Visualization of splay trees

Performed by **Vladimir Yanovsky**

Supervisior: **Prof Alon Itai**

Introduction.

In the project we have created a visual aid that should help to understand changes in balanced trees during operations on them through visualization.

Java applet that we created implements a splay tree with operations find, insert and delete, although thanks to the OOD of the program it is quite easy to change this applet such that it implements any other kind of balanced trees.

For convenience, the applet has a button suspend (resume) to suspend (resume) the animation in the middle, and a button for changing the speed of the animation . Values for speed are from 0 - the fastest to 99 - the slowest. Also we limited the values of the nodes to be from 0 to 99.

Algorithms and Data Structures.

The basic data structure is node.

Class Node has a function to draw itself as a circle with one or two decimal digits inside.

Class Tree is a basic binary tree. It has a field of class ***Node*** - the root, and has 3 fields of type ***Tree*** - the sons and the father.

The next class in the hierarchy is ***BoundBoxedTree*** - subclass of ***Tree*** which has fields used for drawing a static tree using the idea of bounding boxes. In the root of each tree we store its width that is computed recursively as follows :
this.width = left.width + margin + diameter_of_the_node + margin + right.margin (the cases when one of the sons is empty are taken into account in the program.)

To animate a tree, first, after each splay step we draw the tree topologically correct by rearranging edges as needed. Next, for every node of this new tree, we compute the best position accordingly to width of its sons and nodes depth. After this, all the nodes of the tree are moved from the previous location to the just computed one by straight line (actually, it is a broken line which approximate the straight line). When this moving is finished, the tree is ready for the next splay step. Moving of a point by a straight line is implemented in class ***MovingPoint***.

Class Splay contains, in addition to ***BoundBoxedTree*** few functions needed to manipulate the splay trees - rotations, insert, delete and splay.

In class ***MyCanvas*** we implement animation of all operations of on splay trees using classes describe above. We define here the variables ***sleeptime*** that is used to regulate the speed of animation and ***pausetime*** to regulate pauses between two splay steps.

Class ***Control*** is an applet with few controls - buttons for operation on splay trees and textfield for keys.