|  |
| --- |
| ***Teme:***  ***Rekurzivni postopki***  *Načrtovanje:*   * *UML diagrami: class, use-case,activity,* * *Konstrukcija diagramov, obratno-inženirstvo (diagram iz kode),*   *Vizualizacija*   * *Načrtovanje vizualizacije, konstrukcija elementov, vizualizacija vozlišč drevesa,*   *Dinamične strukture :*   * *Postopki na strukturah, vizualizacija struktur, primerjava struktur,* |

|  |
| --- |
| Opomba k realizaciji:  Izhodišča za realizacijo se nahajajo v datotekah s teoretično vsebino repozitorija Dokumenti/vsebine, oštevilčene z ***02\_03 do 02\_07.*** Realizacije dreves se nahajajo v ***02\_07\_dvojiškaDrevesa.***  Kot zgled uporabite tudi datoteko ***02\_04\_qSort\_bSearch\_customRecord.zip*** |

**Naloga 1**

Oglejte si realizacijo dvojiškega iskalnega drevesa (BST) in izvedite:

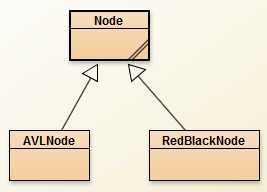
1. Notranji razred ***Node*** izločite iz definicije razreda, in naj bo samostojen razred
2. K metodi ***display(Node),*** ki predstavlja vmesni način (Inorder) izpisa vrednosti vozlišč drevesa, dopišite dve novi metodi: ***displayPreOrder(Node)*** in ***displayPostOrder(Node).*** Imena povedo njihov namen.
3. Odstranite glavno metodo iz razreda in spišite dodaten demonstracijski razred MaDoBST, ki bo demonstriral rabo BST drevesa (glej primere izvedb v demonstracijah ***02\_04\_qSort\_bSearch\_customRecord.zip***)
4. 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**

Kot ste izvedli z BST naredite tudi z razredom v katerem je implementirano AVL drevo.

**Naloga 3**

Primerjajte razredne diagrame, dobljene v nalogah 2 in 3. Opazite lahko, da je struktura obeh dreves skoraj enaka, da so mehanizmi dela več ali manj enaki. Izhodišče za realizacijo naloge je naslednji: AVL drevo je zgolj posebna izvedba navadnega BST drevesa: vozlišče je rahlo drugačno, metode (predvsem dodajanja) se funkcionalno rahlo razlikujejo. V osnovi bi lahko rekli, da je AVL drevo zgolj specializacija BST drevesa (dedovanje). Ker pa (še) ne poznamo abstrakcije podatkov (generičnost?) boste izvedli zgolj posplošitev vozlišča drevesa:

pri tem naj bosta AVL node in RedBlackNode zgolj specializaciji razreda Node (uporabite dedovanje).

Slika 1 Nova hierarhija vozlišč dreves

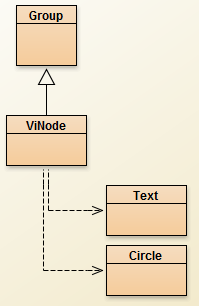
Po transformaciji hierarhije vozlišč morajo demonstracije vseh treh dreves še vedno delovati, kot so delovale pred začetkom izvedbe prve naloge.

Če je potrebno, preoblikujte strukturo RedBlack drevesa v skladu z navodili v nalogi 1.

**Naloga 4**

Za demonstracijo dreves do sedaj uporabljamo 3 demonstracijske programe, hkrati pa je demonstracija slaba. Izvedite demonstracijski program, ki bo omogočal vizualizacijo enega od izbranih dreves. Vizualizacija naj bo grafična, izvedena s tehnologijo JavaFX

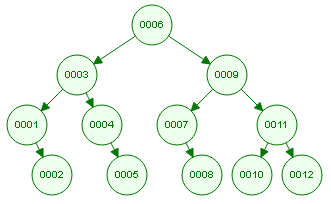
Vozlišča drevesa naj bodo vizualizirana z razredom ViNode:

ViNode predstavlja skupino (javafx.scene.Group) v kateri sta krogec (Circle) in besedilo v krogcu (Text). Besedilo naj bo centirano v krogcu.

Slika 2Hierarhija vizualizacije

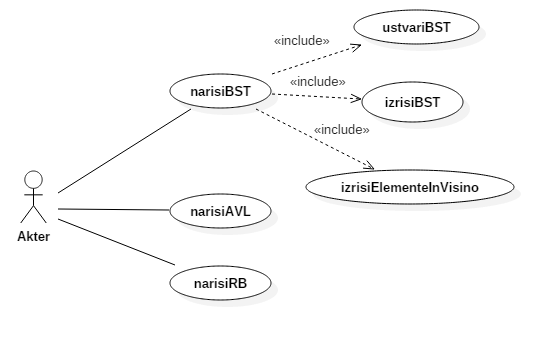
Predlagam, da namesto Group uporabite javafx.scene.layout.StackPane na katerega postavite Circle in Text. Ugotovite zakaj !

Posamezno drevo se mora izrisati v celoti (z vsemi vozlišči in njihovimi vrednostmi), da bo videti kot drevo (glej demonstracije, teorijo) oz.naslednjo sliko:



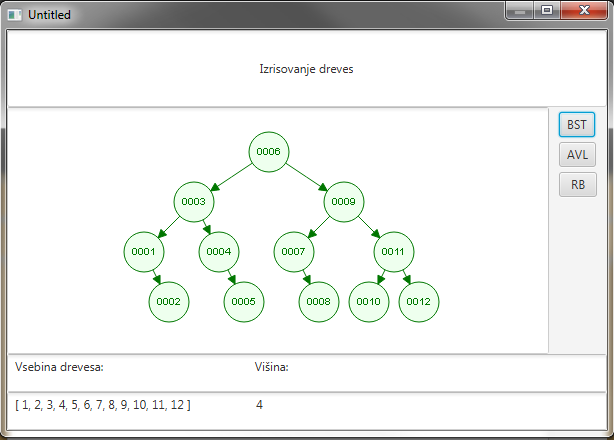
Slika 3Predlog izgleda drevesa

Sama aplikacija naj vsebuje mehanizme, ki omogočajo izbiro enega izmed dreves, ki bi ga želeli izrisati.

Diagram je v celoti podan zgolj za BST drevo, vendar enako izvedete tudi za Avl in RB drevo.

Slika 4Način rabe aplikacije

Zaželen uporabniški vmesnik je podan spodaj.



Slika 5Primer uporabniškega vmesnika

**Naloga 5**

Izdelajte[[1]](#footnote-1) celoten razredni diagram aplikacije, izrišite diagram aktivnosti za vstavljanjeBST, ter diagram aktivnosti za vstavljanjeAVL.

1. StarUML, <http://staruml.io/>, okt.2016 (risanje, forward, reverse engineering)

   MS Visio

   Gliffy, <https://www.gliffy.com/uses/uml-software/>, okt.2016

   Creately, <http://creately.com/Draw-UML-and-Class-Diagrams-Online>, okt.2016

   yUML, <http://yuml.me/,okt.2016>

   Enterprise Architect, <http://www.sparxsystems.com/products/ea/>, okt.2016 (risanje, f+r engineering)

   Visual Paradigm for UML, <http://www.visual-paradigm.com/>, okt.2016 (risanje, forward, reverse engineering) [↑](#footnote-ref-1)