Razvoj programskog jezika Julia

Seminarski rad u okviru kursa
Dizajn programskih jezika

Matematički fakultet

Kristina Jovanovic
Danilo Gligorovic
kiki95sd.kj@gmail.com
danilogligorovic@hotmail.com

Decembar 2019

Sažetak

U ovom radu prikazuju se osnove razvoja programskog jezika Julia. Opisani su jezici koji su najviše uticali na njegov razvoj: Lisp, Python, R, Ruby, Lua i drugi. Za svaki jezik date su osnovne informacije kao i način na koji je taj jezik uticao na osobine i razvoj jezika Julia. Dato je razvojno stablo sa jezicima koji su najviše uticali na jezik Julia.

Sadržaj

1 Uvod

"Mi smo snage Matlab korisnika. Neki od nas su hakeri iz Lispa. Neki su korisnici Pythona, Ruby-ja ili Perla. Neki od nas su koristili programski jezik Mathematica pre nego što je počela da im raste brada. Napravili smo više R "parcela"nego što bi to trebalo normalnom čoveku. C je naš programski jezik na pustom ostrvu."

Julija [2] je dinamički i programski jezik visokog nivoa dizajniran da odgovori na zahteve razvijenih numeričkih i naučnih izračunavanja, ali je takođe efikasan za efektivno opšte programiranje, veb upotrebu, može se koristiti i za specifikaciju programskog jezika Karakteristike Julija programskog jezika obuhvatuju tipski sistem u paketu sa parametarskim tipskim sistemom u obliku celokupnog dinamičnog programskog jezika, kao i u obliku posebnih paketa u obliku sopstvenog jezgra programske paradigme. Omogućava istovremeno, paralelno i distribuirano izračunavanje, i direktno pozivanje biblioteka programskih jezika C i Fortran, bez posebnog koda. Julija je sakupljač smeća, program koji koristi željenu procenu i u sebi sadrži efikasne biblioteke za izračunavanja, linearnu algebru, nasumičnu generaciju brojeva, brze Furijerove transformacije i upoređivanje regularnih izraza.

Slika 1: Logo programskog jezika Julia

2 Osnovno o Julia jeziku

Rad na programskom jeziku Julia započeli su 2009. godine Jeff Bezanson, Stefan Karpinski, Viral B. Shah i Alan Edelman, koji su započeli stvaranje besplatnog jezika koji je brz. Programski jezik Julia je prvi put objavljen u maju 2012. godine. Njegovo jezgro, kao i veliki broj dodatnih paketa, razvili su Jeff Bezanson, Stefan Karpinski, Viral Shah i Alan Edelman. Njihova želja za novim programskim jezikom je proizašla iz činjenica da su većina alata, koje su koristili, imali uz svoje prednosti i vidljive nedostatke. Dok je jedne krasila brzina, drugi su imali jasnoću i jednostavnost korištenja. Najveći nedostatak je bila upravo činjenica da nisu imali jedinstveni alat s kojim bi radili, nego više njih. Juliu karakterišu: mogućnost definisanja funkcija za različite tipove podataka (multiple dispatch), sistem dinamičkih tipova podataka, LLVM JIT (low-level virtual machime just in time) prevodilac, ugradjen menadžer paketa, makro naredbe, mogućnost pozivanja Python i C funkcija, moćne shell mogućnosti i upravljanje drugim procesima, paralelno i distribuirano programiranje, mogućnost definisanja tipova podataka, elegantna konverzija i promocija numeričkih i ostalih tipova podataka, podrška za Unicode standard i MIT

Listing 1: Julia

```
julia> p(x) = 2x^2 + 1; f(x, y) = 1 + 2p(x)y
julia> println("Hello world!")
Hello world!
```

licenca.

Julia sadrži "garbage"kolektor, uključuje biblioteke za izračunavanje s pokretnim zarezom, linearnu algebru, generisanje slučajnih brojeva i redovno podudaranje izraza. Dostupno je mnogo biblioteka, uključujući neke koje su prethodno bile povezane sa jezikom Julia, a sada su odvojene. Sledeći framgent predstavlja jednostavan primer REPL-a [8]:

"Multiple dispatch"je sistem koji omogućava definisanje funkcije s istim imenom, a koja će se nositi s različitim tipovima podataka koje dobija kao ulazne promenljive. Takav sistem u kombinaciji s mogućnošću definisanja vlastitih tipova podataka, uz ostale karakteristike upravljanja podacima, makro naredbe i sintaksu sličnu Python-ovoj, dao je jeziku Julia željenu jednostavnost. A JIT kompajler (prevođenje se događa dok se program izvršava) kombinuje brzinu izvođenja prevedenog koda i fleksibilnost interpretera, čime se dobija brzina slična brzini onim programima napisanih u C-u. Julia podržava paralelno izračunavanje, bilo na više jezgara na jednom procesoru ili distribuirano na mreži na više računara. Za Juliu je još bitno naglasiti da je vrlo jednostavna za učenje, s obzirom na sličnost sintakse Pythonu, ali i ostalim jezicima koji su joj uzor, kao što je Matlab. Julia je besplatna i otvorenog koda, čime se podstiče njen dalji razvoj. Julia je dostupna na sve tri veće platforme: Windows, Linux i MacOS. Dolazi u x86 i x64 verzijama.

Ključna reč **ccall** jezika Julia se koristi za pozivanje C-ove (ili Fortranove) zajedničke biblioteke funkcija. Julia ima pakete koji podržavaju jezike za označavanje kao što su, HTML (takođe za HTTP), XML, JSON itd.

	Julia	Python	R	Matlab
fib	2.11	77.76	533.52	26.89
parse$_i nt$	1.45	17.02	45.75	802.52
quicksort	1.15	32.89	264.54	4.92
mandel	0.79	15.32	53.16	7.58
pi_sum	1.00	21.99	9.56	1.00

Tablica: Poređenje brzine izvođenja raznih algoritama s C-om.

3 Poredjenje sa programskim jezicima

Glavna prednost programskog jezika Julia pred konkurentskim jezicima kao što su Python, R i Matlab je brzina. Testiranja koje su obavili programeri koji su napravili jezik Julia za različite algoritme pokazuju re-

lativnu brzinu u odnosu na C (brzina izvođenja u C-u je 1.0). Testovi su svi izvođeni na istom računaru, bez podrške za paralelno izračunavanje, te je svaki jezik izvodio isti algoritam. Julia se u svim testovima pokazala kao dosledna, a u jednom čak i brža od C-a. Ostali jezici su u većini slučajeva bili dosta sporiji i od C-a i od jezika Julia.

Julia se pokazala bržom od Pythona koji je postao gotovo standardan jezik zahvaljujući svojim modulima NumPy i SciPy kao i činjenici da je kao i Julia jezik "open source"koda. Julia je po sintaksi slična Pythonu, što dosta olakšava prelazak s Pythona, s kojim se susreo gotovo svaki moderni programer.

3.1 Lisp

Lisp [3] je familija programskih jezika sa dugom istorijom. Lisp je drugi najstariji viši programski jezik koji se i danas veoma koristi. Jedino je Fortran stariji (godinu dana). Danas postoji veliki broj dijalekata Lispa, a najpoznatiji među njima su Common Lisp i Scheme.

Ime LISP je nastalo od "LISt Processor". Povezane liste su jedan od glavnih tipova podataka. Lisp je primer funkcionalnog programskog jezika.Lisp je projektovao Džon Makarti 1958. godine. Lisp je prvi put implementiran od strane Stiva Rasela na IBM 704 računaru.

Lisp je jezik koji se oslanja na izraze. Kada se izraz evaluira, on vraća vrednost, koja se može iskoristiti u drugim izrazima. Svaka vrednost može biti bilo koji tip podataka. U Lisp-u se navode dva tipa sintakse: S-izrazi (simbolički izrazi), koji oslikavaju unutrašnju reprezentaciju i koda i podataka, i M-izrazi (meta izrazi), koji izražavaju funkcije S-izraza. Skoro svi Lisp jezici danas koriste S-izraze za kod i podatke. Ono što je Lisp odmah odvojilo od drugih porodica jezika, jeste upotreba zagrada. Sintaksa koja koristi S-izraze je zaslužna za lakše korišćenje programskog jezika Lisp.

Sve funkcije su generičke i koriste "multiple dispatch". Liste argumenata ne moraju koristiti isti obrazac. Opcionalni argumenti i argumenti ključnih reči se koriste na različiti način. Nejasnoće metoda se ne rešavaju kao u uobičajenom sistemu objekata Lisp.

Funkcionalni stil programiranja u potpunosti podržava jezik, uključujuci i zatvaranja, ali nije uvek najpametnije rešenje za jezik Julia. Neka rešenja ce možda biti potrebna za performanse prilikom modifikacije zarobljenih promenljivih.

Makroi počinju sa "@"i nisu tako neprimetno integrisani u jezik kao što je Lisp. Upotreba makroa nije tako rasprostranjena kao u Lispu.

3.2 Matlab

Matlab [5] je okruženje za numeričke proračune i programski jezik četvrte generacije koji je razvila firma -MathWorks. MATLAB omogućava lako manipulisanje matricama, prikazivanje funkcija i fitovanje, implementaciju algoritama, stvaranje grafičkog korisničkog interfejsa kao i povezivanje sa programima pisanim u drugim jezicima među kojima su C, C++, Java, Fortran i Python. MATLAB je nastao kao skraćenica za "MATrix LABoratory"("laboratorija za matrice").

Julia je jezik sa očiglednim, poznatim matematičkim notacijama, snažan za linearnu algebru kao Matlab. U Matlab-u najviši levi objekat ima prednost, ali klase koje su definisane od strane korisnika imaju prednost nad ugrađenim klasama. Matlab takođe ima mehanizam za kreiranje prilagođene hijerarhije. Julia uglavnom izbegava pojam „ugrađenog“

umesto „korisničkog definisanog“, fokusirajuci se na metodu koja treba da se primenjuje na osnovu kombinacije tipova i dobijanja visokih performansi kao nusprodukta.

3.3 C

C [1] je proceduralni programski jezik, nastao 1972. godine. Autor jezika je Denis Riči, a nastao je u istraživačkom centru Bell Laboratories u Nju Džerziju za potrebe operativnog sistema UNIX. Programski jezik C je jezik opšte namene. Iako je razvijan kao "jezik za sistemsko programiranje", podjednako dobro se koristi za pisanje značajnih programa u različitim oblastima. C je takođe veoma uticao na mnoge druge popularne programe, posebno na C++, koji je originalno razvijan kao nadgradnja C-a.

Organizacione celine C-a su funkcije. Početna tačka izvršavanja programa je tzv. "glavna funkcija", a celokupan program se organizuje u određeni broj drugih funkcija, u zavisnosti od potrebe. Funkcije se posmatraju kao odvojene, izolovane celine čime se postiže modularnost koda koji je kao takav lak za održavanje i dalje razvijanje.

Svaka funkcija ima svoj imenski prostor i nema direktan pristup promenljivama drugih funkcija. Indirektan pristup promenljivama drugih funkcija je obezbeđen mehanizmom pokazivača i argumenata funkcije.
Telo funkcije je ograničeno vitičastim zagradama, kako bi joj se označili početak i kraj. Unutar funkcija delovi koda mogu biti odvojeni u funkcijske podblokove.

Program se izvršava počev od prve linije koda funkcije main. Ona može pozivati druge funkcije ili samu sebe. Po završetku funkcije main, čitav program se završava i oslobađa iz memorije. Uz standardni C se distribuiraju i standardne biblioteke. To su datoteke koje sadrže mnoštvo korisnih alata koji se koriste u svakodnevnom radu. Programer može da uključi jednu ili više ovih biblioteka u svoj program, a može i sam da piše biblioteke za svoje potrebe. Svaki C program se sastoji od funkcija i promenljivih. main funkcija je posebna — svaki program mora sadržati ovu funkciju jer se program izvršava od početka main funkcije.

Prva linija našeg programa je pretprocesorska direktiva koja govori da treba usvojiti informaciju o standardnoj ulazno-izlaznoj biblioteci. Program se izvršava instrukciju po instrukciju, počevši od prve a završavajući se poslednjom instrukcijom funkcije main. Program se može završiti i ranije, koristeći ključnu reč **return** ili bibliotečku funkciju **exit**. Pored imperativnih instrukcija, u programskom jeziku C postoje i instrukcije koje kontrolišu tok programa, tzv. kontrolne strukture. One se dele na grananja i petlje.

U programskom jeziku C, tipovi promenljivih se dele na sledeće: elementarni tipovi, nizovi, tipovi koje definiše korisnik i pokazivačke promenljive.

Programski jezik C koristi biblioteke kao osnovni način proširivanja svoje funkcionalnosti. U C-u, biblioteka je skup funkcija. Svaka biblioteka obično ima zaglavlje , koje sadrži prototipove funkcija, sadržanih u biblioteci, koje može da koristi program, i deklaraciju specijalnih tipova podataka i makroa koje koriste ove funkcije. Da bi program koristio biblioteku, mora da uključi zaglavlje biblioteke, i biblioteka mora biti povezana sa programom.

Dizajn zasnovan na generičkim funkcijama i sistemu bogatog tipa istovremeno omogućava ekspresivan programski model i uspešno zaključivanje

tipa, što dovodi do dobrih performansi za širok spektar programa. To omogućava da se veliki deo biblioteka napiše u samom jeziku Julia, a istovremeno uključuje najbolje C biblioteke.

U programskom jeziku Julia moguće je implementirati sve osnovne funkcionalnosti - nikada nemojte prisiljavati programera da koristi C. Julia rešava dva jezička problema. Osnovna funkcionalnost mora biti brza: aritmetika celog broja, za petlje, rekurziju, operacije s pokretnim zarezom, pozivanje C funkcija i manipulisanje strukturama sličnim C-ovim. Iako ove funkcije nisu važne samo za numeričke programe, bez njih sigurno ne možete upisati brzi numerički kod.

3.4 Perl

Perl je slobodni, nezavisni od platforme i interpretirani programski jezik kojeg je razvio Amerikanac Leri Vol 1987. godine. Nastao je kao sinteza programskog jezika C, nekih komandi operativnog sistema Unix i drugih elemenata.

Skraćenica PERL potiče od "Practical Extraction and Report Language"koja precizno objašnjava najjače osobine Perla - Practical za praktičnost tj. brže pisanje programa nego u programskom jeziku C, Extraction za izdvajanje i analizu datoteka i podataka, Report za generisanje izlaznih podataka i Language za programski jezik - iako ga neki svrstavaju samo u grupu skript-jezika i time neopravdano omalovažavaju. Perl ima sintaksu vrlo sličnu onoj u C++-u[1].

Programski jezik Julia je pogodan za obradu stringova kao Perl.

3.5 Python

Programski jezik **Python** [6] nastao je početkom devedesetih godina prošlog veka. Njegov autor je Gvido van Rosum. Programi u jeziku Python se uglavnom interpretiraju. Interpretatori i standardne biblioteke modula se stalno razvijaju i prenose na veliki broj različitih platformi. Glavne podržane platforme su Linux, Mac OS, Microsoft Windows i Java.

Podaci u programskom jeziku Python su predstavljeni objektima. Svaki podatak je predstavljen objektom ili relacijom među objektima. Prevedene funkcije, metode i neki drugi elementi jezika Python takođe su predstavljeni objektima tokom izvršavanja programa.

Tip podatka u programskom jeziku Python nije vezan za promenljivu. Svakoj promenljivoj tokom izvršavanja programa može da bude dodeljena vrednost bilo kog tipa kao i da ta vrednost bude zamenjena drugom vrednošću različitog tipa. Tip podatka vezan je za vrednost koju sadrži promenljiva. Sve vrednosti promenljivih su objekti.

Svaki objekat sadrži tip objekta i njegovu vrednost. Jednom kreirani objekat ne može da menja tip, dok vrednost nekih objekata može da bude promenjena. Promenljivost objekata određena je njihovim tipom.

Postoje:

- nepromenljivi objekti(brojevi, niske i n-torke)

[1] Jezik **C++** se razlikuje od običnog C-a pre svega podrškom objektno-orijentisanom programiranju. Ali, u njemu ima i niz novih mogućnosti, koje nisu objektnog karaktera, zbog kojih je pisanje programa koji nisu objektne prirode udobnije realizovati u C++-u nego u C-u. C++ je danas nesporno jedan od najmoćnijih, ali i najkompleksniji programski jezik. Zbog kompaktnosti programa, brzine izvršavanja i prenosivosti predstavlja nezamenljiv alat svakog profesionalca. Zbog ovih osobina zauzima jedno od dominantnih mesta u svetu profesionalnog programiranja.

- promenljivi objekti(rečnici i liste)

Za programski jezik Python razvijen je veliki broj standardnih modula koji omogućavaju efikasan rad u mnogim oblastima. Standardna biblioteka modula omogućava pisanje programa vezanih za Internet podržavajući veliki broj standardnih formata i protokola. Postoje moduli za kreiranje grafičkog korisničkog interfejsa, vezu ka relacionim bazama podataka, za aritmetiku sa proizvoljnim željenim brojem decimala, za rad na leksičkoj analizi primenom regularnih izraza, kao i standardni moduli za mnoge druge poslove.

Objekti ugrađeni u jezik:

Python poseduje jedan broj tipova objekata koji su ugrađeni u jezik. Jezik može da se proširi dodatnim tipovima preko modula.

Sledeći spisak sadrži neke od tipova objekata koji su ugrađeni u programski jezik Python: - Brojevi
- nizovi
- preslikavanja
- izvršivi tipovi(ovo su objekti na koje može da se primeni poziv funkcije)

Julia može pozivati Python biblioteke. Takođe je moguće međusobno povezivanje sa Python kodom pomoću PyCall biblioteke, pa čak i deljenje podataka između Python-a i Julia-e.

I Python i Julia mogu paralelno da rade. Međutim, Pythonove metode za paralelizaciju operacija često zahtevaju da se podaci serializuju i deserializuju između niti ili čvorova, dok je paralelizacija programskog jezika Julia bolja. Sintaksa paralelizacije jezika Julia je lakša od Pythonovih.

3.6 Lua

Lua [4] je jednostavan, imperativni i funkcionalni programski jezik, dizajniran kao skript jezik sa proširivom semantikom kao primarnim ciljem. Samo ime jezika potiče od portugalske reči **Lua** što znači mesec. Jezik je kreiran 1993. godine. Lua programi se ne interpretiraju direktno iz tekstualne Lua datoteke, već se kompajliraju na bajtkod koji se potom izvršava na Lua virtuelnoj mašini. Korišćenjem minimalnog skupa tipova podataka, Lua pokušava da balansira između veličine i moći.

Kao i većina skript jezika, Lua je dinamički tipiziran programski jezik. Postoji osam osnovnih tipova podataka: null, boolean, number, string, userdata, function, thread i table. Funkcija type vraća tip promenljive čija vrednost joj je prosleđena. Funkcije su glavni mehanizam za apstrahovanje naredbi i izraza u programskom jeziku Lua. Funkcije mogu da izvršavaju specifične zadatke ili da izračunavaju i vraćaju vrednosti. U prvom slučaju funkciju koristimo kao naredbu, a u drugom slučaju je koristimo kao izraz. U oba slučaja argumenti funkcije se nalaze unutar zagrada, zagrade se moraju pisati i kada funkcija nema arugmenata koje bi dobila. Definicija funkcije u jeziku Lua ima sličnu sintaksu kao i u drugim programskim jezicima.

Julia i Lua su sjajne u pogledu jednostavnosti i imaju sličnu sintaksu. Znači da se dobija puno funkcionalnosti izvan okvira za obavljanje svih vrsta stvari. Jednostavni su, ekspresivni i veoma visokih performansi.

3.7 R

R [7] je programski jezik i programsko okruženje za statističke izračune i grafike. R je implementacija programskog jezika S[2] pomešanog sa programskim jezikom Scheme. R su stvorili Ros Ihaka i Robert Džentlmen na Aukland univerzitetu, Novi Zeland.

Projekat je nastao 1992. godine, prva verzija je izašla 1995. godine dok je stabilna verzija izašla 2000. godine. R obezbeđuje širok izbor statističkih(linearnih i nelinearnih modela, klasičnih statističkih testova, analiza vremenskih serija, klasifikacija, klasteri, i ostalo) i grafičkih tehnika. R može biti proširen, kroz pakete obezbeđene od korisnika, za specifične funkcije ili specifične oblasti proučavanja. Zbog svog S nasleđa, R ima bolju podršku za Objektno-orijentisano programiranje nego ostali statistički programski jezici. Sledeća prednost R-a su njegove grafičke mogućnosti, koje obezbeđuju grafike kvaliteta dovoljno dobrog za publikovanje koji uključuju matematičke simbole. R ima sopstveni LaTeX format dokumenata, koji se koristi za predstavljanje dokumentacije, preko interneta u brojnim formatima ili kao štampana kopija.

Programski jezik Julia teži da bude jednostavna za statistiku kao R. 'Vektorizacijski jezici' poput R-a skrivaju svoje petlje i čitave operacije, ali oni su i dalje tu. Julia u potpunosti uklanja ovo razdvajanje, dopuštajuci kodu na visokom nivou da 'samo napiše petlju' ako se desi da je to najbolji način za rešavanje problema.

3.8 Ruby

Ruby [9] je objektno-orijentisani programski jezik. U sebi kombinuje sintaksu inspirisanu jezicima Perl i Ada, sa objektno-orijentisanim osobinama nalik jeziku Smalltalk, a deli i neke osobine sa jezicima Python, Lisp, Dylan i CLU. Rubi je jednoprolazni interpretirani jezik. Njegova glavna implementacija je slobodni softver pod licencom "open source"koda. Yukihiro Matsumoto je sa razvojem ovog jezika počeo u februaru 1993. godine. Prvi put je objavljen 1995. godine.

OSOBINE:
Jednostavna i čitljiva sintaksa, "čisto" objektno-orijentisani jezik, Dinamična promena imena i nadogradnja klasa prilikom izvršenja programa, Iteratori, Obrada izuzetaka, automatsko oslobađanje nepotrebno zauzete memorije (garbage collection), jedinstven interfejs za pristup bazama podataka, mogućnost i funkcionalnog i proceduralnog programiranja.

Rubi je interpretirani programski jezik, što znači da se izvorni kod prevodi u kod razumljiv računaru prilikom svakog izvršavanja programa. Nedostatak interpretiranih programskih jezika je to da su sporiji nego kompajlirani programski jezici kao što je na primer C. Prednost je njihova veća fleksibilnost i kraće vreme izrade programa. Rubi je potpuno objektno-orijentisan programski jezik, što znači da se programski problemi rešavaju definisanjem klasa, a iz klasa nastaju pojedinačni objekti. Jedna od prednosti objektno-orijentisanog pristupa je to da on olakšava rad na velikim

[2] **S** [10] je jezik visokog nivoa i okruženje za analizu podataka i grafiku. 1998. godine, Udruženje za računarske mašine (ACM) je dodelilo nagradu za softverski sistem Johnu M. Chambersu, glavnom dizajneru kompanije S, za S sistem, koji je zauvek izmenio način na koji ljudi analiziraju i manipulišu podacima. S je elegantan, široko prihvacen i trajan softverski sistem, sa konceptualnim integritetom, zahvaljujuci John Chambers-u.

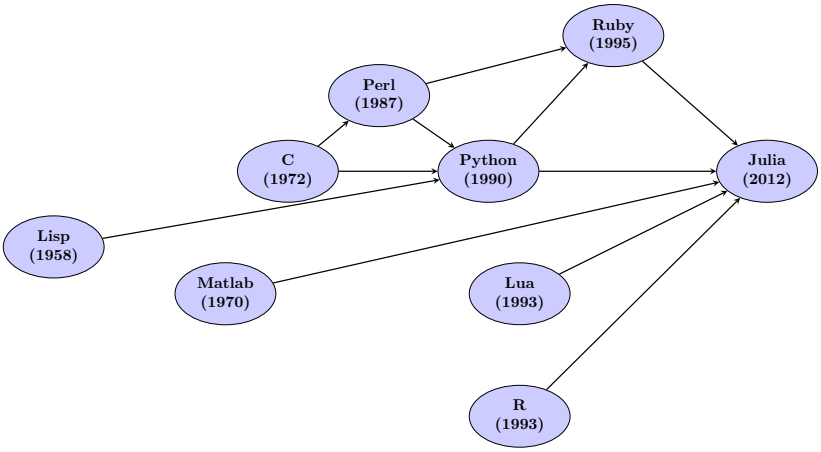
projektima jer je zadatak moguće podeliti na više programera, od kojih svaki može posebno raditi na jednoj klasi ili na više njih.

Julia je kreirana od početka do kraja kako bi iskoristila moderne tehnike za efikasno izvršavanje dinamičkih jezika. Kao rezultat toga, Julia ima performanse statički kompajliranog jezika, istovremeno pružajuci interaktivno dinamičko ponašanje i produktivnost poput jezika Ruby.

Za razliku od jezika Ruby, budući da može obeležiti tipove na kojima funkcija radi, mogu se preopteretiti imena funkcija, tako da se može koristiti isto ime funkcije za mnoge tipove podataka. Tako se može zadržati jednostavna opisana imena funkcija i ne moraju se izmišljati veštačka imena funkcija da bi se odvojili od tipa na kome rade.

4 Razvojno stablo

Julia pruža lakoću i ekspresivnost za numeričko računanje visokog nivoa, na isti način kao što su jezici poput R-a, MATLAB-a i Python-a, ali takođe podržava opšte programiranje. Julia pozajmljuje mnogo od popularnih dinamičkih jezika, uključujući i Lisp, Perl, Python, Lua i Ruby.



5 Zaključak

U ovom radu ukratko su predstavljene osnove razvoja programskog jezika Julia. Prikazano je i elementarno stablo jezika Julia sa jezicima koji su uticali na razvoj ovog jezika. Julia se pokazala kao programski jezik jednostavan za učenje onima koji već poznaju jezike kao što su Python i Matlab. Brzina jezika Julia je uporediva s C-om za pravilno pisane programe. Za ostale informacije poželjno je posetiti zvanični sajt programskog jezika Julia [2].

Literatura

[1] C. Zvanična stranica programskog jezika C. https://www.cprogramming.com/.

[2] Julia. Zvanična stranica programskog jezika Julia. https://julialang.org/.

[3] Lisp. Lisp. https://en.wikipedia.org/wiki/Lisp_(programming_language).

[4] Lua. Zvanična stranica programskog jezika Lua. https://www.lua.org/.

[5] Matlab. Zvanična stranica programskog jezika Matlab. https://www.mathworks.com/products/matlab.html.

[6] Python. Zvanična stranica programskog jezika Python. https://www.python.org/.

[7] R. Zvanična stranica programskog jezika R. https://www.r-project.org/.

[8] Repl. Repl. https://docs.julialang.org/en/latest/manual/getting-started/.

[9] Ruby. Zvanična stranica programskog jezika Ruby. https://www.ruby-lang.org/en/.

[10] S. Zvanična stranica programskog jezika S. https://cran.r-project.org/doc/FAQ/R-FAQ.html#Why-is-R-named-R_003f.