public class JazelAVL

{

public int info;

public JazelAVL left;

public JazelAVL right;

int visina;

public JazelAVL() {}

public JazelAVL(int i) { info=i; }

public int vis(JazelAVL p) {

if (p==null)

return -1;

else

return p.visina;

}

public int maks(int a, int b) {

if (a>b) return a;

else return b;

}

public void pecatiJazel() {

System.out.print(info);

}

} // end class Jazel

--------------------------

import java.io.\*;

import java.util.\*;

class TreeAVL extends JazelAVL

{

public TreeAVL() {}

/\* Ovaa funkcija moze da se povika samo ako k2 ima levo dete \*/

/\* Pravi rotacija pomegu jazel (k2) i negovoto levo dete \*/

/\* Se azuriraat visinite, a potoa se vraka noviot koren \*/

public JazelAVL singleRotateWithLeft(JazelAVL k2)

{

JazelAVL k1;

k1=k2.left;

k2.left=k1.right;

k1.right=k2;

k2.visina=maks(vis(k2.left), vis(k2.right))+1;

k1.visina=maks(vis(k1.left), k2.visina)+1;

return k1; //noviot koren

}

/\* Ovaa funkcija moze da se povika samo ako k1 ima desno dete \*/

/\* Pravi rotacija pomegu jazel (k1) i negovoto desno dete \*/

/\* Se azuriraat visinite, a potoa se vraka noviot koren \*/

public JazelAVL singleRotateWithRight(JazelAVL k1)

{

JazelAVL k2;

k2=k1.right;

k1.right=k2.left;

k2.left=k1;

k1.visina=maks(vis(k1.left), vis(k1.right))+1;

k2.visina=maks(vis(k2.right), k1.visina)+1;

return k2; //noviot koren

}

/\* Ovaa funkcija moze da se povika samo ako k3 ima levo dete \*/

/\* i levoto dete na K3 ima desno dete \*/

/\* Se pravi leva – desna rotacija \*/

/\* se vraka noviot koren \*/

public JazelAVL doubleRotateWithLeft(JazelAVL k3)

{

/\* Rotacija pomegu K1 i K2 \*/

k3.left = singleRotateWithRight( k3.left );

/\* Rotacija pomegu K3 i K2 \*/

return singleRotateWithLeft( k3 );

}

/\* Ovaa funkcija moze da se povika samo ako k3 ima desno dete \*/

/\* i desnoto dete na K3 ima levo dete \*/

/\* Se pravi desna –leva rotacija \*/

/\* se vraka noviot koren \*/

public JazelAVL doubleRotateWithRight(JazelAVL k3 )

{

/\* Rotate between k1 and k2 \*/

k3.right = singleRotateWithLeft( k3.right );

/\* Rotate between k3 and k2 \*/

return singleRotateWithRight( k3 );

}

public JazelAVL insert(int x, JazelAVL t)

{

if( t == null )

{

/\* Kreiraj I vrati steblo so eden jazel \*/

t = new JazelAVL() ;

t.info = x; t.visina = 0;

t.left = t.right = null;

}

else

if( x < t.info )

{

t.left = insert( x, t.left );

if (( vis( t.left ) - vis( t.right )) == 2 )

if( x < t.left.info )

t = singleRotateWithLeft( t );

else

t= doubleRotateWithLeft( t );

}

else

if ( x > t.info ) {

t.right = insert( x, t.right );

if( vis( t.right ) - vis( t.left ) == 2 )

if( x > t.right.info )

t = singleRotateWithRight( t );

else

t =doubleRotateWithRight( t ); }

/\*Else x e vo stebloto; ne pravime nisto \*/

t.visina = maks(vis(t.left),vis(t.right))+1;

return t;

}

public void inorder(JazelAVL r)

{

if (r!=null)

{

inorder(r.left);

System.out.print(r.info+" ");

inorder(r.right);

}

}

} // end class

import java.io.\*;

import java.util.\*;

class TreeAVLApp

{

public static void main(String[] args) throws IOException

{

JazelAVL p=null;

TreeAVL t= new TreeAVL();

p=t.insert(15,p);

p=t.insert(6,p);

p=t.insert(18,p);

p=t.insert(3,p);

p=t.insert(2,p);

p=t.insert(4,p);

p=t.insert(7,p);

p=t.insert(13,p);

p=t.insert(9,p);

p=t.insert(17,p);

p=t.insert(20,p);

t.inorder(p);

}

}