|  |
| --- |
| ***Teme:***  ***Rekurzivni postopki, rekurzivno definiran strukture (seznami: linearni, krožni, enostransko povezani)***   * *Dinamične strukture, postopki na strukturah* * *OOP : razredi, notranji razredi, raba in implementacija vmesnika* |

|  |
| --- |
| ***Dan je del implementacije in poskus rešitve problema v LinEnPovSez:***  /\*\*  \* interface ShowableList  \*/  public interface ShowableList {  public void showSezLast2First( LinEnPovSez.Node first );  public void showSezFirst2Last( LinEnPovSez.Node first );  }  public class LinEnPovSez implements ShowableList{    private Node prvi = null;  public LinEnPovSez() {  }    public void showSezLast2First( Node first ){  if ( first != null){  showSezLast2First(first.next);  System.out.print(first.value);  }  }    public void showSezFirst2Last( Node first ){  if ( first != null){  showSezLast2First(first.next);  System.out.print(first.value);  }  }    } |

**Naloga 1**

LinEnPovSez je implementacija urejenega, linearnega, enostransko povezanega seznama. Dokončajte realizacijo seznama, da bo :

1. metoda dodaj("krokodil") dodala 8 novih elementov seznama; vsak element bo vseboval eno črko s parametrom podane besede;
2. metoda dodaj('o') dodala en sam nov element seznama, vsebina elementa bo dana črka;
3. metoda normaliziraj() odstranila duplikate iz seznama, če ti obstajajo;
4. metoda soDuplikati() ugotovila, ali so v seznamu duplikati;
5. metoda prestejDuplicirane() ugotovila, koliko zapisov seznama je 'dupliciranih' po pravilu "deeehllzzzz" -> 3 (e,l,z so duplicirani);
6. metoda preštej duplikate() preštela duplikate po pravilu "deeehllzzzz" -> 6 (e-2x, l-1x, z-3x);
7. Izrišite razredni diagram hierarhije razredov, ki jo dobite. Razredni diagrami naj vsebujejo VSE metode in lastnosti razredov ne glede na njihove kvalifikatorje dostopov.

**Naloga 2**

Izvedite modificirano verzijo realizacije predhodne naloge brez uporabljenega vmesnika ShowableList, katerega metode naj nadomestijo metode, ki iz vsebine seznam formirajo nize (ekvivalentne izpisom metod iz ShowableList) in nize pri klicu vrnejo.

**Naloga 3**

KroEnPovSez naj bo krožen, enostransko povezan urejen seznam, pri katerem pa kazalec na ‘začetek’ seznama ni nujno tudi najmanjši element v seznamu. Struktura seznama je v zasnovi enaka kot pri linearni varianti, le da je naslednik zadnjega elementa ‘začetni’ element seznama (zadnji ne kaže na null, temveč na prvega). Seznam naj implementira ShowableList. Na takem seznamu realizirajte vse metode, kot so dane v Nalogi 1 te vaje. Dodatno implementirajte metode:

* showCurrent() prikaže element, ki je referenciran s kazalcem ‘prvi’ (glej strukturo razreda LinEnPovSez).
* next() postavi referenco ‘prvi’ na naslednji element v seznamu
* home() postavi referenco ‘prvi’ na najmanjši element v seznamu

Preverite delovanje metod (metodi iz vmesnika znata biti koristni), in implementaciji razreda dodajte še metodi:

* showSezLast2FirstOrdered
* showSezFirst2LastOrdered

ki izpišeta urejeno zaporedje elementov vrste ne oziraje se na trenutno pozicijo reference ‘prvi.