Објектно орјентисано програмирање

Владимир Филиповић
vladaf@matf.bg.ac.rs
Александар Картељ
kartelj@matf.bg.ac.rs

Типови података у Јави

Владимир Филиповић
vladaf@matf.bg.ac.rs
Александар Картељ
kartelj@matf.bg.ac.rs

Математички факултет
{vladaf,kartelj}@matf.bg.ac.rs

Типови података у Јави

- Тип података представља један од основних појмова у строго типизираном програмском језику.
- Тип у Јави има следеће карактеристике:
  1. Тип података одређује скуп вредности које могу бити додељене промењљивима или изразима.
  2. Над њима се могу извршавати одређене операције, односно функције.
  3. Тип промењљиве или израза може се одредити на основу изгледа или описа, а да није неопходно извршити неко израчунавање.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Типови података у Јави (2)

- Свака операција или функција реализује се над аргументима фиксираног типа.
  - Тип резултата се одређује према посебним фиксираним правилима.
- Увођењем типова података омогућава се да преводилац лако открије неисправне конструкције у језику.
  - Ово даље представља један вид семантичке анализе.
- Типови података доприносе:
  - прегледности програма,
  - лакој контроли операција од стране преводиоца
  - и већој ефикасности преведеног програма.
- У језику Јава се прави строга разлика између појединих типова и није дозвољено мешање типова.
  - На пример, целобројни тип не може да се третира као логички, што је у неким језицима дозвољено.

Математички факултет
{vladaf,kartelj}@matf.bg.ac.rs

Типови података у Јави (3)

- У језику Јава се нови типови података дефинишу преко већ постојећих.
- Дакле, унапред морају постојати некакви ***прости*** (примитивни, предефинисани) ***типови*** података, који немају компоненте.
- Новокреирани податак назива се објекат.
- Како се објектима приступа преко посебних променљивих, које се називају и референце,
  објектни тип се још назива и референцни тип.
- Према томе, у Јави разликујемо две врсте типова података:
  1. Примитивни и
  2. Објектни (или референцни)

```
<tip> :: = <primitivni tip>|<objektni tip>
```

Математички факултет
{vladaf,kartelj}@matf.bg.ac.rs

Примитивни типови података

- Примитивни тип је одређен:
  - скупом вредности које се формирају из одговарајућих литерала
  - и скупом операција над тим вредностима.
- То су унапред дефинисани типови у Јави и одмах стоје на располагању кориснику.
- Као примитивни типови појављују се:
  - бројеви (цели или реални),
  - знаковни тип
  - и логички тип.

```
<primitivni tip>::=<aritmetički tip> | boolean
<aritmetički tip> ::= <celobrojni tip>|<realni tip>
<celobrojni tip> ::= byte | short | int | long | char
<realni tip> ::= float | double
```

Математички факултет

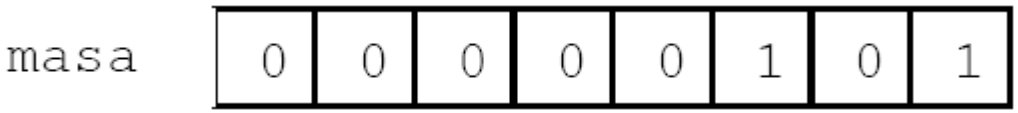
{vladaf,kartelj}@matf.bg.ac.rs

Примитивни типови података (2)

- Ако је нека променљива примитивног типа, она представља локацију у коју ће бити смештена примитивна вредност.
- Такве променљиве се још називају променљивима контејнерског типа. На пример, ако имамо декларацију

```
byte masa=5;
```

- У меморији рачунара постојаће локација којој је додељено име masa и која ће садржати вредност 5 у бинарном облику, као на следећој слици:



- У зависности од конкретног примитивног типа, величина меморијске локације може бити различита, али она ће увек садржати вредност примитивног типа.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Целобројни типови података

- У оквиру целобројних типова података можемо разликовати: byte, short, int, long и char.
- За сваки од тих типова постоји одређени интервал (одређен величином меморијске речи) из којег могу узимати вредности.
- На пример, податак типа byte се уписују у меморијску реч дужине осам ћелија (у потпуном комплементу) па су вредности из $[-2^7, 2^7-1]$.
- Сви целобројни типови, осим знаковног, могу имати негативне вредности.
- Цели бројеви су у Јави репрезентовани у формату потпуног комплемента.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Целобројни типови података (2)

- Следећа табела садржи интервале вредности за све целобројне типове.

Тип	Репрезентација	Интервал
`byte`	8-битни, означен број у потпуном комплементу	`-128 до 127`
`short`	16-битни, означен број у потпуном комплементу	`-32768 до 32767`
`int`	32-битни, означен број у потпуном комплементу	`-2147483648 до 2147483647`
`long`	64-битни, означен број у потпуном комплементу	`-9223372036854775808 до 9223372036854775807`
`char`	16-битни, неозначен број, Unicode	`'\u0000' до '\uffff'`

- Целобројни тип карактеришу следећи оператори:
  - аритметички
  - релациони
  - по битовима
- У Јави се аритметичке операције извршавају превођењем свих осталих целобројних типова у int или long.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Реални типови

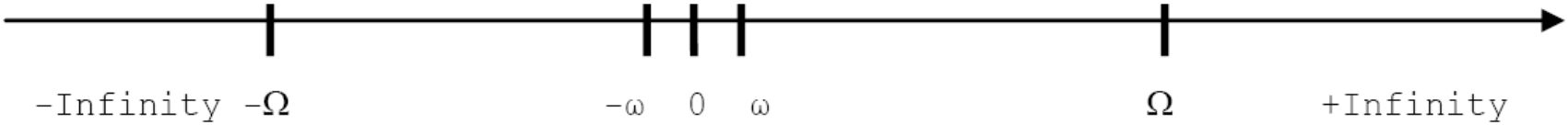
- У Јави постоје два реална типа:
  - float – једноструке тачности односно 32 бита
  - и double – двоструке тачности односно 64 бита
- Бројеви се записују према стандарду IEEE 754.
- Интервали из којих се могу представљати реални бројеви за оба типа приказани су у следећој табели:

Тип	Интервал
float	1.40239846e-45 до 3.40282347e+38
double	4.94065645841246544e-324 до 1.79769313486231570e+308

Математички факултет
{vladaf,kartelj}@matf.bg.ac.rs

Реални типови (2)

- За сваки од претходна два подтипа постоје:
  најмањи и највећи негативан реални број, нула, најмањи и највећи позитиван реални број.
- Стога реални тип можемо представити помоћу бројне осе на следећи начин:



- Овде су са Ω и ω означени, редом, максимални и минимални реалан број по апсолутној вредности у оквиру одговарајућег реалног типа.
- Реални бројеви из области (-∞, -Ω) не могу се регистровати.
- Ако је резултат неке операције из тог интервала, наступило је прекорачење (енг. overflow)
  - тај резултат третира се као -∞ (-Infinity).
- Слично важи за бројеви из области (Ω, +∞) само је (+Infinity).

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Реални типови (3)

- Реални бројеви из области (-ω, 0)∪(0, ω), такође, не могу бити регистровани.
- Ако је резултат неке операције из ове области, појављује се поткорачење (енг. underflow).
- За разлику од прекорачења, резултат се третира као нула зато што је реч о веома малим бројевима – блиским нули.
- Међутим, при оперисању са оваквим бројевима треба бити опрезан јер се могу добити некоректни резултати.
- Реални бројеви из области [-Ω, -ω]∪{0}∪[ω, Ω] могу се регистровати у Јави.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Реални типови (4)

- У ствари, тачно се могу регистровати само тзв. централни бројеви, а сви остали само приближно.
- Ако је x централни број, тада се сви реални бројеви (у математичком смислу), из довољно мале околине за x, замењују бројем x.
- На реални тип података могу да се примењују релациони и аритметички оператори.
- Приликом операција са реалним бројевима може се као резултат појавити нешто што није број.
- На пример, ако се нула дели нулом и стога постоји посебна вредност означена са NaN (енг. Not a number).

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Логички тип

- Логички (boolean) тип је окарактерисан:
  - скупом логичких константи **true** и **false** које не могу имати друго значење,
  - скупом логичких оператора и операторима једнакости и неједнакости.
- Логички тип је добио назив по имену енглеског математичара Була (George Boole, 1815-1864) који се сматра оснивачем математичке логике.
- Следеће наредбе у Јави:
  - **if, while, for, do-while**
  - и условни оператор **?:**

  захтевају логичке вредности за навођење услова.

Математички факултет
{vladaf,kartelj}@matf.bg.ac.rs

Објектни тип

- Појам објекта је кључан у сваком објектно оријентисаном језику.
- Објекат у себи обједињује скуп података и поступака за рад са тим подацима.

Објектни тип у Јави може бити:

- кориснички,
- низовни
- и набројиви.

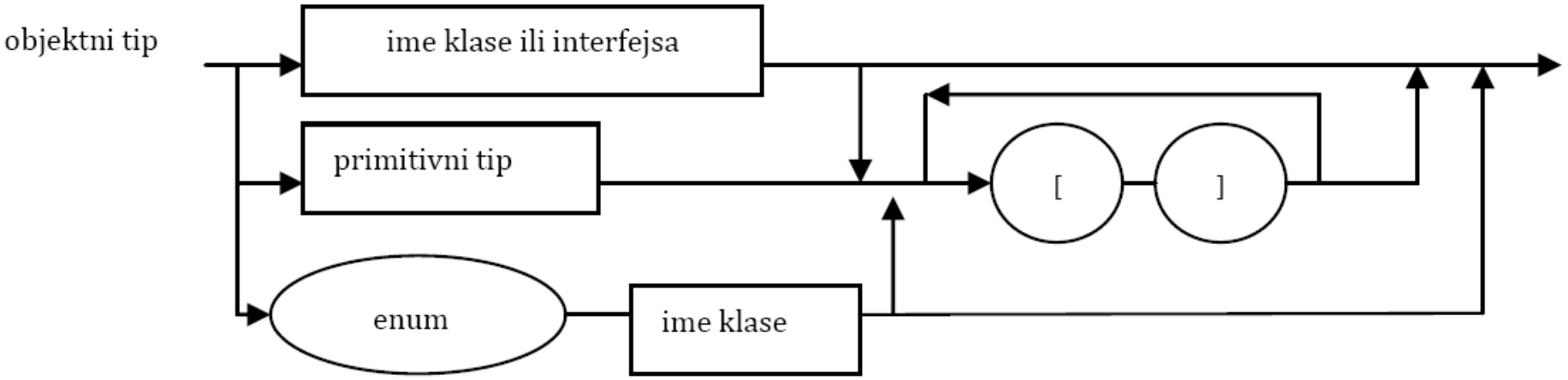
- Кориснички објектни тип дефинише сам корисник преко имена класе или имена интерфејса.
- Низовни тип се може дефинисати било преко корисничког објектног типа, било преко примитивног типа.
- Набројиви тип дефинише се преко кључне речи **enum** и имена класе.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Објектни тип (2)

- Дакле,



- Овде је :

```
<ime klase> ::= <identifikator>
<ime klase ili interfejsa> :: = <identifikator>
```

- У наставку презентације описујемо само кориснички објектни тип, а низовни и енумерисни тип ће бити описани касније.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Кориснички објектни тип

- Декларацијом класе практично се дефинише нови кориснички тип у Јави.
- Име класе може да се користи за декларисање променљивих, као што се код примитивних типова користе резервисане речи попут: int, boolean, double,…
- **Пример:** ако дефинишемо класу Figura на следећи начин:

```
class Figura { … }
```

тада има смисла декларисати променљиве:

```
Figura a, b;
```

- Дакле, наредбом Figura a, b; декларисане су две променљиве помоћу којих можемо приступати конкретним објектима класе Figura.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Кориснички објектни тип (2)

- Из постојеће класе може се креирати нова класа тако што ће имати све особине постојеће класе и неке додатне.

- Таква класа назива се ***поткласа*** постојеће класе, и пошто има све особине постојеће класе (***наткласе***), за њу се каже да је ***наследила*** постојећу класу.

- Дакле, сви креирани објекти поткласе имају особине постојеће класе и неке додатне које су наведене у поткласи.

- Механизам наслеђивања је, такође, битан за објектно оријентисане језике јер омогућава креирање нових класа из постојећих.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Кориснички објектни тип (3)

- Начин записа података објектног типа (објеката) у меморији се разликује од начина записа података примитивног типа.
- Подацима у меморији се приступа преко променљивих:
  - Код примитивних типова променљиве садрже податке са којима се оперише,
  - док су код објектног типа променљиве показивачи на објекте.
- **Пример:** ако декларишемо класу:

```
class Osoba{ …}
```

и креирамо примерак класе коришћењем променљиве p

```
Osoba p = new Osoba ();
```

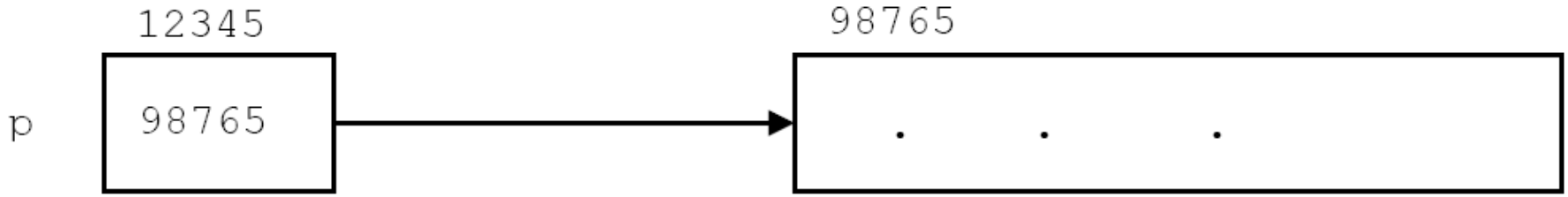
онда је p референца (показивач) на адресу у меморији од које почиње запис креираног објекта.

Математички факултет

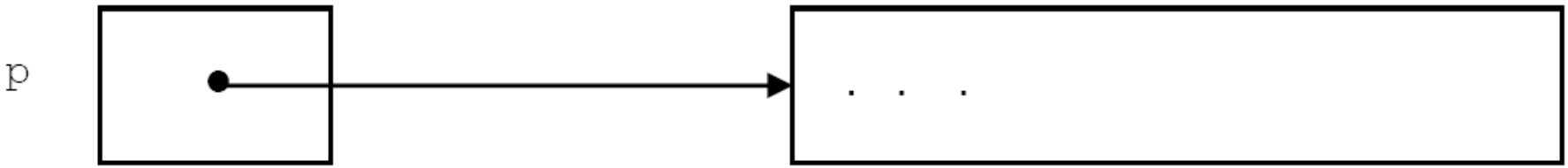
{vladaf,kartelj}@matf.bg.ac.rs

Кориснички објектни тип (4)

- Дакле, ако се за променљиву p користи локација са адресом 12345, а запис објекта почиње од адресе 98765, онда то графички представљамо на следећи начин:



- Све адресе у меморији изражене су у бинарном облику, али смо због прегледности овде користили декадне бројеве.
- Адресе су овде небитне па је однос променљиве p и објекта погодније приказати на следећи начин:



Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Кориснички објектни тип (5)

- Још једна суштинска разлика између објеката и података примитивног типа:
  - Објекти се креирају динамички, тек приликом извршавања програма, тада се за њих резервише меморија,
  - док се за податке примитивног типа меморијски простор резервише статички још у фази превођења.
- Позивање метода над објектом:
  - Над променљивама објектног типа се могу позивати методи припадајуће класе.
  - То се рати тако што се наведе име променљиве, затим пунктуални оператор (тј. тачка), а затим назив метода који се позива праћен аргументима позива између малих заграда.
  - Ако нема аргумената позива, тада се иза назива метода обавезно морају написати отворена и затворена мала заграда.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Кориснички објектни тип (6)

Пример

Претпоставимо да променљива **prvi** класе **Ucenik** показује на новонаправљени објекат.

Позив метода **stampajIme** који постоји у класи **Ucenik** над објектом на који реферише променљива **prvi** реализује се следећом наредбом:

```
prvi.stampajIme();
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 1

Изворни код за програм Zdravo svete!

```
/* ZdravoSveteApp.java */
class ZdravoSveteApp{
        /**
        * Tradicionalni program "Zdravo svete!".
        */
        public static void main (String args[]) {
                // Pisi na standardni izlaz.
                System.out.println("Zdravo svete!");
        }
}
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 1 (2)

- Чувамо претходни програм у датотеку **ZdravoSveteApp.java** која је смештена у директоријум **c:\vladofilipovic**

```
C:\vladofilipovic>dir

    Volume in drive C is ATHOME
    Volume Serial Number is 1CE3-2551
    Directory of C:\vladofilipovic

.               <DIR>          01-24-96 10:42p .
..              <DIR>          01-24-96 10:42p ..
HELLOW~1 JAV              265 01-22-96 3:38p ZdravoSveteApp.java
          1 file(s)                265 bytes
          1 dir(s)       348,585,984 bytes free
```

- Преводимо .java датотеку коришћењем програма **javac**

```
C:\vladofilipovic>javac ZdravoSveteApp.java
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 1 (3)

- Извршавамо .class датотеку **ZdravoSveteApp.class** из директоријума **c:\vladofilipovic** коришћењем интерпетатора тј. програма **java**, што доводи до приказа поруке на екрану тј. конзоли.

```
C:\vladofilipovic>java ZdravoSveteApp
Zdravo svete!
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 2

- Креирати класу Ucenik помоћу које се могу генерисати конкретни објекти тако да сваки садржи име ученика и разред.
- Поред тога, класа треба да садржи два метода:
  - један за штампање имена ученика,
  - а други за испитивање да ли се ученик бави спортом.
- У посебној класи креирати неколико примерака класе Ucenik.

```
class Ucenik {
        String ime;
        int razred;
        boolean baviSeSportom(String sport) {
                if (sport == null)
                        return false;
                return true;
        }
        void stampajIme() {
                System.out.println("Ime ucenika je: "+ime);
        }
}
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 2 (2)

- У класи **TestUcenik** тестирамо класу **Ucenik** тако што у main методу правимо неколико примерака
(main метод је могао да се позове и из класе **Ucenik**).

```
class TestUcenik {

        public static void main (String args []) {
                Ucenik prvi = new Ucenik();
                prvi.ime = "Petar Peric";
                Ucenik drugi; drugi = new Ucenik();
                drugi.ime= "Milan Mikic";
                drugi.razred= 2;
                prvi.stampajIme();
                System.out.println("Ucenik se bavi sportom: " +
                        prvi.baviSeSportom("kosarka"));
                drugi.stampajIme();
                System.out.println("Ucenik se bavi sportom: " +
                        prvi.baviSeSportom(null));
        }
}
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 2 (3)

- Конкретни објекти, тј. примерци су објекти на које указују променљиве **prvi** и **drugi**.
- Они се креирају уз помоћ оператора **new** и класе **Ucenik**.
- Преко наредбе **prvi.stampajIme();** послата је порука објекту **prvi**, а саопштење је други део поруке, тј. **stampajIme().**
- Након извршавања програма, добија се:

```
Ime ucenika je: Petar Peric
Ucenik se bavi sportom: true
Ime ucenika je: Milan Mikic
Ucenik se bavi sportom: false
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 3

- Формирати потклasу класе **Ucenik** под називом **Srednjeskolac** и проширити класу за тестирање креирајући и примерке поткласе.

```
class Srednjeskolac extends Ucenik {
        String vrstaSkole;
        int uzrast;
        String uzetiVrstuSkole() {
                return vrstaSkole;
        }
        void prepoznaje() {
                if (uzrast>20)
                        System.out.println("Ne zavrsava redovno skolu");
                else
                        System.out.println("redovan!");
        }
}
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 3 (2)

- Класа **Srednjeskolac** наслеђује (проширује) класу **Ucenik**, што је саопштено помоћу резервисане речи **extends**.
- То значи да примерци ове класе могу да користе све променљиве и методи из класе **Ucenik**.
- Поред овога, примерци класе **Srednjeskolac** имају и додатна својства: врста школе и узраст.
- Ту су и два метода у класи **Srednjeskolac** :
  - један служи за препознавање врсте школе,
  - а други казује да ли ученик редовно или ванредно похађа школу.

- У главном програму који следи креирају се примерци, тј. објекти класе **Srednjeskolac**.
- Видећемо да се поред променљивих и метода из класе **Srednjeskolac** може приступати и променљивима и методима из класе **Ucenik**.

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 3 (3)

```
class TestSrednjeskolac {
        public static void main (String args []) {
                Ucenik prvi = new Ucenik();
                prvi.ime = "Petar Peric";
                prvi.stampajIme();
                System.out.println("Ucenik se bavi sportom:" +
                prvi.baviSeSportom("kosarka"));
                System.out.println("==================================");
                Srednjeskolac sred1 = new Srednjeskolac();
                sred1.ime = "Ana Skovic";
                sred1.vrstaSkole = "Gimnazija";
                sred1.uzrast = 16;
                sred1.stampajIme();
                System.out.println ("Ime skole je: " +
                sred1.uzetiVrstuSkole());
                System.out.print("Ucenik je: ");
                sred1.prepoznaje();
                Srednjeskolac sred2 = new Srednjeskolac();
                sred2.ime="Marko Rodic";
                sred2.uzrast =22;
                sred2.stampajIme();
                sred2.prepoznaje();
        }
}
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Пример Јава програма 3 (4)

- Покретањем програма из класе TestSrednjeskolac добија се:

```
Ime ucenika je: Petar Peric
Ucenik se bavi sportom: true
============================
Ime ucenika je: Ana Skovic
Ime skole je: Gimnazija
Ucenik je: redovan!
Ime ucenika je: Marko Rodic
Ucenik ne zavrsava redovno skolu
```

Математички факултет

{vladaf,kartelj}@matf.bg.ac.rs

Захвалница

Велики део материјала који је укључен у ову презентацију је преузет из презентације коју је раније (у време када је он држао курс Објектно орјентисано програмирање) направио проф. др Душан Тошић.

Хвала проф. Тошићу што се сагласио са укључивањем тог материјала у садашњу презентацији, као и на помоћи коју ми је пружио током конципцирања и реализације курса.