

# Programski jezik Objective-C

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

Dalma Beara, Denis Aličić, Mateja Marjanović, Anja Miletić  
kontakt email prvog, drugog, trećeg, četvrtog autora

23. mart 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 ispoštujete!) kao i par tehničkih pomoćnih uputstava. Pročitajte tekst pažljivo jer on sadrži i važne informacije vezane za zahteve obima i karakteristika seminarskog rada.

## Sadržaj

<b>1</b>	<b>Uvod</b>	<b>3</b>
<b>2</b>	<b>Osnovna uputstva</b>	<b>3</b>
<b>3</b>	<b>Engleski termini i citiranje</b>	<b>3</b>
<b>4</b>	<b>Slike i tabele</b>	<b>4</b>
<b>5</b>	<b>Kôd i paket listings</b>	<b>4</b>
<b>6</b>	<b>Nastanak i istorijski razvoj, mesto u razvojnom stablu, uticaji drugih programskih jezika</b>	<b>5</b>
6.1	Nastanak i istorijski razvoj . . . . .	5
6.2	Mesto u razvojnom stablu . . . . .	5
6.3	Uticaj drugih programskih jezika . . . . .	5
<b>7</b>	<b>Osnovna namena programskog jezika, svrha i mogućnosti</b>	<b>6</b>
<b>8</b>	<b>Osnovne osobine ovog programskog jezika, podržane paradigme i koncepti</b>	<b>6</b>
<b>9</b>	<b>Najpoznatija okruženja (framework) za korišćenje ovog jezika i njihove karakteristike</b>	<b>6</b>

<b>10 Instalacija i uputstvo za pokretanje na Linux/Windows operativnim sistemima</b>	<b>6</b>
10.1 Instalacija neophodnih paketa . . . . .	6
<b>11 Primer jednostavnog koda i njegovo objašnjenje</b>	<b>6</b>
<b>12 Sve ono što je specifično i važno za sam taj programski jezik</b>	<b>7</b>
12.1 ... podnaslov . . . . .	7
12.2 ... podnaslov . . . . .	7
<b>13 Zaključak</b>	<b>7</b>
<b>Literatura</b>	<b>7</b>
<b>A Dodatak</b>	<b>7</b>

## 1 Uvod

Kada budete predavali seminarski rad, imenujete datoteke tako da sadrže redni broj teme, temu seminarskog rada, kao i prezimena članova grupe. Precizna uputstva na temu imenovnja će biti data na formi za predaju seminarskog rada. Predaja seminarskih radova biće isključivo preko veb forme, a NE slanjem mejla. Link na formu će biti dat u okviru obaveštenja na strani kursa. Vodite računa da prilikom predavanja seminarskog rada predate samo one fajlove koji su neophodni za ponovno generisanje pdf datoteke. To znači da pomoćne fajlove, kao što su .log, .out, .blg, .toc, .aux i slično, **ne treba predavati**.

## 2 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.**

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 ...
```

## 3 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 3.1** *Problem zaustavljanja (eng. halting problem) je neodlučiv [?].*

**Primer 3.2** *Za prevođenje programa napisanih u programskom jeziku C može se koristiti GCC kompajler [?].*

**Primer 3.3** *Da bi se ispitivala ispravnost softvera, najpre je potrebno precizno definisati njegovo ponašanje [?].*

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 3.4** U odeljku ?? precizirani su osnovni pojmovi, dok su zaključci dati u odeljku 13.

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 ...

## 4 Slike i tabele

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

**Primer 4.1** 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 4.2** 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

## 5 Kôd i paket listings

Za ubacivanje koda koristite paket **listings**: [https://en.wikibooks.org/wiki/LaTeX/Source\\_Code\\_Listings](https://en.wikibooks.org/wiki/LaTeX/Source_Code_Listings)

Primer ubacivanja koda

```

1000 # This program adds up integers in the command line
import sys
1002 try:
    total = sum(int(arg) for arg in sys.argv[1:])
1004     print 'sum =', total

```

```
1006 | except ValueError:  
      |     print 'Please supply integer arguments'
```

## 6 Nastanak i istorijski razvoj, mesto u razvojnom stablu, uticaji drugih programskih jezika

### 6.1 Nastanak i istorijski razvoj

Iako se pojam jezika Objective-C primarno vezuje za proizvode kompanije Epl (engl. Apple) – MAC OS X, iPhone itd, on je zapravo nastao mnogo pre njih. Njegove temelje postavili su Bred Koks i Tom Lav 1981. godine, težeći da nađu način za povećavanje produktivnosti programera. Te godine pojavio se novi, revolucionarni jezik Smalltalk, koji je tada unapredio koncept objektno-orijentisanog programiranja. U osnovi tog jezika bilo je posmatranje programa kao skupova objekata koji su mogli da komuniciraju jedni sa drugima dinamički pozivajući metode. To je omogućilo da se stanje programa menja pod uticajem korisnika. Koks je u ovoj ideji video mogućnost da se vrtoglavo ubrza pisanje programa, pošto su se u njemu mogle praviti biblioteke objekata i posle koristiti u drugim programima bez izmene. Međutim, Smalltalk je bio veoma spor i zahtevao je da se svi programi pišu i pokreću u posebnom okruženju. Tada se javila potreba za spajanjem objektno-orijentisanih ideja Smalltalk-a i brzog jezika C. Koks je 1983. objavio naučni rad u kom je predstavio objektno-orijentisani prekompilator (engl. Object-Oriented Precompiler). Da bi ovu svoju ideju izbacili na tržište, njih dvojica su osnovali kompaniju Stepstoun (engl. Stepstone), izmenili kompilator za OOPC i preimenovali jezik u Objective-C. Iako je glavna ideja ovog projekta bila pisanje i prodavanje pomenutih biblioteka koje su mogle ponovo da se koriste, korisnici su imali dosta zamerki, te je jezik sve više poprimao karakteristike objektno-orijentisanih jezika kakve ih danas znamo, kao što su sakupljač otpada i interpreter, te je kompanija uskoro propala. Stiv Džobs je 1988. kupio licencu, a nekoliko godina kasnije i sva prava za Objective-C za potrebe svoje kompanije NeXT Computer koja se 1996. pripojila Apple-u. Sa tom tranzicijom, sam jezik je pretrpeo razne promene, a najvažnije su uvođenje dveju novih „komponenti”: kategorija (engl. categories), danas poznatih kao ekstenzije (engl. extensions) i protokola (engl. protocols). Kategorije su služile da programerima omoguće da sami dodaju funkcije u objekte iz već postojećih biblioteka, a protokoli su omogućavali objektima da komuniciraju jedni s drugima. Oni su danas podržani u jeziku Java i poznati su nam kao interfejsi (engl. interfaces).

### 6.2 Mesto u razvojnom stablu

### 6.3 Uticaj drugih programskih jezika

Kao što je već navedeno, Objective-C je nastao kao kombinacija koncepta na kojima počiva Smalltalk utočenih u sintaksu jezika C. Isprva je čak funkcionisao tako što je kod prevođen na C i onda izvršavan. S druge strane, on je uticao na jezik Swift koji je razvio Epl. Swift se često naziva i „Objective-C bez C-a”.

## 7 Osnovna namena programskog jezika, svrha i mogućnosti

## 8 Osnovne osobine ovog programskog jezika, podržane paradigme i koncepti

## 9 Najpoznatija okruženja (framework) za korišćenje ovog jezika i njihove karakteristike

## 10 Instalacija i uputstvo za pokretanje na Linux/Windows operativnim sistemima

### 10.1 Instalacija neophodnih paketa

Da bismo mogli da kompajliramo programe napisane u Objective-C, potrebno je prvo da instaliramo par paketa. Za kompajliranje Objective-C programa koristi se gcc kompajler, uz jos par dodataka. Prvo sto treba da uradimo je da instaliramo gcc podrsku za Objective-C.

```
1000 $ sudo apt-get install gobjc
```

Zatim treba da instaliramo framework na kom se moderni Objective-C zasniva i bez kog ne bi imalo puno smisla raditi (iako je moguće). Taj framework se zove GNUstep i instalira se komandom

```
1000 $ sudo apt-get install gnustep
$ sudo apt-get install gnustep-devel
```

## 11 Primer jednostavnog koda i njegovo objašnjenje

Zatim, navedimo primer zastarelog (od verzije MAC 10.5), ali zanimljivog koda koji omogućava „imitiranje“ (engl. posing).

```
1000 #import <Foundation/Foundation.h>
1002 @interface MyString : NSString
1004 @end
1006 @implementation MyString
1008 - (NSString *)stringByReplacingOccurrencesOfString:(
    NSString *)target
    withString:(NSString *)replacement {
1010     NSLog(@"The Target string is %@",target);
    NSLog(@"The Replacement string is %@",replacement);
1012 }
```

```

1014 @end
1016 int main() {
    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc]
        init];
1018 [MyString poseAsClass:[NSString class]];
    NSString *string = @"Test";
1020 [string stringByReplacingOccurrencesOfString:@"a"
        withString:@"c"];

1022 [pool drain];
    return 0;
1024 }

```

## 12 Sve ono što je specifično i važno za sam taj programski jezik

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

### 12.1 ... podnaslov

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

### 12.2 ... podnaslov

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

## 13 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.

## A Dodatak

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