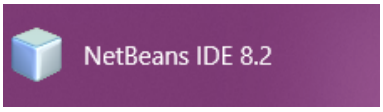
Nombre de la práctica	**Arboles AVL**			**No.**	
Asignatura:	**Lenguajes y Autómatas II**	**Carrera:**	**Ingeniería en Sistemas Computacionales**	**Duración de la práctica (Hrs)**	

NOMBRE DEL ALUMNO: Ana Karen Martínez Carpio
GRUPO: 3061

BALANCEO DE ÁRBOLES AVL

- Es un árbol binario de búsqueda en el cual se cumple: "Para todo nodo T del árbol, la altura de los subárboles izquierdo y derecho no debe diferir en más de una unidad".
- Son llamados árboles AVL en honor a sus inventores G.M. AdelsonVelskii y E.M. Landis.
- La idea central de éstos es la de realizar reacomodos o balanceos, después de inserciones o eliminaciones de elementos.

1. La realización de esta practica es en Java, en el programa NetBeans.



2. Se creo un nuevo proyecto y dentro de este se creo una clase

3. Primero se colocó una clase, en donde se declaran unas variables, así como asignar valores a los nodos que se van a utilizar más adelante.

```
26      class Node {
27          int N;
28          int altura = 0;
29          Node parent=null,izquierda  = null, derecha = null;
30
31          Node(int n, int h, Node P) {
32              N = n;
33              altura = h;
34              parent = P;
35          }
36      }
37      Node root = null;
38
```

4. Posteriormente se creo el primer método en donde se va a realizar la inserción de los nodos, en este se encuentran las condiciones simples de como debe funcionar(derecha, izquierda)

```
39      void insert(int N, Node R){
40          if (root == null){
41              root = new Node(N, 1, null);
42          }else{
43              if (N <= R.N) {
44                  if (R.izquierda == null) {
45                      R.izquierda = new Node(N, 1, R);
46                      dobalance(R.izquierda);
47                  } else
48                      insert(N, R.izquierda);
49              }else{
50                  if (R.derecha == null) {
51                      R.derecha = new Node(N, 1, R);
52                      dobalance(R.derecha);
53                  } else
54                      insert(N, R.derecha);
55              }
56          }
57      }
```

5. Posteriormente se crea un nuevo método, en el cual define la dirección, así como la altura de los nodos, de igual manera evalúa los valores en de los nodos, par poder realizar la rotación doble según sea el caso(Las rotaciones dobles son para equilibrar el árbol)

```
Node dobalance(Node x){
    int h1=0,h2=0;
    Node n1=null,rent=null;
    Node y=x,z;
    while(y.parent!=null){
            if(y.parent.izquierda ==y)n1=y.parent.derecha;
            else if(y.parent.derecha ==y)n1=y.parent.izquierda;
            h1=y.altura;
            if(n1 == null)
                    h2=0;
            else
                h2= n1.altura;
                if(Math.abs(h2-h1)>1)
                break;
                y.parent.altura=1+Math.max(h1,h2);
                y=y.parent;
    }

    if(y.parent == null)
        return null;
    z=y.parent;
    rent=z;

    h1=(z.izquierda == null)?0:z.izquierda.altura;
    h2=(z.derecha == null)?0:z.derecha.altura;
    if(h1<h2)
        y=z.derecha;
     else
    y=z.izquierda;
    h1=(y.izquierda == null)?0:y.izquierda.altura;
    h2=(y.derecha == null)?0:y.derecha.altura;
    if(h1<h2)
        x=y.derecha;
        else
    x=y.izquierda;

    y.parent=z;
    x.parent=y;
    if(z.izquierda == y){
        if (y.izquierda ==x)
            Rrotacion(y,z);
        else{
            Lrotacion(x,y);
            x.altura++;
            Rrotacion(x,z);
        }

    }else{

        if(y.derecha==x){
            Lrotacion(y,z);
        }else{
            Lrotacion(x,y);
            x.altura++;
            Rrotacion(x,z);
        }
    }
    return rent;
}
    void Rrotacion (Node y, Node z){
```


*Cualquier documento no identificado como **Controlado** se considera **COPIA NO CONTROLADA** y no es auditable.*

6. Después se agregaron otros métodos que nos permiten organizar las rotaciones adecuadas.

```
void Rrotacion (Node y, Node z){
    y.parent=z.parent;
    if(y.parent==null)
        root=y;
    else if (y.parent.izquierda==z)
        y.parent.izquierda=y;
    else
        y.parent.derecha=y;
    z.izquierda=y.derecha;
    if(z.izquierda!=null)
        z.izquierda.parent=z;
    y.derecha=z;
    z.parent=y;
    z.altura--;
}
```

```
void Lrotacion(Node y, Node z){
    y.parent=z.parent;
    if(z.parent == null)
        root=y;
    else if (z.parent.izquierda==z)
        y.parent.izquierda=y;
    else
        y.parent.derecha=y;
    z.derecha=y.izquierda;
    if(z.derecha!=null)
        z.derecha.parent=z;
    y.izquierda=z;
    z.parent=y;
    z.altura--;
}
```

7. El siguente metódo nos permitira organizar las lineas ya que deben de colocarse según se coloque el nodo, ya que es la conexión entre el nodo Padre y los nodos hijos.

```
Node search(int N, Node r){
    if(r==null) return null;
    if(N==r.N)
        return r;
    else if(N<r.N)
        return search(N, r.izquierda);
    else
        return search(N, r.derecha);
}
```

8. Posteriormente se crea una nueva clase estática llamada **Drawtree** con una extensión de Applet(Son programas Java que pueden ejecutarse de forma gráfica de forma similar a los programas de escritorio, es decir, utilizando formularios y colocando de forma visual los elementos que van a formar parte del programa), que nos permitirá crear un grafico para representar el árbol que se ha creado, conforme a las inserciones, aquí solo se crean variables, en donde se le asignas valores.

```
static class Drawtree extends Applet{
    private static final long serialVersionUID = -7654352523443329890L;
    final Color bg = Color.white;
    final Color fg = Color.BLACK;
    final BasicStroke stroke = new BasicStroke (2.0f);
    final BasicStroke wideStroke = new BasicStroke (8.0f);
    Dimension totalSize;
    int altura,width;
    Node r=null;
```

9. De igual manera se encuentra otro método en donde se indica el color que va a tener el Applet, asi como su tamaño.

```
public void init(Node N, int x){
    setBackground(fg);
    setForeground(bg);
    r=N;
    width=x;
}
Graphics2D g2;
```

10. Este método nos sirve para dibujar el grafico y que se reconozca como tal

```
public void paint(Graphics g){
    g2 = (Graphics2D) g;
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_ON);
    getSize();
    inorder(r,0,width,80);
}
```

11. Ese método nos ayudara a dar diseño al gráfico, ya que en este se esta indicando de que color y tamaño será los círculos(nodos), de igual manera el grosor que tendrá la línea(aquí se están utilizando las variables andes declaradas).

```
public void draw(int x1, int x2, int y,String n, int d){
    g2.setStroke(stroke);
    g2.setPaint(Color.white);
    int x=(x1+x2)/2;
    if (d ==1)
    g2.draw(new Line2D.Double (x2, y-30,x+15,y));
    else if(d==2)
    g2.draw(new Line2D.Double(x+15,y,x1+30, y-30));
    g2.setPaint(Color.green);
    Shape circle=new Ellipse2D.Double((x1+x2)/2,y,30,30);
    g2.draw(circle);
    g2.fill(circle);
    g2.setPaint(Color.black);
    g2.drawString(n, x+10, y+18);
}
```

En este método indica de qué manera se van a dibujar los nodos ingresados

```
void inorder(Node r, int x1, int x2,int y){
    if(r==null) return;
        inorder(r.izquierda,x1,(x1+x2)/2,y+40);
        if(r.parent==null)
            draw(x1,x2,y,r.N+"",0);
        else{
        if(r.parent.N<r.N)
            draw(x1,x2,y,r.N+"",2);
        else
            draw(x1,x2,y,r.N+"",1);
        }
            inorder(r.derecha,(x1+x2)/2, x2, y+40);
      }
      }
```


*Cualquier documento no identificado como **Controlado** se considera **COPIA NO CONTROLADA** y no es auditable.*

12. Por ultimo se coloca la método main(método Principal, el cual es el que se ejecuta), indicando que será de tipo Jframe por lo que se creara un Formulario, primero se colocan algunas propiedades al Jframe como es un título, su tamaño, la ubicación en que va a aparecer al ejecutarse, y posteriormente se esta creando un Panel, y dos botones (los cuales se les agrego un evento), estos botones se colocan en el panel, y el debemos de agregar el panel al Jframe para poder visualizarse.

```
static class Main extends JFrame implements ActionListener{

    private static final long serialVersionUID = -2829448395694197965L;

    public Main(){
        setTitle("Arbol ALV");
        setVisible(true);
        setSize(250,200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        panel1 = new JPanel();

        button0= new JButton("Insertar");
        button3 = new JButton ("Mostrar Arbol");
        button0.addActionListener(this);
        button3.addActionListener(this);

        panel1.add(button0);
        panel1.add(button3);

        Border b= BorderFactory.createEmptyBorder(45,0,0,0);
        panel1.setBorder (b);

        this.add(panel1);
        this.setVisible(true);

    }
```

En las líneas siguientes se está colocando los eventos que van a realizar los botones.

Si se presiona el boton0 el programa va a pedir que se ingrese un valor, el cual se va a guardar en una variable para su uso posterior, y así sucesivamente, pero si se presiona el botón 3, se va a crear un nuevo Jframe en donde se va a imprimir el grafico, mandando a llamar los métodos que se crearon anteriormente.

```
public void actionPerformed(ActionEvent e){
    if(e.getSource () == button0){
        String s=JOptionPane.showInputDialog("Ingrese el valor Entero");
        int i=Integer.parseInt(s);
        B.insert(i,B. root);

    }else if (e.getSource() == button3){
        JFrame f = new JFrame ("Grafic");
         f.addWindowListener(new WindowAdapter(){
              public void windowClosing (WindowEvent e){}
    });
        Drawtree applet = new Drawtree();
        f.getContentPane().add("Center",applet);
        Toolkit tk = Toolkit.getDefaultToolkit();
        int xSize = ((int) tk.getScreenSize().getWidth());
        applet.init(B.root,xSize-50);
        f.pack();
        f.setSize(new Dimension(xSize,500));
        f.setVisible(true);

    }
    }
```


*Cualquier documento no identificado como **Controlado** se considera **COPIA NO CONTROLADA** y no es auditable.*

FUNCIONAMIENTO

Se insertaran los siguientes valores: 10, 26, 32, 15, 6, 9, 3, 1, 5