LABORATORIO No. 8: Tablas de Símbolos Ordenadas y Balanceadas

Objetivos

Comprender la implementación del Tipo Abstracto de Datos Tabla de Símbolos Ordenadas y Balanceada (RBT) y su uso para la solución de problemas.

Al finalizar este laboratorio el estudiante estará en capacidad de:

- Utilizar la estructura de datos Tabla de Símbolos Ordenados y Balanceados para almacenar datos.
- Comprender el uso de los árboles binarios de búsqueda balanceados (RBTs) como estructura de datos y su impacto en los órdenes de crecimiento temporal.
- Proponer la forma de utilización de un árbol binario de búsqueda balanceado (RBTs) como alternativa de solución a un problema que implican el manejo de llaves ordenadas.
- Integrar los árboles binarios de búsqueda balanceados (RBTs) con las otras estructuras de datos vistas en el curso.

Fecha Límite de Entrega

Miércoles **27 de octubre** antes de la media noche (11:59 p.m.).

Preparación del Laboratorio

- Revisar el API del TAD Map ubicado en **`DISClib\ADT\orderedmap.py`**
- Revisar las estructuras de datos **`orderedmapstructure.py`** y **`rbt.py`** ubicado en **`DISClib\DataStructures`**

Trabajo Propuesto

PASO 1: Copiar el ejemplo en su organización

Copie/Haga **Fork** del repositorio del laboratorio en su organización con el procedimiento aprendido en las prácticas anteriores.

El repositorio del proyecto base que utiliza este laboratorio es el siguiente:

- https://github.com/ISIS1225DEVS/ISIS1225-SampleTree.git

Antes de clonar el repositorio en su computador diríjase a su organización (Ej.: ***EDA2021-1-SEC02-G01*** para el primer grupo de la sección 2 del curso) y cambie el nombre del repositorio de acuerdo con el

esquema **LabBalancedTrees-S<<XX>>-G<<YY>>** donde **XX** es el número de la semana de la práctica y donde **YY** es el número del grupo de trabajo. (Ej.: **LabBalancedTrees-S07-G01** para este **noveno laboratorio** hecho por el **grupo 1** de la **sección 2).**

Recuerde que **NO necesita** agregar la sección o el semestre en este nombre porque ya está identificado en su organización.

PASO 2: Descargar el ejemplo

Después de renombrar el proyecto dentro de su organización ya puede clonar el proyecto. Descargue el código en su máquina local **(`git clone`)** siguiendo lo aprendido en las practicas anteriores.

Recuerde modificar el **`README`** del repositorio para incluir los nombres de los integrantes del grupo.

PASO 3: Ejecutar y explorar el ejemplo

El proyecto **`SampleTree`** busca familiarizarlos con el TAD Tabla de Símbolos Ordenadas (Tree) y su uso para solucionar problemas y una forma de probar su desempeño en aplicaciones MVC.

Antes de iniciar a explorar y modificar el ejemplo, recuerde descargar los datos de trabajo **`Boston Crimes`** disponibles en el portal oficial del curso en BrightSpace. Descargue el **Zip,** descomprímalo y guarde los archivos CSV en la carpeta **`*/Data/`** de su copia local de código.

Diríjase al archivo **`view.py`** y ejecútelo, y seleccione secuencialmente la **opción 1** y **2** para iniciar el analizador y cargar información respectivamente.

```
*****************************************
Bienvenido
1- Inicializar Analizador
2- Cargar información de crimenes
3- Consultar crimenes en un rango de fechas
4- Consultar crimenes por codigo y fecha
0- Salir
*****************************************
Seleccione una opción para continuar
>2

Cargando información de crimenes ....
Crimenes cargados: 319073
Altura del arbol: 29
Elementos en el arbol: 1177
Menor Llave: 2015-06-15
Mayor Llave: 2018-09-03
```

Al cargar la información, verá el número de registros cargados, la altura del árbol (BST), el número de nodos en el árbol (BST), la menor llave encontrada, y la mayor llave encontrada. El árbol guarda un índice con parejas llave-valor, donde la llave es la fecha del crimen (YYYY-MM-DD) y el valor, los crímenes ocurridos ese día.

Tome nota del total de elementos en el árbol y la altura del BST reportada.

PASO 4: Modificación del Código en VSCode

Abra el archivo **model.py**, específicamente donde se crea el **dateIndex** usando un BST:

```
def newAnalyzer():
    """ Inicializa el analizador

    Crea una lista vacia para guardar todos los crimenes
    Se crean indices (Maps) por los siguientes criterios:
    -Fechas

    Retorna el analizador inicializado.
    """
    analyzer = {'crimes': None,
                'dateIndex': None
                }

    analyzer['crimes'] = lt.newList('SINGLE_LINKED', compareIds)
    analyzer['dateIndex'] = om.newMap(omaptype='BST',
                                      comparefunction=compareDates)
    return analyzer
```

Realice la modificación del parámetro **omaptype** para utilizar un árbol rojo-negro (RBT), como se muestra a continuación:

```
analyzer['dateIndex'] = om.newMap(omaptype='RBT',
                                  comparefunction=compareDates)
```

Ahora, diríjase al archivo **view.py** y ejecútelo, y seleccione secuencialmente la **opción 1** y **2** para iniciar el analizador y cargar información respectivamente.

```
*****************************************
Bienvenido
1- Inicializar Analizador
2- Cargar información de crimenes
3- Consultar crimenes en un rango de fechas
4- Consultar crimenes por codigo y fecha
0- Salir
*****************************************
Seleccione una opción para continuar
>2

Cargando información de crimenes ....
Crimenes cargados: 319073
Altura del arbol: 13
Elementos en el arbol: 1177
Menor Llave: 2015-06-15
Mayor Llave: 2018-09-03
```

Al cargar la información, verá el número de registros cargados, la altura del árbol (RBT), el número de nodos en el árbol (RBT), la menor llave encontrada, y la mayor llave encontrada. El árbol guarda un índice con parejas llave-valor, donde la llave es la fecha del crimen (YYYY-MM-DD) y el valor, los crímenes ocurridos ese día.

A continuación, responda las siguientes preguntas y registre su respuesta en el documento de observaciones:

a. ¿Qué diferencia existe entre las alturas de los dos árboles (BST y RBT)?
b. ¿Por qué pasa esto?

PASO 5: Actualizar el repositorio

Para el repositorio del laboratorio confirme los cambios con los comandos **`Commit`** y **`Push`** en la rama **main** en GitHub con el comentario *"Laboratorio 8 - Entrega final"* antes de la fecha límite de entrega.

PASO 6: Copiar la plantilla del Reto No. 3 en su organización

Copie/Haga **Fork** del repositorio del laboratorio en su organización con el procedimiento aprendido en las prácticas anteriores.

El repositorio del proyecto base que utiliza este laboratorio es el siguiente:

- https://github.com/ISIS1225DEVS/Reto3-Template

Antes de clonar el repositorio en su computador diríjase a su organización (Ej.: EDA2021-1-SEC02-G01 para el primer grupo de la sección 2 del curso) y cambie el nombre del repositorio de acuerdo con el esquema Reto3-G<<XX>> donde XX es el número del grupo de trabajo. (Ej.: Reto3-G01 para el grupo 1 de la sección 2).

Recuerde que **NO** necesita agregar la sección o el semestre en este nombre porque ya está identificado en su organización.

PASO 7: Descargue el Reto No. 3

Después de renombrar el proyecto dentro de su organización clone el código en su máquina local siguiendo lo aprendido y modifique el **README** principal con los nombres de los integrantes del grupo e identificar claramente cual miembro Implementará cual requerimiento individual, por ejemplo:

- *Req. 2 - Santiago Arteaga, 200411086, sa-artea@uniandes.edu.co*
- *Req. 3 - Carlos Lozano, 200211089, calozanog@uniandes.edu.co*

Paso 8: Crear el menú para el Reto

Tomando inspiración del código estudiado en el ejemplo, implemente en el **`view.py`** del reto el menú de opciones para la carga de archivos y los cinco requerimientos correspondientes.

PASO 9: Cargar datos del Reto

Descargue los datos oficiales del reto de la sección unificada de la clase de la carpeta de contenido **RETOS/Reto 3/Datos** el grupo de archivos ZIP de la página contienen los CSV necesarios para el desarrollo del reto.

Recuerde seleccionar el set más apropiado para las capacidades de sus máquinas. Los conjuntos de trabajo disponibles son:

- **Small:** porción pequeña de los datos.
- **5pct:** 5.0% de los datos originales.
- **10pct:** 10.0% de los datos originales.
- **20pct:** 20.0% de los datos originales.
- **30pct**: 30.0% de los datos originales.
- **50pct:** 50.0% de los datos originales.
- **80pct:** 80.0% de los datos originales.
- **Large:** 100.0% de los datos originales.

Para acelerar el desarrollo y las pruebas del código recomendamos utilizar la versión **`-small`** en la implementación.

Ahora modifique el código en el **`view.py`** para que la consola tenga las opciones para crear y cargar el catalo, pero además las opciones para cumplir con los requerimientos del reto 3. Por ahora **solo** las opciones para crear y cargar el catálogo serán funcionales.

Implemente código en el **`model.py`** y en **`controller.py`** para crear el catálogo y cargar los datos dentro de las estructuras de datos estudiadas en el curso y en particular **TAD orderedmap**.

PASO 10: Avanzar en el Requerimiento 1

Para avanzar en la implementación del requerimiento 1 cree un árbol RBT como índice para solucionar el requerimiento. Al finalizar esta práctica los datos característicos (altura y número de elementos) deberán desplegarse adecuadamente al ejecutar la **opción 3**.

PASO 11: Actualizar el repositorio en la rama principal

Confirme los cambios con los comandos **`Commit`** y **`Push`** en la rama **`main`** local y de GitHub con el comentario "Primera entrega – Reto 3" antes de la fecha límite de entrega.

PASO 12: Revisar entregables de la practica

Finalmente, para realizar la entrega del laboratorio revise que sus entregables de la practica estén completos. Para ello, siga las siguientes indicaciones:

1) Acceso al profesor de laboratorio y los monitores de su sección a la organización del grupo.
2) **README** del repositorio con los datos completos de los integrantes del grupo (nombre completo, correo Uniandes y código de estudiante).
3) Enlace al repositorio GitHub **LabBalancedTrees -S<<XX>>-G<<YY>>** con rama **`main`** actualizada con el comentario *"Laboratorio 8 – Entrega final"* antes del límite de entrega.
4) Incluir en repositorio del laboratorio en la carpeta **Docs** el documento **observaciones-lab8.pdf** con las respuestas a las preguntas de observación.
    a) ¿Qué diferencia existe entre las alturas de los dos árboles (BST y RBT)?
    b) ¿Por qué pasa esto?
5) Enlace al repositorio GitHub Reto3-G<<XX>> con rama **`main`** actualizada con el comentario "Avance Req 1– Reto 3" antes de la fecha límite de entrega.

PASO 13: Compartir resultados con los evaluadores

Envíe el **enlace (URL)** del repositorio por **BrightSpace** antes de la fecha límite de entrega.

Recuerden que cualquier documento solicitado durante la práctica debe incluirse dentro del repositorio GIT y solo se calificarán los entregables hasta el último ***COMMIT*** realizado previo a la media noche **(11:59 PM) del 27 de octubre de 2021.**