



PREDMET

Osnove Java Programiranja

Čas 17-18

Kolekcija

Copyright © 2010 – UNIVERZITET METROPOLITAN, Beograd. Sva prava zadržana. Bez prethodne pismene dozvole od strane Univerziteta METROPOLITAN zabranjena je reprodukcija, transfer, distribucija ili memorisanje nekog dela ili čitavih sadržaja ovog dokumenta., kopiranjem, snimanjem, elektronskim putem, skeniranjem ili na bilo koji drugi način.

Copyright © 2010 BELGRADE METROPOLITAN UNIVERSITY. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the prior written permission of Belgrade Metropolitan University.

Oktober 2011.

SADRŽAJ

Kreiranje liste	3
Rezultat rada	3
Metode Liste	3
Brisanje Liste	4
Pozicija u listi	4
Lista u niz	4
Pomešana Lista	5
Sortirana Lista	5
Obrnuta lista	5
Lista sa duplikatima	5
Set – bez duplikata	6
Set – pokušaj dupliranja	6
Unmodifiable	6
Povezana lista	6
Dodavanje na početak i kraj	7
Spajanje dve kolekcije	7
Uklanjanje sa liste	7
Mapa	7
Štampa mape	8
Pronalaženje ključa	8
Pražnjenje sadržaja mape	8
Klasa Osoba	8

Čas 17-18

Kolekcija

Kreiranje liste

```
//ArrayList<Osoba> arrayList = new ArrayList<Osoba>();  
List<Osoba> arrayList = new ArrayList<Osoba>();  
Osoba osoba = new Osoba("Ivan", "Ivanović");  
arrayList.add(osoba);  
arrayList.add(new Osoba("Miroslava", "Novaković"));  
arrayList.add(new Osoba("Dragan", "Petričeveić"));  
arrayList.add(new Osoba("Sanja", "Sanovič"));  
stampa(arrayList, "Lista");  
Štampanje liste (kolekcije)  
private void stampa(Collection<Osoba> arrayList, String title) {  
  
System.out.println("-----" + title + "-----");  
for (Osoba o : arrayList) {  
System.out.println(o);  
}  
}
```

Rezultat rada

-----Lista-----

Osoba: Ivan Ivanović

Osoba: Miroslava Novaković

Osoba: Dragan Petričeveić

Osoba: Sanja Sanovič

Metode Liste

```
System.out.println("-----");  
System.out.println("arrayList.size() = " + arrayList.size());  
System.out.println("arrayList.contains(osoba) = " + arrayList.contains(osoba));  
System.out.println("arrayList.contains(osoba) = " + arrayList.contains(new Osoba("Ivan", "Ivanović")));  
-----
```

```
arrayList.size() = 4  
arrayList.contains(osoba) = true  
arrayList.contains(osoba) = true
```

Brisanje Liste

```
arrayList.clear();  
stampa(arrayList, "Nakon brisanja");  
arrayList.add(osoba);  
arrayList.add(new Osoba("Miroslava", "Novaković"));  
arrayList.add(new Osoba("Dragan", "Petričeveić"));  
arrayList.add(new Osoba("Sanja", "Sanovič"));  
stampa(arrayList, "Nakon vraćanja");
```

-----Nakon brisanja-----

-----Nakon vraćanja-----

Osoba: Ivan Ivanović

Osoba: Miroslava Novaković

Osoba: Dragan Petričeveić

Osoba: Sanja Sanovič

Pozicija u listi

```
System.out.println("-----");  
System.out.println("arrayList.indexOf(new Osoba(\"Dragan\",  
\"Petričeveić\")) = \" + arrayList.indexOf(new Osoba(\"Dragan\",  
\"Petričeveić\"));  
System.out.println("-----");  
System.out.println("arrayList.get(2) = \" + arrayList.get(2));
```

arrayList.indexOf(new Osoba("Dragan","Petričeveić")) = 2

arrayList.get(2) = Osoba: Dragan Petričeveić

Lista u niz

```
System.out.println("-----");  
Object[] nizOsoba = arrayList.toArray();  
for (Object o : nizOsoba) {  
    System.out.println((Osoba) o);  
}
```

Osoba: Ivan Ivanović

Osoba: Miroslava Novaković

Osoba: Dragan Petričeveić

Osoba: Sanja Sanovič

Pomešana Lista

```
Collections.shuffle(arrayList);  
stampam(arrayList, "Pomešana Lista");
```

-----Pomešana Lista-----

Osoba: Miroslava Novaković

Osoba: Ivan Ivanović

Osoba: Sanja Sanović

Osoba: Dragan Petričević

Sortirana Lista

```
//Potrebno Comparable interface na klasi Osoba  
// Collections.sort((List) arrayList);  
Collections.sort(arrayList);  
stampam(arrayList, "Sortirana Lista");
```

-----Sortirana Lista-----

Osoba: Ivan Ivanović

Osoba: Miroslava Novaković

Osoba: Dragan Petričević

Osoba: Sanja Sanović

Obrnuta lista

```
Collections.reverse(arrayList);  
stampam(arrayList, "Obrnuta Lista");
```

-----Obrnuta Lista-----

Osoba: Sanja Sanović

Osoba: Dragan Petričević

Osoba: Miroslava Novaković

Osoba: Ivan Ivanović

Lista sa duplikatima

```
arrayList.add(new Osoba("Sanja", "Sanović"));  
arrayList.add(new Osoba("Sanja", "Sanović"));  
stampam(arrayList, "Lista sa duplikatima");
```

-----Lista sa duplikatima-----

Osoba: Sanja Sanović

Osoba: Dragan Petričević

Osoba: Miroslava Novaković

Osoba: Ivan Ivanović

Osoba: Sanja Sanović

Osoba: Sanja Sanović

Set – bez duplikata

```
Set<Osoba> set = new TreeSet<Osoba>(arrayList);
```

```
stampaa(set, "Set - Nema duplikata, i sortirano");
```

-----Set - Nema duplikata, i sortirano-----

Osoba: Ivan Ivanović

Osoba: Miroslava Novaković

Osoba: Dragan Petričević

Osoba: Sanja Sanović

Set – pokušaj dupliranja

```
set.add(new Osoba("Sanja", "Sanović"));
```

```
set.add(new Osoba("Sanja", "Sanović"));
```

```
stampaa(set, "Set - pokušaj dupliranja");
```

-----Set - pokušaj dupliranja-----

Osoba: Ivan Ivanović

Osoba: Miroslava Novaković

Osoba: Dragan Petričević

Osoba: Sanja Sanović

Unmodifiable

```
Collections.unmodifiableSet(set);
```

```
set.add(new Osoba("Mitar", "Pekić"));
```

```
stampaa(set, "Nakon unmodifiable metode");
```

-----Nakon unmodifiable metode-----

Osoba: Ivan Ivanović

Osoba: Miroslava Novaković

Osoba: Mitar Pekić

Osoba: Dragan Petričević

Osoba: Sanja Sanović

Povezana lista

```
LinkedList<Osoba> linkedList = new LinkedList<Osoba>();
```

```
linkedList.add(new Osoba("Boško", "Klepić"));
```

```
linkedList.add(new Osoba("Aleksandra", "Dragić"));
```

```
stampaa(linkedList, "Povezana lista");
```

-----Povezana lista-----

Osoba: Boško Klepić

Osoba: Aleksandra Dragić

Dodavanje na početak i kraj

```
linkedList.addFirst(new Osoba("Ivana", "Novak"));  
linkedList.addLast(new Osoba("Snežana", "Tošić"));  
stampala(linkedList, "Dodavanje na početak i na kraj");
```

-----Dodavanje na početak i na kraj-----

Osoba: Ivana Novak

Osoba: Boško Klepić

Osoba: Aleksandra Dragić

Osoba: Snežana Tošić

Spajanje dve kolekcije

```
linkedList.addAll(set);  
stampala(linkedList, "Spojene kolekcije");
```

-----Spojene kolekcije-----

Osoba: Ivana Novak

Osoba: Boško Klepić

Osoba: Aleksandra Dragić

Osoba: Snežana Tošić

Osoba: Ivan Ivanović

Osoba: Miroslava Novaković

Osoba: Mitar Pekić

Osoba: Dragan Petričević

Osoba: Sanja Sanović

Uklanjanje sa liste

```
linkedList.removeFirst();  
linkedList.remove(4);  
linkedList.removeFirst();  
linkedList.removeLast();  
stampala(linkedList, "Nakon uklanjanja");
```

-----Nakon uklanjanja-----

Osoba: Aleksandra Dragić

Osoba: Snežana Tošić

Osoba: Ivan Ivanović

Osoba: Mitar Pekić

Osoba: Dragan Petričević

Mapa

```
Map<String, Osoba> map = new HashMap<String, Osoba>();  
map.put("ivan", osoba);
```

```
map.put("alex", linkedList.getFirst());  
map.put("sneza", linkedList.get(2));  
stampaMape(map, "Kreirana Mapa");
```

-----Kreirana Mapa-----

Osoba: Ivan Ivanović

Osoba: Ivan Ivanović

Osoba: Aleksandra Dragić

Štampa mape

```
private void stampaMape(Map map, String title) {  
    System.out.println("-----" + title + "-----");  
    for (Object s : map.keySet()) {  
        System.out.println(map.get(s));  
    }  
}
```

Pronalaženje ključa

```
System.out.println("-----");  
System.out.println("map.containsKey(\"alex\") = " +  
map.containsKey("alex"));  
System.out.println("map.containsKey(\"pera\") = " +  
map.containsKey("pera"));
```

map.containsKey("alex") = true

map.containsKey("pera") = false

Pražnjenje sadržaja mape

```
System.out.println("-----");  
System.out.println("map.isEmpty() = " + map.isEmpty());  
map.clear();  
System.out.println("-----Nakon pražnjenja-----");  
System.out.println("map.isEmpty() = " + map.isEmpty());
```

map.isEmpty() = false

-----Nakon pražnjenja-----

map.isEmpty() = true

Klasa Osoba

```
public class Osoba implements Comparable<Osoba>{  
    private String ime;  
    private String prezime;  
    public Osoba() {
```



```
}  
public Osoba(String ime, String prezime) {  
    this.ime = ime;  
    this.prezime = prezime;  
}  
public Osoba(Osoba o) {  
    this.ime = o.getIme();  
    this.prezime = o.getPrezime();  
}  
public String getIme() {  
    return ime;  
}  
public void setIme(String ime) {  
    this.ime = ime;  
}  
public String getPrezime() {  
    return prezime;  
}  
public void setPrezime(String prezime) {  
    this.prezime = prezime;  
}  
@Override  
public boolean equals(Object obj) {  
    if (obj == null) {  
        return false;  
    }  
    if (getClass() != obj.getClass()) {  
        return false;  
    }  
    final Osoba other = (Osoba) obj;  
    if (this.ime != other.ime && (this.ime == null ||  
    !this.ime.equals(other.ime))) {  
        return false;  
    }  
    return true;  
}  
@Override  
public String toString() {  
    return "Osoba: " + ime + " " + prezime;  
}
```

```
@Override  
public int hashCode() {  
    int hash = 7;  
    hash = 47 * hash + (this.ime != null ? this.ime.hashCode() :  
    0);  
    hash = 47 * hash + (this.prezime != null ?  
    this.prezime.hashCode() : 0);  
    return hash;  
}  
public int compareTo(Osoba o) {  
    if(this.prezime.compareTo(o.getPrezime())!=0)  
        return this.prezime.compareTo(o.getPrezime());  
    else  
        return this.ime.compareTo(o.getIme());  
}  
}
```