ако не улеснява, поне не усложнява работата по изучаването му.

Достъпността на езика също е критерий. Тъй като Python е свободен и с отворен код, той е наличен за всички ОС и приложения за него може да се пишат и изпълняват лесно дори в браузер.

Широкото му разпространение пък означава, че съществува огромен набор от учебни материали и ръководства във всевъзможни форми, както и голяма общност от разработчици, които да отговарят на всеки свързан с него въпрос в stackoverflaw.com.

Като цяло може да се каже че Python заема нишата, оставена от изчезващия Pascal/Delphi и донякъде Visual Basic 6. В зенита си Pascal беше идеален избор за първи език, тъй като именно обучението по програмиране е целта на създаването му от Никлаус Вирт. В периода около смяната на хилядолетието много хора навлязоха в програмирането с Delphi, което позволяваше изключително лесно да се пишат надеждни приложения с графичен интерфейс, както и с Visual Basic, който бе първият избор на не-специалисти, нуждаещи се да разработят софтуерно приложение.

В сравнение с останалите популярни в момента езици, авторът нарежда Python на първо място по лекота на учене. На второ е несъмнено JavaScript, доказателство за което е неимоверно широкото му разпространение, включително отвъд браузера – сървъри и вградени системи. Причината да не бъде на първо място е именно фактът, че той не е измислен като език с общо предназначение и въпреки че началото е лесно от дадена комплексност на разработваното софтуерно приложение, това почва да оказва влияние. Java е измислен като заместник на C++, решавайки някои от проблемите в него, така че несъмнено е по-лесен от него, ако не за друго, то дори само и заради автоматичното управление на паметта. По-ниското ниво и фактът, че те са чисто производствени езици, не особено подходящи за обучение по програмиране, поставя C++ и C накрая на нашата подредба.

Годишник Телекомуникации 2019, том 6, с. 1-15

Yearbook Telecommunications 2019, vol. 6, p. 1-15

eISSN 2534-854X https://telecommunications.nbu.bg/bg/godishnik-telekomunikacii

DOI: https://doi.org/10.33919/YTelecomm.19.6.1

Подредба:

- Python
- JavaScript
- Java
- C++
- C

2.2. ЛЕСЕН ЗА ЧЕТЕНЕ

Колко лесно се чете код на даден език е важно по няколко причини. Преизползваемостта на кода, към която бизнеса се стреми по икономически причини, означава, че код, който е писан в предишен момент и/или от друг разработчик, трябва да е лесен за разбиране. Лесният за четене и осмисляне код, който е и компактен, не позволява на разработчика да загуби нишката и да допусне грешка, а грешките струват скъпо в софтуерния свят, независимо на кой етап са отстранени. Документацията рядко обяснява всичко по програмния код, особено що се касае до въпросите как и защо. Ако кодът е разбираем, то важността на качествената и изчерпателна документация (което е химера) се намалява. Това качество на езика също спомага при изучаването му, за което стана въпрос по-горе.
Четимостта е залегнала в самата философия на Python и в резултат той е със сбит, но разбираем синтаксис, който почти изцяло елиминира кода – пълнеж (т. нар. fluff code).
C и C++ са пословично трудни за четене езици, а щедростта в изказа на Java прави кода обемен, като пълнежът скрива алгоритмичната същност.

Подредба:

- Python
- JavaScript
- Java
- C++
- C

2.3. ЛЕСЕН ЗА ПИСАНЕ

В известна степен това качество на езика се припокрива с лекотата, с която се учи. Когато един език се използва в софтуерното производство е важно той да има къс синтаксис, тъй като разработчиците не обичат да натискат клавишите излишно. Това е ясно изразено в езика C и някои негови производни. Освен къси ключови думи и скоби вместо думи за маркиране на блок, е важно също самата семантика на езика да не предразполага или изисква многословие, т. е. колко изразителен е езика. Постигането на тези цели, обаче, често е за сметка на четимостта. Python успява да реши този проблем доста умело, успявайки да нахрани вълка и да спаси агнето, като кодът написан на него е лесен и за писане и за четене. Той е доста експресивен език, като реализираните на него алгоритми са по-къси от тези на други езици. Лекотата за писане също зависи от удобна среда и компоненти на трети страни. Python не е обвързан с конкретна развойна среда, но съществува цял спектър от такива, които варират от прости редактори със синтактично оцветяване до тежки среди с вградени Lint системи, анализиране на кода, автоматично подсказване и т. н. Концепцията на езика за „batteries included“ пък означава, че допълнително функционалност се постига лесно – един ред за внасяне и няколко реда за имплементация е напълно достатъчно в много реални сценарии. Поради експресивността, изключителното разнообразие на качествени библиотеки и добрите среди Python отива на първа позиция. Огромната популярност на JavaScript вероятно е причината за много

добрите среди за разработка. Той е единственият скриптов език освен Python в списъка, а този тип езици улесняват писането. JavaScript взаимства базовия синтаксис от C, така че е с къс синтаксис, въпреки че специфичното му приложение ограничава експресивността. От останалите езици несъмнено Java е най-лесен за писане поради автоматичното управление на паметта. Така подредбата е в низходящ ред спрямо абстракцията на езика.

Подредба:

- Python
- JavaScript
- Java
- C++
- C

2.4. ПРОИЗВОДИТЕЛНОСТ

Въпросът за производителността на кода е комплексен, но несъмнено когато се цели сурова производителност Python не е първи избор. Програми, написани на C или C++ или дори Java се изпълняват по-бързо от еквивалентни на Python, понякога значително. Производителността на един програмен език често се разбира не само като скорост на изпълнение, а и като това колко бързо се написва работещ код (лекота на писане). Погледнато по този начин става ясно, че създателите на Python са се фокусирали върху бързото достигане на качествен код, но не непременно бърз за изпълнение. И за много от целите, за които се използва езика - например при научни изчисления – ограниченията в производителността наистина се компенсират от краткото време за написването на кода.
Суровата производителност се компенсира донякъде и с модули, написани на C, които извършват тежката работа при сложни алгоритми. Самата имплементация cPython също има място за подобрения и такива съществуват. JIT компилаторът pypy, който е съвместима алтернативна имплементация на езика, дава по техни данни (https://speed.pypy.org/) около 4 пъти по-висока производителност. Статичният компилатор Cython, който не е изчерпателна имплементация на Python, компилира Python код и дава между 20% и 60% по-добра производителност от стандартната cPython [1] имплементация.
Колкото по-малко абстрактен е един език толкова по-бърз е (при наличието на оптимизиращ компилатор). Затова C, C++ и Java държат първите три места. Неимоверната популярност на JavaScript доведе до големи инвестиции в JIT компилатори и подобряване на производителността му. Така Python остава на последно място в класацията по този критерий.

Подредба:

- C
- C++
- Java
- JavaScript
- Python

2.5. ПРИЛОЖИМОСТ

Има няколко сфери на приложение, в които Python е идеален кандидат.

Обучение

Фактът, че даден език е лесен за учене сам по себе си е индикатор, че е подходящ за обучение, тъй като всеки иска да постига максимален резултат за единица вложено

Годишник Телекомуникации 2019, том 6, с. 1-15

Yearbook Telecommunications 2019, vol. 6, p. 1-15

eISSN 2534-854X https://telecommunications.nbu.bg/bg/godishnik-telekomunikacii

DOI: https://doi.org/10.33919/YTelecomm.19.6.1

усилие. Именно за това много курсове, особено за ученици, използват Python като средство за обучение по програмиране. Развойният едноплатков компютър Raspberry Pi използва Python като основен език. MIT замениха SCHEME (диалект на LISP) от въвеждащия курс по програмиране с Python. Възможността на езика да комуникира с хардуер на ниско ниво позволява да се програмират роботи и други хардуерни платформи, което пък е приложимо в кръжоци и по-нестандартни форми на обучение. Лекотата на учене също така означава, че в Python пътя до писането на приложения с графичен инструмент е по-къс и по-лек в сравнение с езици като Java или C# и несравним спрямо C++ и C.

Хоби проекти, включително хардуерни

С нарастването на популярността на вградените системи и демократизирането на развойните инструменти [7] нарасна интересът към хардуерни хоби-проекти. Въпреки всичко те традиционно ползват езика C, въпреки опити да се популяризира C++, Java и дори JavaScript. Платформата Arduino не поддържа директно Python, но се появиха алтернативни платки с microPython, които значително улесняват разработката на IOT проекти. Също така както беше споменато в частта за лекота на учене, Python замести Delphi и Visual Basic като език за аматьори, които нямат за цел да стават програмисти, но поради едни или други причини искат да разработят сами софтуерен продукт.

Бързо прототипиране

Софтуерният пазар е изключително развит и продължаващ да се развива, което поставя опериращите в него бизнес единици в една наситено конкурентна среда. В такава ситуация е важно да може да се достигне бързо до прототип, особено при малки компании и startup-и. Лекотата на писане и мощността на Python го правят идеален за софтуерен процес, който утилизира подхода за бързо прототипиране.

Dev-ops и тестване

Фактът, че е скриптов, но универсален език, прави Python подходящ за тестване и dev-ops [2]. В този случай отново натежава качеството му да се пише лесно и бързо код на него. Той е значително по-лесен за учене от PERL, например, без да е по-малко функционален и много по-мощен от езиците на функционалната обвивка (bash shell например). Разработени са множество библиотеки и софтуерни рамки за тези цели и Python замества успешно традиционните в тази област инструменти.

Научни изчисления

Наличността на библиотеки с доказано качество за научни изчисления и безплатният му характер правят Python все по-използван в научните среди. Съвременната тенденция да се изисква кода при научните публикации също дава основание за разрастване на употребата му. Отворените данни и отворен код нямат голяма полза, ако те работят само с изключително скъп специализиран софтуер от типа на SPSS. Доказателство за приложимостта му в тази сфера са имена като CERN и NASA, които го използват.

Обработка на естествен език и машинно самообучение

Отново благодарение на своята универсалност и добри научни библиотеки Python стана дефакто стандарт в приложенията за машинно самообучение. Всички големи пакети от

библиотеки за невронни мрежи имат binding-и и препоръчват употребата му. В обработката на естествен език той все още има конкуренти, но също е чест избор.
В заключение можем да кажем, че Python за кратко време навлезе и зае позиции в различни сфери на приложение на софтуера. Разбира се има места, на които той не представлява сериозен кандидат, но това важи за всички езици. Може би най-универсален поне като потенциал е C++, макар че много места, на които може да се използва не се използва, тъй като има по-добри кандидати. Java е най-универсален като реално приложение в софтуерния свят. В областите на приложение най-силните конкуренти на Python са Java и JavaScript.

Работа с данни
Последните години има огромно развитие на областта „обработка на данни“, особено на големи обеми от данни (т. нар. big data, data mining). Универсалността на езика, наличието на високо-качествени библиотеки както за обработка на данни, така и за визуализация и за съхранение в не-релационни бази от данни, направиха Python лидер в този род приложения.

Обработка на естествен език и машинно самообучение
Отново благодарение на своята универсалност и добри научни библиотеки Python стана дефакто стандарт в приложенията за машинно самообучение. Всички големи пакети от библиотеки за невронни мрежи имат binding-и и препоръчват употребата му. В обработката на естествен език той все още има конкуренти, но също е чест избор.
В заключение можем да кажем, че Python за кратко време навлезе и зае позиции в различни сфери на приложение на софтуера. Разбира се има места, на които той не представлява сериозен кандидат, но това важи за всички езици. Може би най-универсален поне като потенциал е C++, макар че много места, на които може да се използва не се използва, тъй като има по-добри кандидати. Java е най-универсален като реално приложение в софтуерния свят. В областите на приложение най-силните конкуренти на Python са Java и JavaScript.
Подредба:

- Java
- C++
- Python
- JavaScript
- C

2.6. ПРЕНОСИМОСТ

Разнообразието на софтуерни и хардуерни платформи винаги е оказвало натиск върху езиците за програмиране подтиквайки ги кодът им да бъде преносим. Референтният интерпретатор на Python е реализиран за всички основни операционни системи – Windows, Mac Os, Linux и хардуерни платформи – X86 и ARM. Python обаче има имплементации на Java (Jython) и на C# (ironPython), което го прави приложим навсякъде, където тези платформи са налични.
Подредба:

- C
- C++
- Python/Java
- JavaScript

Годишник Телекомуникации 2019, том 6, с. 1-15

Yearbook Telecommunications 2019, vol. 6, p. 1-15

eISSN 2534-854X https://telecommunications.nbu.bg/bg/godishnik-telekomunikacii

DOI: https://doi.org/10.33919/YTelecomm.19.6.1

2.7. ПОПУЛЯРНОСТ

На пръв поглед популярността на езика е странен избор за критерий за оценката му. Но в съвременния свят хората не могат да си позволят да учат език, който заема ниша и е „бутиков“ по характер. Популярността също така гарантира наличие на общност, която отговаря на въпроси и наличие на много и разнообразни учебни материали и ръководства. Най-вече популярният език гарантира намирането на работа, което е основната причина повечето хора да се занимават с програмиране в наши дни.

Като пазарен дял Python не се нарежда на първо място. Ако погледнем популярният индекс TIOBE [3], той е на четвърто място за 2018, но на трето за първото полугодие на 2019-а. Според същата класация той е най-бързо растящия в класацията за 2018-а. Като перспективи Python се препоръчва навсякъде на челните две позиции, но нека използваме по-чистите данни на TIOBE.

Подредба:

- Java
- C
- Python
- C++
- JavaScript

2.8. ПОМОЩНИ ИНСТРУМЕНТИ И КОМПОНЕНТИ

Колкото и добър да е един език сам по себе си от практическа гледна точка, важен критерий е наличността на богата системна библиотека и компоненти на трети страни. Концепцията „batteries included“ на Python, за която стана въпрос, е причина вградените модули да са мощни и изключително лесни за използване. Същото важи и за компонентите на трети страни, които се инсталират тривиално и използват лесно. Хранилища за компоненти, подобни на CPAN за PERL има и за Python – най-популярното е Python Package Index [4], който се използва от пакетния мениджър pip.

Вместо това има няколко различни, вариращи по функционалност. Помощни инструменти от типа на lint има и за Python [5], но тези за Java са по-добре развити (нуждата там е по-голяма).

Подредба:

- Java
- Python, JavaScript
- C++
- C

3. ПРОГРАМНИ ЕЗИЦИ ЗА РАЗЛИЧНИ ЦЕЛИ (PYTHON ПЕРСПЕКТИВА)

За да се извърши реално сравнение на избраните езици, те трябва да се разглеждат в контекста на конкретно приложение. Имайки подредбите на избраните езици по всеки критерий можем да ги използваме като оценки/стойности и с тяхна помощ да решим няколко задачи за най-подходящ език за дадена цел. Това означава да изберем измежду няколко алтернативи (езици), оценявайки ги по множество критерии, което по същество е многокритериална задача. За да направим това обаче ще трябва да дадем приоритет или важност на всеки от критериите. Те по дефиниция са субективни, но авторът ще аргументира избора на тегла. За оценка на важността на критериите ще използваме скалата от 1 до 9 със следните вербални съответствия:

1. изключително ниско

2. извънредно ниско
3. много ниско
4. ниско
5. средно
6. високо
7. много високо
8. извънредно високо
9. изключително високо

Тъй като всички критерии са ранжиращи (алтернативите са подредени за всеки от тях, но нямат стойност) и са на минимум (по-ниската стойност е по-добра), можем да опростим значително методологията на решаване на многокритериалната задача. Матрицата алтернативи на критерии показва кой език на коя позиция за даден критерий е. Векторът на теглата на критериите е, разбира се, с размерност броя на критериите. Умножавайки матрицата по вектора, получаваме вектор на окончателна агрегирана подредба на алтернативите, взимайки предвид всички отегловени критерии.

3.1. НАЙ-ПОДХОДЯЩ ЕЗИК ЗА ОБУЧЕНИЕ

В задачата за избор на език за обучение е ясно, че лекотата с която се учи, е от изключителна важност. За да се учи език, е ценно да може да се чете лесно, поради което тук оценката е висока. Лекотата за писане е по-важна, тъй като при програмирането важи принципът „учене чрез правене“. Производителността на езика не е от съществено значение при обучението, тъй като обикновено задачите и примерите са малки и не особено изчислително интензивни. Приложимостта е много важна за самите обучаеми, тъй като, както бе споменато по-горе, те предпочитат да учат програмиране с език, който е утвърден на пазара, а не е специализиран само за обучение. Преносимостта на езика има значение доколкото позволява гъвкавост в оборудването на учебното заведение и не ограничава обучаемите в избора на ОС и хардуерна платформа за самоподготовка. Независимо от формата на обучение, в наши дни се ползва Интернет за търсене на допълнителни материали и отговори на въпроси, поради което популярността е с висока важност. Според една поговорка, инструментът прави половината майстор, а колкото по-начинаещ е човек толкова по-голяма нужда има да разчита на добри инструменти. От друга страна, за да се научи, понякога човек трябва да мине през по-прости инструменти, за да разбере как функционират нещата, които инструментът дава на готово. Затова средата и помощните инструменти получават много висока, но не повече, важност.

- Лесен за учене – изключително високо (9)
- Лесен за четене – високо (6)
- Лесен за писане – много високо (7)
- Производителност - ниско (4)
- Приложимост – много високо (7)
- Преносимост – средно (5)
- Популярност - високо (6)
- Помощни инструменти и компоненти – много високо (7)

След решаването на задачата се получава следният краен резултат:

- Python
- Java
- JavaScript
- C++
- C

Годишник Телекомуникации 2019, том 6, с. 1-15

Yearbook Telecommunications 2019, vol. 6, p. 1-15

eISSN 2534-854X https://telecommunications.nbu.bg/bg/godishnik-telekomunikacii

DOI: https://doi.org/10.33919/YTelecomm.19.6.1

3.2. НАЙ-ПОДХОДЯЩ ЕЗИК ЗА ОБРАБОТКА НА ЕСТЕСТВЕН ЕЗИК

При компютърната обработка на естествен език често участват не само софтуерни специалисти, но и лингвисти, филолози, статистици и др. Поради това лекотата, с която езикът се учи, има осезаемо значение. Именно защото специалисти от различни области ще работят върху алгоритмите, има значение и колко лесно ще се чете програмния код. Обработката на естествен език е бързо развиваща се област с голяма конкуренция. Нови алгоритми и подходи се измислят непрекъснато, което поражда често задачи за прототипиране. Поради това, лекотата за писане е от много голямо значение. Повечето съвременни задачи в тази област използват машинно самообучение по един или друг начин, а то е извънредно ресурсоемко. Това означава, че теглото на производителността трябва да отрази този факт. Дефинирахме приложимост като свойство, което показва колко приложим в различни области е даден език. Поради това е некоректно да използваме този критерий, когато сме се ограничили до една такава област. Така на критерият приложимост даваме тегло 0, с което на практика го елиминираме от задачата. Преносимостта е важна, тъй като приложенията на обработката на естествен език обхващат различни архитектури и класове компютри – от мобилни, през настолни, до сървъри, със съответните операционни системи и софтуерни стекове. Разработването на най-модерни сложни алгоритми означава, че разработчиците на софтуера често се намират в непознати води и добрата общност и наличието на ръководства е от много голямо значение. Поради същите причини наличието на удобни инструменти е от особено голямо значение.

Теглата на критериите са:

- Лесен за учене – високо (6)
- Лесен за четене – средно (5)
- Лесен за писане – много високо (7)
- Производителност – извънредно високо (8)
- Приложимост – не е релевантна (0)
- Преносимост – високо (6)
- Популярност - много високо (7)
- Помощни инструменти и компоненти – извънредно високо (8)

След решаването на задачата се получава следният краен резултат:

- Python
- Java
- C
- JavaScript
- C++

Първите две места отразяват в известна степен реалната картина. На пръв поглед C изглежда не на място на трета позиция, но суровата скорост е критична при големи обеми и някои библиотеки са написани именно на C.

3.3. НАЙ-ПОДХОДЯЩ ЕЗИК ЗА ОБРАБОТКА НА ДАННИ

Както и при предходното приложение, в обработката на данни често участват не само програмисти, но и математици и статистици и други специалисти от конкретната предметна област (физици, биолози, психолози). Затова лекотата за учене и за четене е

важна. При обработката на данни често задачите са специализирани за даден клиент и поръчка, т.е. кодът не се продава, а се изпълнява веднъж или в някакъв фиксиран срок. За да се отговори на търсенето на такива услуги, използваният език трябва да позволява бързо писане, затова лекотата на писане е от много голяма важност. Обработката на данни в наши дни означава обработка на големи масиви от данни, събрани благодарение на свързаността на устройствата. Това определя важността на производителността за извънредно висока. Преносимостта тук играе сравнително малка роля, свързана най-вече с поддръжката на различи облачни технологии. Популярността на езика също е от малко значение. Наличието на подходящи софтуерни библиотеки, поддържащи различните noSQL решения и математическата обработка на данните е от изключително голямо значение, тъй като това е същината на задачата.
Теглата на критериите са:

- Лесен за учене – средно (5)
- Лесен за четене – средно (5)
- Лесен за писане – много високо (7)
- Производителност – извънредно високо (8)
- Приложимост – не е релевантна (0)
- Преносимост – много ниско (3)
- Популярност – ниско (4)
- Помощни инструменти и компоненти – извънредно високо (9)

Така подредбата е следната:

- Python
- Java
- JavaScript
- C++
- C

3.4. НАЙ-ПОДХОДЯЩ ЕЗИК ЗА УЕБ ПРИЛОЖЕНИЯ

Уеб продължава да бъде доминантната софтуерна платформа, като с усъвършенстването на технологиите все повече и повече софтуерни услуги се изнасят в него. Голямата заетост в този сектор привлича разработчици с различна степен на квалификация, поради което е важно до колко лесен за учене е даден език. Екипите, които работят по такива проекти, също включват разнообразни специалисти – front-end, дизайнери и други, които имат ограничени познания по програмиране, но въпреки това имат взаимодействие с програмния код. Така четенето става много важно за използваните езици. Пазарът, който е динамичен, но голям, изисква бързото генериране на големи количества надежден код, затова лекотата, с която се пише, е от много голямо значение. Уеб технологията като архитектура поставя огромно натоварване върху сървърния софтуер. Той трябва да може да обработва стотици хиляди заявки в секунда и от друга страна да връща отговор в рамките на милисекунди. Това прави производителността извънредно ключова. Преносимостта не е от голямо значение, стига да се поддържат основните сървърни операционни системи. Хетерогенността на разработчиците в тази сфера означава, че е важно да има услужлива общност. И наистина в специализираните форуми въпросите, свързани с уеб технологии, са значителен дял. Пак поради същите причини наличието на удобни инструменти е от извънредно голяма важност.
Теглата на критериите са:

- Лесен за учене – високо (6)
- Лесен за четене – много високо (7)

Годишник Телекомуникации 2019, том 6, с. 1-15

Yearbook Telecommunications 2019, vol. 6, p. 1-15

eISSN 2534-854X https://telecommunications.nbu.bg/bg/godishnik-telekomunikacii

DOI: https://doi.org/10.33919/YTelecomm.19.6.1

- Лесен за писане – много високо (7)
- Производителност – извънредно високо (8)
- Приложимост – не е релевантна (0)
- Преносимост – извънредно ниско (2)
- Популярност – високо (6)
- Помощни инструменти и компоненти – извънредно високо (8)

Вземайки предвид това, крайната подредба е:

- Python
- Java
- JavaScript
- C++
- C

Ако за момент се абстрахираме от Python, подредбата отразява реалната картина на пазара, в който Java (и C# даже повече от него) се използва повече от JavaScript на уеб-сървърите, а C++ и C просто не присъстват на него. Разбира се, в реалността PHP заема огромен дял от този пазар, но поради гореспоменати причини той не участва в нашия списък. А колкото до първото място, то може да се интерпретира само като логичен претендент в един идеален свят, в който изборът на език има обективни технически и други причини. Наистина сравнен със застаряващия PHP, проектиран да бъде смесван с HTML код, Java/C#, които са доста трудни езици, особено вземайки предвид средното ниво на заетите специалисти и JavaScript, и който въпреки огромните усилия, хвърлени от титаните в индустрията да го направят универсален инструмент и да го откъснат от браузера, си остава обвързан с него, Python изглежда идеален кандидат. Той се използва, най вече под формата на фреймуърковете Jango и Flask, но заема несъществен дял, изпреварван дори от Ruby on Rails. Според [6] Python, обаче, заема по-голям дял от JavaScript, което е впечатляващо. Също така известни проекти използват Python, включително Instagram, Google, Facebook, YouTube.

3.5. НАЙ-ПОДХОДЯЩ ЕЗИК ЗА МОБИЛНИ ПРИЛОЖЕНИЯ

При разработката на мобилни приложения значение има не толкова езикът, колкото софтуерната рамка, която се използва. Познаването на спецификите на програмния интерфейс е по-важно отколкото доброто познаване на езика. Затова лекотата на учене има малко значение в тази сфера. Мобилните приложения като цяло не се изработват от особено големи екипи от хора, затова четенето на кода е важно само до колкото това се изисква от добрите софтуерни практики. Комплексните приложни интерфейси и софтуерни рамки означават, че разработчиците трябва да бъдат улеснени от езика, затова лекотата на писане тук е важна. Мобилните устройства разчитат на батерия за източник на енергия, затова производителността е важна от гледна точка на консумацията на енергия. Преносимостта тук трябва да се тълкува по по-различен начин. Въпреки че едва ли има хардуер на планетата, който да няма компилатор за C/C++, то липсата на поддръжка на софтуерната рамка за тях и сравнително ниското им ниво реално ги изключва като вариант. Затова се налага да променим подредбата на езиците по критерия преносимост за тази задача. Много т. нар. крос-платформени приложения се пишат на JavaScript, защото има софтуерни рамки, които позволяват писането на мобилни приложения на този език. Така той отива на първо място по преносимост. Java е на второ място, въпреки че

потенциалът ѝ не е използван, тъй като тя се използва само за Android приложения. На трето място е Python, който може да бъде изключително успешен на мобилния пазар, както бе показано преди години с устройството N900 на Nokia, но поради пазарни причини компаниите предпочитат да разработят собствени езици от класа на Python, вместо да използват него. Именно поради това съществува Swift за iOS и Go за Android. Въпреки почти нулевия пазарен дял, мобилни приложения на C++ могат да се пишат за Android, така че той отива на предпоследна позиция, следван само от родителя си C. Отново поради спецификата на софтуерния стек разработчиците имат нужда от добра общност, затова популярността е от високо значение. И пак по същата причина инструментите (налични библиотеки и компоненти) са от изключително високо значение. Тук отново се налага да нанесем корекции в подредбата на езиците по този критерий, за да изразява наличността не просто на инструменти, а на специфични такива за мобилните приложения. Java несъмнено е на първа позиция, следвана от JavaScript, следвана от C++, следвана от Python и накрая е C.
Теглата на критериите са:

- Лесен за учене – ниско (4)
- Лесен за четене – много ниско (3)
- Лесен за писане – високо (6)
- Производителност – високо (6)
- Приложимост – не е релевантна (0)
- Преносимост – средно (5)
- Популярност – високо (6)
- Помощни инструменти и компоненти – изключително високо (9)

Така класацията за тази дейност на езиците е:

- Java
- JavaScript
- Python
- C++
- C

Според автора, Python има огромен потенциал в тази област, но доминиращите корпорации в областта имат други планове и той едва ли ще заеме сериозен дял в обозримото бъдеще.

3.6. НАЙ-ПОДХОДЯЩ ЕЗИК ЗА ПРИЛОЖЕНИЯ С ГРАФИЧЕН ИНТЕРФЕЙС

Мобилните и уеб приложенията отнемат от дяла на настолните и така последните стават поле само за големи бизнес приложения и за експериментиране и учене от страна на начинаещите програмисти. Поради второто, авторът счита лекотата за учене за относително важна. Големият размер на проектите в тази категория изискват максимално улеснение на поддръжката на кода и колаборацията при писането му. Затова лекотата на четене е много голямо значение, а лекотата с която се пише от още по-голямо. Като се изходи от факта, че приложенията с графичен интерфейс са събитийно-ориентирани, производителността има значение само там, където бизнес-логиката е изчислително интензивна. Преносимостта е важна в аспекта поддръжка на различни ОС, като трите доминиращи настолните компютри са Windows, Mac OS и Linux. Въпреки почти Паретовското разпределение на пазара от трите ОС, софтуерните производители се стремят продуктите им да са налични са всички тях, затова преносимостта тук е от високо значение. Популярността на езика тук играе ниска роля. При големи проекти

Годишник Телекомуникации 2019, том 6, с. 1-15

Yearbook Telecommunications 2019, vol. 6, p. 1-15

eISSN 2534-854X https://telecommunications.nbu.bg/bg/godishnik-telekomunikacii

DOI: https://doi.org/10.33919/YTelecomm.19.6.1

инструментите и компонентите са от много голямо значение. Критерият, обаче, отново трябва да се промени, за да отговаря на спецификата. Най-подходящи библиотеки от петте езика има Java, следван от C++, C (имайки предвид GTK), след това Python и на практика несъществуващия дял на JavaScript.
Теглата на критериите са:

- Лесен за учене – средно (5)
- Лесен за четене – много високо (7)
- Лесен за писане – извънредно високо (8)
- Производителност – средно (5)
- Приложимост – не е релевантна (0)
- Преносимост – високо (6)
- Популярност – ниско (4)
- Помощни инструменти и компоненти – много високо (7)

На базата на тези тегла крайната подредба е:

- Python
- Java
- C++
- C
- JavaScript

Подобно на уеб-приложенията, без първата позиция останалите отразяват реалния пазарен дял на езиците. И отново първото място показва по-скоро потенциала на езика Python. Но традиционно настолните приложения се пишат на компилируеми езици, така че C#, Java и C++ ще продължат да доминират пазара.

3.7. НАЙ-ПОДХОДЯЩ ЕЗИК ЗА ИГРИ

Производството на компютърни игри е огромен и разширяващ се бизнес. В него са заети както програмисти, така и множество други специалисти – дизайнери на нива, графични дизайнери, 3D дизайнери, и други. Самите игри са изключително разнообразни – мобилни, конзолни, настолни и уеб, вариращи много по сложност. Също така, от огромно значение е доколко са използвани готови библиотеки и игрови машини (game engines), което определя до колко е необходима работа на ниско ниво. Все пак компютърните игри са конкретен сегмент от софтуерния пазар, затова авторът счита, че лекотата за учене няма голямо значение. Тъй като обаче проектите могат да бъдат огромни по обем и утилизиращи сложни алгоритми, то до колко четим е кода и колко е лесен за писане са важни аспекти. Независимо от типа игра, производителността неизменно е от изключителна важност. Преносимостта също е важна, тъй като стремежът е играта да поддържа максимален спектър от платформи, но въпреки това поради други съображения целевите платформи се ограничават до една-две. Тук отново се налага да адаптираме подредбата на езиците към спецификата на задачата. Най-преносим е C++, следван от C. Тъй като JavaScript игрите се изпълняват в браузера, а почти всички платформи имат браузери, той е на трето място. Java е на четвърто, а Python на последно. Популярността на езика не е от особено голямо значение, тъй като, както и в някои други специализирани сегменти, по-важно от наличието на добра общност за езика е наличието на такава за специализираните инструменти. И ставайки въпрос за тях, тук отново ще променим

подредбата на езиците, за да отразява спецификата на наличността на иструменти за правене на игри – библиотеки, игрови машини и др. Несъмнено от нашия списък C++ доминира, следван от C. JavaScript е трети поради развитието на WebGL, asm.js и други технологии, позволяващи достъп до ускорено изчертаване на игри, написани на него. Python заема малка ниша, предимно с библиотеката PyGame, която обаче е ограничена до не особено сложни 2D игри. Java като цяло не се използва за игри, така че и наличието на инструменти за правенето им е ограничено.
Теглата на критериите са:

- Лесен за учене – средно (5)
- Лесен за четене – високо (6)
- Лесен за писане – високо (6)
- Производителност – средно (5)
- Приложимост – не е релевантна (0)
- Преносимост – високо (6)
- Популярност – ниско (4)
- Помощни инструменти и компоненти – изключително високо (9)

Вземайки всичко това предвид крайния резултат е:

- C++
- C
- JavaScript
- Python
- Java

На пръв поглед може да се направи заключението, че скриптовите езици няма как да заместят компилируемите, особено тези от ниско ниво, но JavaScript показва, че случаят не е такъв. За целта обаче трябва да се извършат много усилия в посока инструменти, а за момента такива има само за JavaScript, но не и за Python. Все пак неговите качества го правят идеален за сценариен език, което се и ползва от някои игрови машини.

4. ЗАКЛЮЧЕНИЕ/ИЗВОДИ

Целта на тази статия не е да докаже превъзходството на Python във всички сфери на информационните технологии или да аргументира пристрастията на автора към езика, а по-скоро да покаже потенциала на един език, който за своето не дълго съществуване направи огромен възход и има значително поле за развитие. Също така статията цели да покаже, че популярността на даден програмен език в реалния свят не зависи (само) от качествата му, а и от пазарни механизми и корпоративни съображения. Това разбира се е известно на всеки софтуерен и хардуерен инженер, но често се пропуска в подобен род дискусии.

ЛИТЕРАТУРНИ ИЗТОЧНИЦИ (REFERENCES):

1. *Cython* [online]. [viewed 18 December 2019]. Available from: https://github.com/cython/cython/wiki/FAQ#is-cython-a-python-implementation
2. GALL, Richard. Why Python is a crucial part of the DevOps toolchain. *Jaxenter* [online]. 31 August 2017 [viewed 17 December 2019]. Available from: https://jaxenter.com/python-crucial-devops-toolchain-136694.html
3. TIOBE Index for August 2019. *TIOBE* [online]. [viewed 18 December 2019]. Available from: https://www.tiobe.com/tiobe-index//
4. *Python Package Index (PyPI)* [online]. [viewed 17 December 2019]. Available from: https://pypi.org/project/blist/
5. *Pylint* [online]. [viewed 17 December 2019]. Available from: https://www.pylint.org/

Годишник Телекомуникации 2019, том 6, с. 1-15

Yearbook Telecommunications 2019, vol. 6, p. 1-15

eISSN 2534-854X https://telecommunications.nbu.bg/bg/godishnik-telekomunikacii

DOI: https://doi.org/10.33919/YTelecomm.19.6.1

6. Usage of server-side programming languages for websites. *W3Techs* [online]. [viewed 18 December 2019]. Available from: https://w3techs.com/technologies/overview/programming_language/all

7. ПЕТРОВ, Георги, Филип АНДОНОВ и Тодор ДАЧЕВ. *Разработка на приложения с отворени хардуерни платформи*. София: Авангард Прима, 2015. ISBN 978-619-160-506-4.; PETROV, Georgi, Filip ANDONOV i Todor DACHEV. Razrabotka na prilozheniya s otvoreni harduerni platformi. Sofiya: Avangard Prima, 2015. ISBN 978-619-160-506-4.

Информация за автора/ите:
гл. ас. д-р Филип Богданов Андонов, НБУ, София, бул. Монтевидео 21, fandonov@nbu.bg, 02 8110610

Contacts:
chief assistant professor Filip Bogdanov Andonov, Ph. D., NBU, Montevideo blvrd 21, Sofia, Bulgaria, fandonov@nbu.bg, +359 2 811 06 10

Дата на постъпване на ръкописа (Date of receipt of the manuscript): 03.09. 2019.
Дата на приемане за публикуване (Date of adoption for publication): 27.09. 2019.