SVEUČILIŠTE U SPLITU  
FAKULTET ELEKTROTEHNIKE, STROJARSTVA I BRODOGRADNJE

SEMINARSKI RAD

USPOREDBA PERFORMANSI PROGRAMSKIH JEZIKA C I PYTHON

Luka Dorić, Nikolina Duvnjak

Martin Josipović, Jakov Kovač

Split, Ožujak 2024.

Tablica 1‑1 Doprinosi koautora rada

|  |  |
| --- | --- |
| **Koautor** | **Kratki opis doprinosa koautora u izradi rada** |
| Luka Dorić |  |
| Nikolina Duvnjak |  |
| Martin Josipović |  |
| Jakov Kovač |  |

**IZJAVA**

Ovom izjavom potvrđujem da sam završni rad s naslovom „**Usporedba performansi programskih jezika C i Python**“ pod mentorstvom prof. dr. sc. Stipe Čelara pisao samostalno, primijenivši znanja i vještine stečene tijekom studiranja na Fakultetu elektrotehnike, strojarstva i brodogradnje, kao i metodologiju znanstveno-istraživačkog rada te uz korištenje literature koja je navedena u radu. Spoznaje, stavove, zaključke, teorije i zakonitosti drugih autora koje sam izravno ili parafrazirajući naveo u završnom radu citirao sam i povezao s korištenim bibliografskim jedinicama.

Student

Luka D., Nikolina D., Martin J., Jakov K.

**SADRŽAJ**

[1 UVOD 1](#_Toc166314966)

[2 PROGRAMSKI JEZIK C 2](#_Toc166314967)

[2.1 Povijest i karakteristike jezika C 2](#_Toc166314968)

[2.1.1 Povijest jezika C 2](#_Toc166314969)

[2.2 Struktura i sintaksa jezika C 3](#_Toc166314970)

[2.3 Prednosti i mane jezika C 5](#_Toc166314971)

[2.3.1 Prednosti jezika C 5](#_Toc166314972)

[2.3.2 Nedostatci jezika C 6](#_Toc166314973)

[2.4 Primjeri koda u jeziku C 6](#_Toc166314974)

[3 PROGRAMSKI JEZIK PYTHON 9](#_Toc166314975)

[3.1 Povijest i karakteristike jezika Python 9](#_Toc166314976)

[3.1.1 Povijest jezika Python 9](#_Toc166314977)

[3.1.2 Karakteristike Pythona 9](#_Toc166314978)

[3.2 Struktura i sintaksa jezika Python 10](#_Toc166314979)

[3.3 Prednosti i mane jezika Python 12](#_Toc166314980)

[3.3.1 Prednosti jezika Python 12](#_Toc166314981)

[3.3.2 Nedostatci jezika Python 12](#_Toc166314982)

[3.4 Primjeri koda u jeziku Python 13](#_Toc166314983)

[4 USPOREDBA PERFORMANSI 15](#_Toc166314984)

[4.1 Metodologija testiranja performansi 15](#_Toc166314985)

[4.1.1 Testne funkcije 15](#_Toc166314986)

[4.1.2 Proces testiranja 15](#_Toc166314987)

[4.2 Proces mjerenja vremena izvođenja i potrošnje memorije 16](#_Toc166314988)

[4.2.1 Mjerenje vremena izvođenja i potrošnje memorije u C-u 16](#_Toc166314989)

[4.2.2 Mjerenje vremena izvođenja i potrošnje memorije u Python-u 17](#_Toc166314990)

[5 REZULTATI TESTIRANJA 18](#_Toc166314991)

[5.1 Analiza rezultata testiranja performansi 18](#_Toc166314992)

[5.2 Vizualni prikaz rezultata 18](#_Toc166314993)

[5.3 Tumačenje dobivenih rezultata 18](#_Toc166314994)

[6 ZAKLJUČAK 19](#_Toc166314995)

[LITERATURA 20](#_Toc166314996)

[PRILOZI 22](#_Toc166314997)

[Kazalo slika, tablica i kodova 22](#_Toc166314998)

[Kazalo slika 22](#_Toc166314999)

[Kazalo tablica 22](#_Toc166315000)

[Kazalo kodova 22](#_Toc166315001)

[Popis oznaka i kratica 22](#_Toc166315002)

[SAŽETAK I KLJUČNE RIJEČI 23](#_Toc166315003)

# UVOD

U današnjem digitalnom dobu, programski jezici su neizostavan alat za rješavanje različitih problema i izgradnju raznovrsnih aplikacija. S mnoštvom dostupnih jezika, izbor onog pravog za određeni zadatak može biti izazovan. U ovom radu usredotočili smo se na usporedbu performansi dva popularna programska jezika: C i Python.

C je imperativni programski jezik s dugom i bogatom poviješću, poznat po svojoj brzini i efikasnosti. Često se koristi za izradu sustavskog softvera, ugrađenih sustava i aplikacija gdje je brzina izvršavanja od ključne važnosti.

Python je s druge strane interpretirani, dinamički tipizirani jezik koji se ističe jednostavnom sintaksom i čitljivošću koda. Popularan je u širokom spektru područja, uključujući znanost podataka, strojno učenje, web programiranje i automatizaciju zadataka.

Izbor programskog jezika je poput izbora alata u radionici. Svaki jezik ima svoje prednosti i mane, a izbor idealnog alata ovisi o zadatku koji želimo obaviti. Usporedba različitih jezika pomaže nam da razumijemo njihove specifičnosti i pronađemo onaj koji će nam olakšati i optimizirati rad.

Usporedbom C-a i Pythona, dva popularna jezika s različitim karakteristikama, dobivamo uvid u utjecaj različitih faktora na performanse programa. Spoznajemo prednosti i mane oba jezika, te ih možemo procijeniti u kontekstu specifičnih zadataka.

Ovaj rad je imao za cilj dati sveobuhvatnu usporedbu C-a i Pythona. Kroz kratak opis jezika, izradu primjera koda i definiranje metodologije testiranja, dobili smo jasnu sliku o njihovim performansama. Usporedba brzine izvršavanja koda i potrošnje memorije pomogla nam je da procijenimo praktičnu primjenu oba jezika.

Analiza rezultata testiranja i rasprava o prednostima i manama C-a i Pythona donijele su nam cjeloviti uvid u njihove specifičnosti. Osim toga, razmatranje različitih faktora koji su utjecali na performanse programa dodatno je obogatilo naše razumijevanje ova dva jezika.

Zaključak rada sažima dobivene rezultate i daje praktične smjernice za izbor optimalnog programskog jezika u različitim kontekstima. Očekujemo da će ova usporedba biti korisna svim programerima koji žele proširiti svoje znanje i vještine te pronaći idealan alat za rješavanje različitih problema.

# PROGRAMSKI JEZIK C

## Povijest i karakteristike jezika C

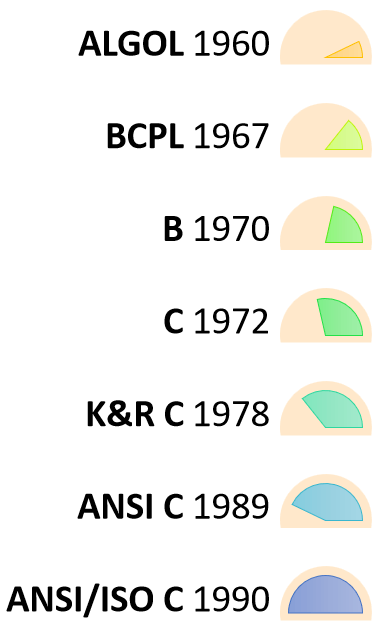
Programski jezik C danas je jedan od najpopularnijih i najutjecajnijih jezika u svijetu. Njegova fleksibilnost i jednostavnost korištenja čine ga idealnim za izradu širokog spektra aplikacija, operativnih sustava (uključujući Windows) i složenih programa poput Oracle baze podataka, Git-a i Pythona. C se često naziva “temeljni jezik” I “bazom programiranja” jer je poznavanje C-a ključno za razumijevanje i učenje drugih programskih jezika koji koriste njegov koncept.

### Povijest jezika C

* **1960-e:** ALGOL se pojavljuje kao "otac" programskih jezika, s fokusom na strukturirano programiranje, te se uveliko koristi u Europi.
* **1967:** Martin Richards dizajnira BCPL s fokusom na pisanje sistemskog softvera.
* **1970:** Ken Thompson kreira jezik B, koji uključuje značajke BCPL-a i postaje dio UNIX operativnog sustava.
* **1972:** Dennis Ritchie u Bell Laboratoriesu razvija C jezik, integrirajući koncepte iz ALGOL-a, BCPL-a i B-a te dodajući inovativne ideje.

C je u početku bio ograničen na UNIX, ali se s vremenom proširio na mnoštvo različitih operativnih sustava i hardverskih platformi. Njegov rast doveo je do problema s kompatibilnošću starijih i novijih verzija. Kako bi se osigurao standard, Američki nacionalni institut za standarde (ANSI) definirao je 1989. godine komercijalni standard za C jezik. Standard je 1990. godine odobrila i Međunarodna organizacija za standardizaciju (ISO), što je C-u donijelo nadimak "ANSI C". [1]

Slika 2‑1 - Logo C-a [8]



Slika 2‑2 – Povijest C-a [1]

## Struktura i sintaksa jezika C

C program se oslanja na nekoliko ključnih elemenata za izvršenje zadataka. Prvi su predprocesorske naredbe. One počinju s ljestvicama (#) i pojavljuju se na vrhu C programa. Ove naredbe se daju procesoru prije stvarnog prevođenja programa. Primjeri predprocesorske naredbi uključuju uključivanje standardnih biblioteka poput <stdio.h> koja nudi funkcije za unos i ispis podataka, definiranje konstanti i proširivanje makro naredbi. Koriste se i druge biblioteke kao što su math.h, stdlib.h, string.h, time.h...

Svaki C program mora sadržavati funkciju main. To je glavna funkcija programa i predstavlja njegovu izvršnu točku. Unutar ove funkcije često se pozivaju druge funkcije za razbijanje složenih zadataka u manje, upravljivije dijelove. Tijelo funkcije main obično sadrži deklaracije varijabli, naredbe za unos podataka od korisnika, logiku za obradu tih podataka i naredbe za prikazivanje rezultata na ekran.

Varijable su osnova pohrane podataka u C programu. One se moraju deklarirati prije nego što se mogu koristiti. Deklaracija varijable uključuje navođenje njezinog tipa podatka (kao što su cijeli brojevi - int, znakovi - char ili realni brojevi - float) i davanje joj imena. Različiti tipovi podataka omogućuju varijablama da pohranjuju različite vrste informacija.

Funkcije su temeljni građevni blokovi modularnog programiranja u C-u. One omogućavaju razbijanje koda u manje, ponovljive dijelove koji obavljaju specificiran zadatak. Funkcije mogu primati opcionalne ulazne vrijednosti poznate kao argumenti i mogu vraćati vrijednosti programu nakon izvršenja.

Naredbe u C programu završavaju s točka-zarezom (;). Komentari se koriste za pojašnjenje koda i nisu dio izvršnog programa. Mogu se dodati bilo gdje unutar programa uz pomoć znakova // za jednolinijski komentar ili /\*\*…\*/ za višelinijski komentar.

Uobičajene funkcije za unos i ispis podataka u C-u su scanf i printf. Funkcija scanf čita podatke iz standardnog ulaza (najčešće tipkovnice) i pohranjuje ih u varijable, dok printf formatirano ispisuje podatke na standardni izlaz (najčešće ekran).

C sadrži i druge važne konceptualne elemente poput pokazivača, koji pohranjuju memorijske adrese drugih varijabli, i strukture, koje omogućavaju definiranje složenih tipova podataka za grupiranje različitih tipova varijabli zajedno. Također, C ima 32 ključne riječi koje imaju posebno značenje u jeziku i ne mogu se koristiti kao imena varijabli ili funkcija. [2]

|  |  |  |  |
| --- | --- | --- | --- |
| auto | double | int | struct |
| break | else | Long | switch |
| case | enum | register | typedef |
| char | extern | Return | union |
| const | float | short | unsigned |
| continue | for | signed | Void |
| default | goto | sizeof | volatile |
| do | If | static | while |

Tablica 2‑1 Prikaz 32 ključne riječi C-a [2]

U jeziku C koriste se različiti operatori koji igraju važnu ulogu u funkcioniranju programa. Poznajemo operatore kao što su aritmetički (+, -, \*, /, %), relacijski (<, >, >=, <=, ==, !=) te logički (&&, !, ||). Isto tako postoje i uvjetni operatori koji sadrže tri elementa s ciljem ispitivanja vrijednosti aritmetičkog izraza, te operator pridijeljena koji pridjeljuje varijabli s lijeve strane vrijednost izraza s desne strane operatora. Nadalje, veliku važnost dajemo i operatorima inkrementa (++) i dekrementa (--) čiji je zadatak uzeti staru vrijednost varijable, izračunati koliko iznosi ta vrijednost uvećana (ili umanjena) za 1 i rezultat spremiti kao novu vrijednost varijable.

U jeziku postoje 3 osnovne algoritamske strukture pod nazivima linijska, razgranata i ciklička. Linijska nastaje tako da navodimo naredbu za naredbom, razgranata omogućuje uvjetno grananje algoritma u kojem grananje može biti jednostruko (if), dvostruko (if-else) te višestruko (if-else, switch-case). Na kraju, ciklička struktura ima mogućnost da blok naredbi ponovi više puta pomoću tri programske petlje kao što su for, while i do-while petlje.

## Prednosti i mane jezika C

### Prednosti jezika C

Prednosti jezika C su što je on osnovni jezik za druge programske jezike, a učenje ovog jezika omogućava lakše shvaćanje ostalih. Također, C je strukturni jezik, što znači da programi mogu biti podijeljeni po modulima koji se mogu pisati odvojeno, što olakšava održavanje i smanjuje mogućnost pogrešaka.

Jezik se široko koristi za razvoj različitih aplikacija i sustava, uključujući aplikacije za Internet of Things (IoT), operativne sustave poput Appleovog OS X, Microsoftovog Windowsa i Symbiana, te za razvoj stolnih računala, mobilnih telefona, ugrađenih sustava, preglednika i baza podataka. Primjer najpopularnijeg softvera za baze podataka MySQL je napravljenog u C-u.

C je prenosiv jezik, što znači da se programi napisani u njemu mogu izvoditi na različitim računalima. Također, C sadrži razne funkcije koje su dio standardne biblioteke, ali također omogućuje korisnicima da dodaju vlastite funkcije u biblioteku i pristupe im bilo kad. Smatra se relativno jednostavnim jezikom koji omogućuje brže izvršavanje.

### Nedostatci jezika C

Nedostaci programskog jezika C uključuju složeniju sintaksu u usporedbi s drugim jezicima te nedostatak automatskog upravljanja memorijom, što zahtijeva ručno upravljanje memorijom. To može dovesti do problema poput curenja memorije ili prekoračenja memorije. Ručno upravljanje memorijom također može dovesti do sigurnosnih propusta.

C ne podržava sučelja i manje je apstraktan u usporedbi s jezicima kao što su Java i Python. Problemi se također mogu pojaviti prilikom korištenja funkcija `malloc()` i `calloc()`, gdje nepravilno oslobađanje memorije može uzrokovati greške, te postoji mogućnost da više pokazivača sadrže istu adresu, što može smanjiti optimizaciju programa.

## Primjeri koda u jeziku C

Kod 2‑1 prikazije funkciju koja učitava tekst iz datoteke i broji slova u njemu. Prvo pročita sadržaj datoteke i pomoću funkcije isalnum pronalazi samo alfanumeričke znakove te zapisuje njihov broj ponavljanja. Nakon čitanja cijele datoteke, ona se zatvara

A computer screen shot of a program code

Description automatically generated

Kod 2‑1 Brojanje slova u datoteci (C)

Kod 2‑2 prikazuje funkciju koja množi dvije matrice čiji su elementi spremljeni u zasebnim csv datotekama. Obje datoteke se otvaraju i čita se sadržaj red po red. Za svaki red koristi se funkcija strtok za izdvajanje pojedinačnih elemenata matrice odvojenih zarezima (","). Ti se elementi pretvaraju u cijele brojeve pomoću funkcije atoi te se brojevi upisuju u dinamički alocirane matrice. Nakon što su obje matrice učitane, program pristupa množenju matrica. Konačno, funkcija zatvara datoteke i oslobađa zauzetu memoriju prije svog završetka.

A screenshot of a computer program

Description automatically generated

Kod 2‑2 Množenje matrica (C)

Kod 2‑3 prikazuje funkciju koja sortira brojeve iz .csv datoteke pomoću bubble sort algoritma. Bubble sort prolazi kroz listu i zamjenjuje susjedne elemente ako su u pogrešnom redoslijedu, sve dok cijeli popis ne bude sortiran.

A screen shot of a computer code

Description automatically generated

Kod 2‑3 Bubble sort (C)

# PROGRAMSKI JEZIK PYTHON

## Povijest i karakteristike jezika Python

### Povijest jezika Python

Projekat Python pokrenut je 1990. godine na nizozemskom nacionalnom institutu (CWI) za matematiku i informatiku. Pokretač je bio Guido Van Rossum. Osmišljen je kao alternativa za ABC programske jezike. U 1995. izdana je verzija Python 1.2, a nakon toga tvorac Van Rossum odlazi u Virginiju u istraživačku korporaciju CNRI i tamo se izdaju verzije Pythona do 1.6. Nakon toga Van Rossum 2000. godine odlazi u kompaniju BeOpen.com i tamo se izdaje verzija 2.0, a nakon toga se osniva institut za Python softver te se objavljuje Python otvorenog koda. Velika nadogradnja Python 3.0 je izdana 2008. godine sa brojnim problemima vezanim za kompatibilnost sa starijom verzijom. Zbog toga je donesena odluka da se ipak većina glavnih funkcionalnosti vrati na prethodne verzije 2.6 i 2.7. [11]

A diagram of a timeline

Description automatically generated

Slika 3‑1 - Povijest Pythona [9]

### Karakteristike Pythona

Python je jezik dizajniran za korištenje u svakodnevnom programiranju. Python je višerazinski, dinamički, objektno orijentirani opći programski jezik koji se može koristiti u raznim primjenama. Osmišljen je da bude jednostavan za razumijevanje i korištenje te se zbog svoje fleksibilnosti često smatra kao prigodan jezik za početnike koji se upoznavaju sa programiranjem. Python ima mogućnost da neke greške zanemari te se program uspješno prevodi i pokreće sve do problematičnog dijela koda. Podržava razne stilove programiranja poput strukturnog i objektno orijentiranog programiranja. [6]

A black and white logo

Description automatically generatedPython se može pohvaliti iznimnom brzinom kodiranja i praktičnošću. Standardna biblioteka sadržava mnoštvo osnovnih operacija, ali i mnoštvo naprednih funkcija. Osim standardne biblioteke postoji mnoštvo biblioteka treće strane koje omogućavaju dodatne funkcionalnosti u okviru jezika Python. Primjerice, moguće je stvoriti web poslužitelj sa samo 3 linije programskog koda

Slika 3‑2 - Logo Pythona [10]

## Struktura i sintaksa jezika Python

Python se ističe svojom jednostavnom i čitljivom sintaksom pa je zato čest odabir programera: kako početnika, tako i onih iskusnih.

Python podržava tipove podataka kao što su: cjelobrojni brojevi („int“), decimalni brojevi („float“), nizovi znakova („str“), liste („list“), tuple-e („tuple“), rječnike („dict“), skupove („set“) i ostale. Tipovi varijabli se dodjeljuju dinamički i ne zahtijevaju eksplicitno deklariranje tipa. Varijable mogu i mijenjati tip ovisno o potrebi i samoj informaciji koju spremamo u njih.

Za kontrolu toka Python koristi ključne riječi poput: „if“, „elif“, „else“, a petlje sa ključnim riječima „for“ i „while“ za iteriranje kroz nizove ili izvršavanje blokova koda određeni broj puta.

Funkcije se u Pythonu definiraju pomoću ključne riječi „def“, kao i metode koje su funkcije definirane unutar klasa.

Python omogućuje organizaciju koda u module i pakete radi lakše upravljivosti i ponovne upotrebe. Moduli se uvoze korištenjem ključne riječi „import“.

Python koristi indentaciju (razmake ili tabulatore) za označavanje blokova koda što kod čini čitljivijim, ali zahtijeva i razinu dosljednosti pri pisanju koda. Naredbe uvučene razmacima ili tabulatorima se smatraju blokom koda podređenom nekoj funkciji, kontroli toka ili petljama.

Python podržava jednolinijske komentare koji se označavaju simbolom „#“, kao i višelinijske komentare koji se stavljaju između parova trostrukih navodnika „ ' ' ' “ ili „ """ “.

Naredbe u Pythonu ne moraju završavati sa točkom i zarezom „ ; “ kao u nekim drugim jezicima, a moguće je i prelamanje naredbe na više linija sa ili bez korištenja kose crte „ / “.

Funkcije print i input se koriste za ispis sadržaja u konzolu, odnosno za unos sadržaja sa konzole u neku varijablu.

A screenshot of a computer

Description automatically generatedPython kao i drugi jezici ima ključne riječi koje se ne mogu koristiti kao identifikatori za varijable ili funkcije. U trenutnoj verziji je 35 takvih riječi.

Slika 3‑3 - Prikaz 32 ključne riječi Pythona [11]

## Prednosti i mane jezika Python

### Prednosti jezika Python

Python je jezik poznat po jednostavnosti učenja i korištenja za početnike. Njegova jednostavna i logična sintaksa ga čini prigodnim za učenje i korištenje, pri čemu nije nužno da programer detaljno razumije princip funkcioniranja memorije i procesora kao što je slučaj kod nižih programskih jezika.

Osim jednostavnosti Python se može pohvaliti i velikom zajednicom programera koji dijele svoja rješenja i implementacije na raznim web servisima poput foruma. Osim zajednice i sveprisutnosti Python ima razne mogućnosti za suradnju i korištenje sa drugim tehnologijama često korištenim u IT industriji. Sposobnost manipulacijom velikim količinama podataka i paralelnom procesiranju ga čini korisnim i za naprednije grane poput podatkovne znanosti i strojnog učenja. Python kao jezik ima široku primjenu u raznim granama te je po TIOBE Indeksu najrasprostranjeniji programski jezik.[link: https://www.tiobe.com/tiobe-index/] Navedene činjenice su razlog zbog kojeg je Python dobar odabir za sve programere, a posebno početnike.

### Nedostatci jezika Python

Python je interpretirani jezik, što znači da se svaka linija koda izvršava jedna po jedna, za razliku od kompajliranih jezika koji se prvo prevode u strojni jezik. Zbog te činjenice Python ima duže vrijeme izvršavanja od kompajliranih jezika. Kod manipulacije manjom količinom podataka Python je dobra opcija, unatoč sporijem izvršavanju koje će na modernim računalima biti zanemarivo, ali kod kompleksnijih kalkulacija i manipulacija velikim količinama podataka sporije vrijeme izvršavanja može doći do izražaja.  
Python se radi dinamičnosti tipova podataka smatra manje sigurnim nego jezici u kojima programer pri pisanju koda sam određuje tipove podataka. Zbog automatske deklaracije vrsta podataka može doći do prekoračenja međuspremnika ili malicioznih napada ubacivanjem u međuspremnik. Činjenica da se Python često koristi čini ga metom proučavanja zlonamjernih zlikovaca engl. *hackers* koji traže sigurnosne propuste. Python je višerazinski programski jezik koji ne pruža korisnicima mogućnosti manipulacije memorijom i drugim osnovnim komponentama računala, što ga čini nepovoljnim za projekte koji zahtijevaju pristup istima.  
Python nudi mnoge apstrakcije i automatizacije pri korištenju, a to dovodi do mogućeg korištenja velikih količina memorije čak i za jednostavnije zadatke. Navedena problematika se očituje kod kompleksnijih manipulacija većom količinom podataka. Osim manje efikasnog korištenja memorije automatizacija otvara mogućnosti i za neočekivano ponašanje pri automatskom dodjeljivanju tipova podataka, kao i otežavanje praćenja i razumijevanja programa zbog mogućih čestih mijenjanja tipova podataka.  
Iako Python ima modul za rad s nitima (engl. *threading* *module*), nije potpuno višenitan (engl. *multithreaded*) zbog Global Interpreter Lock-a (GIL). GIL osigurava da samo jedna nit može izvršavati Python bajtkod u isto vrijeme, čak i na višejezgernim sustavima. To znači da iako se u Pythonu mogu stvoriti višestruke niti, ne mogu se izvršavati paralelno, što ograničava potencijalne dobitke u performansama korištenjem višenitnog programiranja. [7]

## Primjeri koda u jeziku Python

Kod 3‑1 prikazuje funkciju koja učitava tekst iz datoteke i broji slova u njemu, vodeći računa o velikom i malom slovu. Prvo pročita sadržaj datoteke i definira liste znakova koje ne želi brojati (kao što su razmaci i interpunkcijski znakovi). Zatim prolazi kroz svako slovo u tekstu i broji pojavljivanja svakog slova.

A computer screen with text

Description automatically generated

Kod 3‑1 Brojanje slova u datoteci (Pyhton)

Kod 3‑2 prikazuje funkciju koja učitava dvije matrice iz csv datoteka i množi ih. Nužno je uvesti biblioteke "numpy" i "pandas". “Pandas” omogučuje čitanje vrijednosti iz csv datoteka u varijable matrix1 i matrix2. Kasije s koristi .dot funkcija iz biblioteke NumPy za množenje matrica.

A screen shot of a computer program

Description automatically generated

Kod 3‑2 Množenje matrica (Python)

Kod 3‑3 prikazuje funkciju koja sortira brojeve iz .csv datoteke pomoću bubble sort algoritma. Bubble sort prolazi kroz listu i zamjenjuje susjedne elemente ako su u pogrešnom redoslijedu, sve dok cijeli popis ne bude sortiran.

A computer screen with many colorful text

Description automatically generated with medium confidence

Kod 3‑3 Bubble sort (Python)

# USPOREDBA PERFORMANSI

## Metodologija testiranja performansi

Python i C su dva temeljno različita programska jezika koja se razlikuju u gotovo svakom pogledu pa je testiranje performansi i usporedba takva dva jezika poprilično zahtjevna. Iz tog razloga ideja je bila koristiti metodologiju testiranja koja će izolirati što više parametara koji su nevažni za ciljeve ovog istraživanja, a koji bi mogli utjecati na konačne rezultate.

U svrhu usporedbe performansi programskih jezika C i Python, primijenjena je metodologija koja se temelji na kvantitativnim mjerenjima vremena izvršavanja i potrošnje memorije. Odabran je ovakav pristup kako bi se omogućila precizna usporedba između jezika, uzimajući u obzir samo ključne funkcije implementirane u svakom jeziku, a ignorirajući ostale dijelove programa kako bi se eliminirali faktori koji bi mogli utjecati na rezultate.

### Testne funkcije

1. **count\_letters**: Prva testna funkcija, nazvana "count\_letters", dizajnirana je za učitavanje opsežnog teksta iz .txt datoteke te provodi brojanje alfanumeričkih znakova.
2. **matrix\_multiplication**: Druga testna funkcija, nazvana "matrix\_multiplication", služi za učitavanje dvije matrice, pri čemu svaka matrica dolazi iz zasebne .csv datoteke, te izvodi množenje matrica.
3. **bubble\_sort**: Treća testna funkcija, nazvana "bubble\_sort", implementirana je za sortiranje niza od 30000 brojeva koristeći bubble sort algoritam.

### Proces testiranja

Za svaku od testnih funkcija, izvršeno je deset iteracija u svakom jeziku kako bi se osigurala pouzdanost rezultata. Tijekom svake iteracije, zabilježena su vremena izvršavanja funkcija i potrošena memorija. Konačni rezultati dobiveni su izračunavanjem prosječnih i medijalnih vrijednosti vremena izvršavanja te potrošene memorije za svaki jezik.

## Proces mjerenja vremena izvođenja i potrošnje memorije

Kako bi standardizirali metodologiju testiranja i olakšali samo izvođenje, prethodno samom testiranju napravljene su funkcije za oba jezika koje će se pozivati prilikom izvođenja testnih funkcija te mjeriti same performanse koda.

### Mjerenje vremena izvođenja i potrošnje memorije u C-u

A screen shot of a computer program

Description automatically generatedProces mjerenja potrošnje memorije funkcije u C-u je lak za implementirati jer C kao jezik omogućuje baratanje memorijom na niskoj razini pa je njena potrošnja u svakom trenutku jednostavna za dohvatiti. Mjerenje vremena izvođenja funkcije je trivijalno te je samo delta između vremena na kraju izvođenja funkcije i onoga kada je funkcija pozvana.

Kod 4‑1 Mjerenje vremena izvođenja i potrošnje memorije (C)

### Mjerenje vremena izvođenja i potrošnje memorije u Python-u

A screenshot of a computer program

Description automatically generatedProces mjerenja vremena izvođenja funkcije u Pythonu je praktički jednak onome u C-u te je lako izvediv. S druge strane, Python je jezik koji od programera apstraktira rad s memorijom te njeno alociranje u toku izvršavanja programa. Iz tog razloga, mjerenje memorije koju testna funkcija potroši je znatno kompliciranije te ne nudi ni približno precizne rezultate.

Kod 4‑2 Mjerenje vremena izvođenja i potrošnje memorije (Python)

# REZULTATI TESTIRANJA

Slika na kojoj se prikazuje tekst, snimka zaslona, Font, broj

Opis je automatski generiranNakon odrađenih mjerenja i testiranja performansi dvaju jezika služeći se metodologijom opisanom kroz cjelinu 4., kao rezultat smo dobili prosječnu potrošenu memoriju te prosječno vrijeme izvršavanja programa za oba jezika. Podaci su dobiveni tako da je uzet prosjek medijana i aritmetičke sredine 10 mjerenja za svake od tri testne funkcije.

Slika na kojoj se prikazuje tekst, snimka zaslona, Font, broj

Opis je automatski generiranSlika na kojoj se prikazuje tekst, snimka zaslona, Font, broj

Opis je automatski generiranNa prvi pogled, vidljiva je blaga razlika u potrošnji memorije između dvaju programskih jezika, dok se kod vremena izvršavanja funkcije vidi ogromna razlika. Ovi ukupni rezultati, odnosno prosjeci svih rezultata, nam vrlo brzo sumiraju cijelu priču, ali analizirat ćemo je detaljnije da otkrijemo gdje se stvara takva razlika.

## Vizualni prikaz rezultata

Naše se istraživanje svodilo na mjerenje performansi 3 podosta različitih funkcija napisanih u oba programska jezika. Zbog toga ukupni rezultat, prikazan poviše, nije stopostotno primjenjiv na sve oblike funkcija i programa. Iz tog razloga ćemo analizirati i vizualizirati podatke dobivene za svaki od tri testna slučaja.

Slika na kojoj se prikazuje tekst, snimka zaslona, Font, broj

Opis je automatski generiranKod prve testne funkcije, „counting\_letters“ , imamo jasnu sliku razlike vremena izvođenja funkcije koje je oko 6-7 puta duže u Pythonu nego u C-u. Što se tiče potrošnje memorije, u ovom slučaju je ponešto veća u C-u, ali je, kao što je slučaj u cijelom istraživanju, konstantnija kroz sve testne slučajeve.

Slika na kojoj se prikazuje tekst, snimka zaslona, crta, paralelno

Opis je automatski generiran

Slika na kojoj se prikazuje tekst, snimka zaslona, crta, Font

Opis je automatski generiranDruga testna funkcija je jako zahtjevna za realizaciju u programskom jeziku C, a još teže je ostvariti to na optimalan način. Zato u ovom primjeru vidimo bolje performanse vremena izvođenja u programskom jeziku Python koji nam omogućuje rješenje u par linija te se sam bavi optimizacijom bolje od nas. Potrošnja memorije, s druge strane, je ponešto veća i manje konstantna u Python-u.

Slika na kojoj se prikazuje tekst, snimka zaslona, Font, dijagram

Opis je automatski generiranTreća testna funkcija je klasični algoritam za sortiranje – „Bubble sort“. Jednostavan za implementirati u oba jezika, ali performanse izvođenja se drastično razlikuju, pogotovo kada je riječ o vremenu izvođenja. Razlika na testnom nizu od 30000 brojeva je stotinu puta dulje izvođenje u programskom jeziku Python. Također, potrošnja memorije je u prosjeku par puta veća i manje konstantna.

Slika na kojoj se prikazuje tekst, snimka zaslona, Font, dijagram

Opis je automatski generiranSlika na kojoj se prikazuje tekst, snimka zaslona, Font, broj

Opis je automatski generiran

Vizualizirali smo i analizirali sve tri testne funkcije kako bi se pobliže vidjela razlika u performansama između dva programska jezika u različitim zadaćama. Sada treba dobivene rezultate dublje proanalizirati te iz njih izvući ono što je bitno kako bi se došlo do zaključka. Upravo to slijedi u sljedećem podnaslovu.

## Tumačenje dobivenih rezultata

### Objašnjenje ukupnih rezultata

Kroz ovu cjelinu smo brojčano i vizualno prikazali sve relevantne rezultate našeg testiranja, a sada je vrijeme da ih pojasnimo. U tablici #1? vidimo da je ukupno, gledajući prosjeke svih provedenih testova, programski jezik C efikasniji, odnosno ima bolje performanse gledajući potrošnju memorije te vrijeme izvođenja. Glavna, i nama najvažnija, razlika je u vremenu izvođenja programa, odnosno funkcije između dvaju jezika. Prosječno vrijeme izvođenja programa se razlikuje čak preko 30 puta između dva programska jezika tokom našeg testiranja. Za razliku od potrošnje memorije, koja je u tako malim razlikama praktički neprimjetna kod modernih sistema, ovakav rezultat u razlici vremena izvođenja je ipak vrlo značajan. Ipak, ovi rezultati su samo prosjek svega testiranog, dok je situacija drastično drugačija između različitih funkcija i programa. Zbog toga, kako bi našli „bolji“ jezik za našu primjenu, treba znati za što ćemo ga koristiti.

### Rezultati po testnim funkcijama

Prva testna funkcija, vremenske kompleksnosti O(N), učitava velik tekst u obliku .txt datoteke te broji određene karaktere u njemu. Takve funkcije s puno brojanja, sortiranja i slično su uvijek pogodne za C programski jezik, što je i vidljivo u rezultatima, gdje je C za 6-7 puta brži po vremenu izvođenja. Potrošnja memorije je ne zamjetno veća u C-u, ali za to je moguće razlog krivi pristup programera i učitavanje tekstualnih datoteka umjesto binarnih.

Druga testna funkcija, vremenske kompleksnosti O(N^3), je najzahtjevnija testna funkcija koja učitava 1000x1000 matrice iz .csv fileova te vrši množenje matrica. Iako manje potrošnje memorije u odnosu na Python, funkcija se u C-u izvršavala oko 5 puta dulje. To bi upućivalo na lošije performanse kod takve vrste zadatka, ali ovdje je ključnu ulogu u razlici činio pristup programera. Najme, C omogućuje detaljno baratanje memorijom pa je program moguće napisati optimalno sa strane performansi, ali ukoliko programer nema znanja za to, može se desiti obratno. U ovom slučaju, glavni razlozi za lošije performanse u C verziji programa su način učitavanja matrica (iz .csv datoteka umjesto binarno) te neznanje programera da optimizira ovako kompleksnu funkciju.

Treća testna funkcija, vremenske kompleksnosti O(N^2), je zapravo klasični bubble sort algoritam koji sortira niz od 30000 brojeva. Ovakve jednostavni, a iznimno repetitivni zadaci su perfektni za izvođenje u C programskom jeziku, a to se vidi i na rezultatima testiranja. Dok je u potrošnji memorije Python bio samo ponešto lošiji, u vremenu izvođenja se stvorila najveća razlika unutar cijelog testiranja. Kod se u C programskom jeziku izvršavao čak 100 puta kraće u odnosu na onaj u Python-u. Iako 100 puta samo po sebi zvuči puno, bitno je napomenuti da kada bi umjesto niza od 30k brojeva koristili neki od 100k, testiranje u Pythonu bi postalo praktični nemoguće, dok bi u C-u to samo bilo pitanje par sekundi razlike.

### Mjerivi i nemjerivi utjecaji na rezultate

Bitno je napomenuti par stavki koje su na jedan ili drugi način utjecali na ovo testiranje te moguće i na same rezultate testiranja. Vrijeme izvođenja funkcije je podosta jednostavno izmjeriti precizno u oba programska jezika. S druge strane, izmjeriti memoriju koju mjerena funkcija iskoristi je malo kompliciranije, a rezultati koji se dobiju nisu uvijek točni. To je pogotovo slučaj u Python-u jer napravljen da je upravljanje memorijom odvojeno od samog programera pa koliko god to čini stvari jednostavnijima, također otežava ili čak i onemogućava određene pothvate poput preciznog mjerenja potrošene memorije određene funkcije. Iz tog razloga se na rezultatima mjerenja memorije vidi određena ne konstantnost kada je u pitanju Python.

Nadalje, postoji jedna važna stavka koju je teško izmjeriti i direktno usporediti, ali je vrlo važna i predstavlja barem istu važnost kod odabira programskog jezika, kao i testirane performanse. To je lakoća pisanja koda i postavljanja radne okoline. Iako je prva stavka podosta subjektivna, druga nije te je vrlo očita razlika između postavljanja radne okoline za oba testirana jezika. Naime, C je jezik za kojeg je potrebno imati kompajler i linker kako bi se napisani kod mogao naknadno prevesti u strojni te zatim spojiti u jednu izvršnu datoteku. S druge strane, Python je interpreterski jezik pa se taj proces dešava „on the go“. Poanta je da je Python puno jednostavnije postaviti i odmah krenuti raditi, a ta razlika je drastično povećana kod početnika. Što se tiće lakoće pisanja koda, naše je mišljenje da tu također Python odnosi veliku pobjedu, koja će također, za početnika biti još važnija.

# ZAKLJUČAK

Ovaj rad imao je za cilj pružiti detaljnu usporedbu dvaju popularnih programskih jezika, C i Python, s posebnim naglaskom na njihove performanse. Analizirali smo povijest, karakteristike, strukturu i sintaksu oba jezika te smo razmotrili njihove prednosti i mane. Na temelju tih analiza, dobili smo jasnu sliku o specifičnostima i praktičnoj primjeni oba jezika.

Programski jezik C se pokazao izvanrednim u pogledu brzine izvršavanja i efikasnosti memorije. Njegova blizina hardveru omogućuje programerima izravnu kontrolu nad resursima sustava, što rezultira bržim i efikasnijim kodom. Osim toga, C je temelj za mnoge druge programske jezike, što ga čini nezaobilaznim u obrazovanju programera i u razumijevanju temeljnih koncepata računalnih znanosti.

S druge strane, Python je interpretirani jezik koji se ističe jednostavnošću sintakse i čitljivošću koda. Python omogućuje brži razvoj aplikacija zbog svoje dinamičke tipizacije i visoke razine apstrakcije. Iako Python troši više memorije i ima niže performanse u smislu brzine izvršavanja u usporedbi s C-om, njegova fleksibilnost i bogat ekosustav biblioteka često nadmašuju ove nedostatke. Python omogućuje programerima da brzo razvijaju i iteriraju svoje projekte, što je ključno u dinamičnim okruženjima gdje je vrijeme razvoja kritično.

Testiranje performansi provedeno u ovom radu pokazalo je da C nadmašuje Python u pogledu brzine izvršavanja. To je očekivano s obzirom na to da je C kompilirani jezik koji generira efikasan strojni kod, dok je Python interpretirani jezik koji se izvršava na višoj razini apstrakcije. Međutim, Python se pokazao vrlo učinkovitim u brzim iteracijama i razvoju, što je njegova velika prednost. Dok C zahtijeva ručno upravljanje memorijom i detaljno razumijevanje hardverskih aspekata, Python pruža automatizirano upravljanje memorijom i jednostavan pristup složenim strukturama podataka.

Zaključak svega rečenog je da oba jezika imaju svoje primjene te je nemoguće direktno reći da je jedan bolji od drugog. Ipak, kroz ovo istraživanje i testiranje smo vidjeli razlike u njihovim performansama, lakoći pisanja koda i jednostavnosti postavljanja radne okoline. Sve što smo radili, od postavljanja radnog okruženja pa do testiranja, lakše je i jednostavnije bilo napraviti u Pythonu. Zato zaključujemo, da ukoliko nam performanse i baratanje memorijom nisu na vrhu liste prioriteta te ukoliko nam radno okruženje to omogućuje, uporaba Python-a se čini boljim izborom za bilo kojeg programera, od početnika to seniora, kojemu je najvažnija stavka lakoća i brzina pisanja koda te jednostavnost njegovog održavanja.

LITERATURA

1. Barbara Thompson – What is C Programming Langauge?, s Interneta: <https://www.guru99.com/c-programming-language.html>, zadnji pristup: 02.04.2024.
2. Dumančić, F. (2016). Logička zagonetka "Podmornice" u programskom jeziku C, Osijek, s Interneta: <https://repozitorij.etfos.hr/islandora/object/etfos:955/datastream/PDF/view>, zadnji pristup 02.04.2024.
3. Jurak – Programski jezik C, s Interneta: <https://web.math.pmf.unizg.hr/~jurak/C.pdf>, zadnji pristup 02.04.2024.
4. Bilić – Razvoj programskog jezika C, s Interneta: <https://www.scribd.com/document/710510429/Razvoj-Programskog-Jezika-C>, zadnji pristup 02.04.2024.
5. Nosrati, M. (2011). Python: An appropriate language for real world programming. *World Applied Programming*, 1(2), 110-117.
6. Srinath, K. R. (2017). Python–the fastest growing programming language. *International Research Journal of Engineering and Technology*, *4*(12), 354-357.
7. Gavrilova – Pros and Cons of Python Programming Language, s Interneta: <https://serokell.io/blog/python-pros-and-cons>, zadnji pristup 02.04.2024.
8. C, s Interneta:<https://commons.wikimedia.org/wiki/File:C_Programming_Language.svg>, zadnji pristup 18.04.2024.
9. Goyal – What is Python?, s Interneta: <https://unstop.com/blog/what-is-python>, zadnji pristup 18.04.2024.
10. Python, s Interneta: <https://www.python.org/>, zadnji pristup 18.04.2024.
11. PEP 3000, s Interneta: <https://peps.python.org/pep-3000/>, zadnji pristup 18.04.2024.
12. Hansem – Pyhton Keywords: An Introduction, s Internets: <https://realpython.com/python-keywords/>, zadnji pristup 18.04.2024.

PRILOZI

Kazalo slika, tablica i kodova

Kazalo slika

[Slika 2‑1 - Logo C-a [8] 2](#_Toc164353747)

[Slika 2‑2 – Povijest C-a [1] 3](#_Toc164353748)

[Slika 3‑1 - Povijest Pythona [9] 9](#_Toc164353749)

[Slika 3‑2 - Logo Pythona [10] 10](#_Toc164353750)

[Slika 3‑3 - Prikaz 32 ključne riječi Pythona [11] 11](#_Toc164353751)

Kazalo tablica

[Tablica 1‑1 Doprinosi koautora rada 1](#_Toc164353752)

[Tablica 2‑1 Prikaz 32 ključne riječi C-a [2] 4](#_Toc164353753)

Kazalo kodova

[Kod 2‑1 Brojanje slova u datoteci (C) 6](#_Toc166065620)

[Kod 2‑2 Množenje matrica (C) 7](#_Toc166065621)

[Kod 2‑3 Bubble sort (C) 8](#_Toc166065622)

[Kod 3‑1 Brojanje slova u datoteci (Pyhton) 13](#_Toc166065623)

[Kod 3‑2 Množenje matrica (Python) 14](#_Toc166065624)

[Kod 3‑3 Bubble sort (Python) 14](#_Toc166065625)

[Kod 4‑1 Mjerenje vremena izvođenja i potrošnje memorije (C) 16](#_Toc166065626)

[Kod 4‑2 Mjerenje vremena izvođenja i potrošnje memorije (Python) 17](#_Toc166065627)

Popis oznaka i kratica

ALGOL algorithmic language

BCPL basic combined programming language

CNRI Corporation for National Research Initiatives

SAŽETAK I KLJUČNE RIJEČI

**Sažetak[[1]](#footnote-1)**

*U cca 100-150 riječi sažeti cijelu radnju. Piše se na kraju ALI GA PROČITA NAJVEĆI BROJ LJUDI (85%) KOJI OSIM NASLOVA UOPĆE NEŠTO OD RADNJE PROČITAJU. ZATO U NJEMU TREBA SAŽETI ONO ŠTO U RADNJI DONOSIMO NADUGO I NAŠIROKO! ☺☺☺ Piše se u trećem licu jednine (bezlično), uglavnom u pasivu i ne u futuru i sadašnjosti – nego u perfektu (tj. napravljeno je, analizirano, korišteni su, prikazane su, itd). Nema više paragrafa – sve je jedan paragraf.*

**Ključne riječi**

*5-6 ključnih riječi (mogu biti i složeni pojmovi) koji najbolje odražavaju sadržaj radnje. Nema paragrafa – sve je jedan paragraf. Pojmovi se ne razdvajaju TOČKOM I ZAREZOM (;) nego zarezima).*

1. [↑](#footnote-ref-1)