

Naslov seminarskog rada

Seminarski rad u okviru kursa
Metodologija stručnog i naučnog rada
Matematički fakultet

Prvi autor, drugi autor, treći autor, četvrti autor
kontakt email prvog, drugog, trećeg, četvrtog autora

2. april 2019

Sažetak

U ovom tekstu je ukratko prikazana osnovna forma seminarskog rada. Obratite pažnju da je pored ove .pdf datoteke, u prilogu i odgovarajuća .tex datoteka, kao i .bib datoteka korišćena za generisanje literature. Na prvoj strani seminarskog rada su naslov, apstrakt i sadržaj, i to sve mora da stane na prvu stranu! Kako bi Vaš seminarski zadovoljio standarde i očekivanja, koristite uputstva i materijale sa predavanja na temu pisanja seminarskih radova. Ovo je samo šablon koji se odnosi na fizički izgled seminarskog rada (šablon koji *morate* da koristite!) kao i par tehničkih pomoćnih uputstava. Pročitajte tekst pažljivo jer on sadrži i važne informacije vezane za zahteve obima i karakteristika seminarskog rada.

Sadržaj

1	Uvod	3
2	Istorijat	3
3	Osnovna uputstva	4
4	Engleski termini i citiranje	4
5	Slike i tabele	5
6	Kôd i paket listings	6
7	Prvi naslov	6
7.1	Prvi podnaslov	6
7.2	Drugi podnaslov	6
7.3	... podnaslov	6
8	n-ti naslov	6
8.1	... podnaslov	6
8.2	... podnaslov	6
9	Zaključak	7
	Literatura	7

1 Uvod

Elixir(Elik sir) je programski jezik koji se prvi put pojavljuje u javnosti 2012. godine kao projekat kompanije Plataformatec. Ovaj funkcionalni, dinamičan programski jezik*(footnote: Diskusija o specifičnoj paradigmi u daljem tekstu) se pokreće na Erlang vituelnoj mašini pa samim tim i deli pogodna svojstva kao što su konkurentnost i tolerisanje grešaka, koje dolaze sa ovim okruženjem. Njegov tvorac, Jose Valim, navodi da je motivacija za pravljenje ovog jezika upravo bila ljubav prema ovoj virtualnoj mašini i ekosistemu, ali i njeni nedostaci. Iz tačke gledišta olakšavanja svakodnevnog razvoja sofvera, koncepti poput metaprogramiranje - tehnika kojom programi imaju mogućnost da druge programe posmatraju kao svoje podatke i na taj način čitaju pa čak i modifikuju njihov, a samim tim i svoj kod u vreme izvršavanja, zatim polimorfizam (pretp. da čitaoc zna, pitati prof.) i makroi kao i podrška za alate, bili su neke od nepostojećih karakteristika ovog sistema koje bi Elixir trebalo da nadomesti. Cilj ovog rada je da čitaoca bliže upozna sa osobinama, funkcionalnostima i specifičnostima ovog jezika kao i da kroz uporedni prikaz sa jezicima koji dele slične kocepte ili imaju istu upotrebu u određenim domenima, prikaže make i prednosti Elixira.

2 Istorijat

Tokom 1980ih, telekomunikaciona kompanija Ericsson ispitivala je problem konkurentnosti, i nakon izvršenih testiranja odlučila je da razvije svoj jezik koji bi bio zasnovan na ovoj funkcionalnosti: Erlang. U to vreme brojnost jezika nije bila velika, kao danas. Programski jezik C je bio najzastupljeniji, ali uprkos njegovoj prilagodljivosti, tri programera Joe Armstrong, Mike Williams i Robert Virding, koji su bili zaposleni u Ericsson-u, su shvatili da mogu biti produktivniji ukoliko koriste programski jezik višeg nivoa. Nakon eksperimenata sa postojećim jezicima, odlučili su se da ni jedan ne zadovoljava njihove potrebe u dovoljnoj meri, pa su tokom 1987. započeli pisanje Erlanga i predstavili ga tri godine kasnije.

Jezik je najviše korišćen od strane telekomunikacionih kompanija, a glavni fokus je bila distribucija, tolerantnost na otkaze u sistemu, i hot-swappinga koda (mogućnost izmene koda programa bez potrebe prekida izvršavanja). U 1998. Ericsson se odlučuje da kod programskoj jezika Erlang učini dostupan svima i proglašava ga otvorenim (open source). Od tada je jezik korišćen u veoma specifičnim projektima, kao što je dobro poznati broker za poruke RabbitMQ, XMPP server “ejabberd”, i Apache CouchDB, koji je jedna od najpoznatijih distribuiranih baza podataka. Neke kompanije koriste Erlang za telekomunikacione proizvode, a primer su WhatsApp i Riot Games - kompanija koja je napravila igru League of Legends.

Tokom 2010., Jose Valim, u to vreme zaposlen na poziciji programera u kompaniji Plataformatec, radio je na poboljšanju performansi Ruby on Rails framework-a na višejezgarnim sistemima i bio je sve više frustriran ovim poslom. Shvatio je da Ruby nije bio dovoljno dobro dizajniran da reši problem konkurentnosti, pa je započeo istraživanje drugih tehnologija koje bi bile prihvatljivije. Tako je otkrio je Erlang, i upravo ga je interesovanje prema Erlangovoj virtuelnoj mašini podstaklo da započne pisanje Elixira. Uticaj projekta na kome je do tada radio odrazio se na to da Elixir ima sintaksu koja je nalik na Ruby-jevu. Ovaj jezik se pokazao veoma

dobro pri upravljanju milionima simultanih konekcija: u 2015. Phoenix - zabeleženo upravljanje 2 miliona WebSocket konekcija, dok je u 2017. za skalirani Elixir zabeležena obrada 5 miliona istovremenih korisnika. (Q: Možda bismo mogli neki grafik ili sliku o ovome) Elixir se danas koristi u velikim kompanijama, kao što su Pinterest, Moz, a upotrebu nalazi čak i u bankarskim sistemima. (q: pitati jel medium relevantan)(ref: <https://medium.com/margobank/why-elixir-546427542c>)

3 Osnovna uputstva

Vaš seminarski rad mora da sadrži najmanje jednu **sliku**, najmanje jednu **tabelu** i najmanje **sedam referenci** u spisku literature. Najmanje jedna slika treba da bude originalna i da predstavlja neke podatke koje ste Vi osmislili da treba da prezentujete u svom radu. Isto važi i za najmanje jednu tabelu. Od referenci, neophodno je imati bar jednu **knjigu**, bar jedan **naučni članak** iz odgovarajućeg časopisa i bar jednu adekvatnu **veb adresu**.

Dužina seminarskog rada treba da bude od 10 do 12 strana. Svako prekoračenje ili potkoračenje biće kažnjeno sa odgovarajućim brojem poena. Eventualno, nakon strane 12, može se javiti samo tekst poglavlja **Dodatak** koji sadrži nekakav dodatni kôd, ali je svakako potrebno da rad može da se pročita i razume i bez čitanja tog dodatka.

Ко жели, може да пише рад ћирилицом. У том случају, неопходно је да су инсталирани одговарајући пакети: texlive-fonts-extra, texlive-latex-extra, texlive-lang-cyrillic, texlive-lang-other.

Nemojte koristiti stari način pisanja slova, tj ovo:

```
\v{s} i \v{c} i \'c ...
```

Koristite direktno naša slova:

```
š i č i ć ...
```

4 Engleski termini i citiranje

Na svakom mestu u tekstu naglasiti odakle tačno potiču informacije. Uz sve novouvedene termine u zagradi naglasiti od koje engleske reči termin potiče.

Naredni primeri ilustruju način uvođenja engleskih termina kao i citiranje.

Primer 4.1 *Problem zaustavljanja (eng. halting problem) je neodlučiv [3].*

Primer 4.2 *Za prevođenje programa napisanih u programskom jeziku C može se koristiti GCC kompajler [1].*

Primer 4.3 *Da bi se ispitivala ispravnost softvera, najpre je potrebno precizno definisati njegovo ponašanje [2].*

Reference koje se koriste u ovom tekstu zadate su u datoteci *seminarski.bib*. Prevođenje u pdf format u Linux okruženju može se uraditi na sledeći način:

```
pdflatex TemaImePrezime.tex
bibtex TemaImePrezime.aux
pdflatex TemaImePrezime.tex
pdflatex TemaImePrezime.tex
```

Prvo latexovanje je neophodno da bi se generisao `.aux` fajl. `bibtex` proizvodi odgovarajući `.bbl` fajl koji se koristi za generisanje literature. Potrebna su dva prolaza (dva puta `pdflatex`) da bi se reference ubacile u tekst (tj da ne bi ostali znakovi pitanja umesto referenci). Dodavanjem novih referenci potrebno je ponoviti ceo postupak.

Broj naslova i podnaslova je proizvoljan. Neophodni su samo Uvod i Zaključak. Na poglavlja unutar teksta referisati se po potrebi.

Primer 4.4 *U odeljku 7 precizirani su osnovni pojmovi, dok su zaključci dati u odeljku 9.*

Još jednom da napomenem da nema razloga da pišete:

`\v{s}` i `\v{c}` i `\'c` ...

Možete koristiti srpska slova

š i č i ć ...

5 Slike i tabele

Slike i tabele treba da budu u svom okruženju, sa odgovarajućim naslovima, obeležene labelom da koje omogućava referenciranje.

Primer 5.1 *Ovako se ubacuje slika. Obratiti pažnju da je dodato i `\usepackage{graphicx}`*



Slika 1: Pande

Na svaku sliku neophodno je referisati se negde u tekstu. Na primer, na slici 1 prikazane su pande.

Primer 5.2 *I tabele treba da budu u svom okruženju, i na njih je neophodno referisati se u tekstu. Na primer, u tabeli 1 su prikazana različita poravnanja u tabelama.*

Tabela 1: Različita poravnanja u okviru iste tabele ne treba koristiti jer su nepregledna.

centralno poravnanje	levo poravnanje	desno poravnanje
a	b	c
d	e	f

6 Kôd i paket listings

Za ubacivanje koda koristite paket **listings**: https://en.wikibooks.org/wiki/LaTeX/Source_Code_Listings

Primer 6.1 *Primer ubacivanja koda za programski jezik Python dat je kroz listing 1. Za neki drugi programski jezik, treba podesiti odgovarajući programski jezik u okviru defnisanja stila.*

```
1000 # This program adds up integers in the command line
1001 import sys
1002 try:
1003     total = sum(int(arg) for arg in sys.argv[1:])
1004     print 'sum =', total
1005 except ValueError:
1006     print 'Please supply integer arguments'
```

Listing 1: Primer ubacivanja koda u tekst

7 Prvi naslov

Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst.

7.1 Prvi podnaslov

Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst.

7.2 Drugi podnaslov

Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst.

7.3 ... podnaslov

Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst.

8 n-ti naslov

Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst.

8.1 ... podnaslov

Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst.

8.2 ... podnaslov

Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst. Ovde pišem tekst.

9 Zaključak

Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak.
Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak.
Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak.
Ovde pišem zaključak. Ovde pišem zaključak. Ovde pišem zaključak.

Literatura

- [1] Free Software Foundation. GNU gcc, 2013. on-line at: <http://gcc.gnu.org/>.
- [2] J. Laski and W. Stanley. *Software Verification and Analysis*. Springer-Verlag, London, 2009.
- [3] A. M. Turing. On Computable Numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.

A Dodatak

Ovde pišem dodatne stvari, ukoliko za time ima potrebe. HOvde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe. Ovde pišem dodatne stvari, ukoliko za time ima potrebe.