PUCE Virtual

DOCUMENTO DE PROFUNDIZACIÓN

Programación III

Laboratorio de Contenido 1: Árbol Binario de Búsqueda (BST) – Indexación de Eventos

Laboratorio de Contenido 1: Árbol Binario de Búsqueda (BST) – Indexación de Eventos

Explicación básica

Un Árbol Binario de Búsqueda (BST) es una estructura jerárquica donde los valores menores se ubican a la izquierda y los mayores a la derecha. Esto permite búsquedas eficientes en O(log n) en promedio.

Objetivos

- Comprender la propiedad de orden del BST.
- Implementar inserción y búsqueda.
- Aplicarlo a indexación de eventos de seguridad.

Caso práctico

Un SOC necesita almacenar eventos según su timestamp para consultarlos rápidamente.

Problema

Insertar los timestamps:
[50, 30, 70, 20, 40, 60, 80]
Buscar el evento 60.

Aplicación paso a paso

1. Insertar 50 → raíz.
2. Insertar 30 → izquierda.
3. Insertar 70 → derecha.
4. Insertar 20 → izquierda de 30.
5. Insertar 40 → derecha de 30.
6. Insertar 60 → izquierda de 70.
7. Insertar 80 → derecha de 70.

Buscar 60 → comparar 50 → derecha → 70 → izquierda → encontrado.

Implementación en Python

```
class Node:
  def __init__(self, valor):
    self.valor = valor
    self.izquierda = None
    self.derecha = None

class BST:
```

```
    def __init__(self):
        self.raiz = None

    def insertar(self, raiz, valor):
        if raiz is None:
            return Node(valor)
        if valor < raiz.valor:
            raiz.izquierda = self.insertar(raiz.izquierda, valor)
        else:
            raiz.derecha = self.insertar(raiz.derecha, valor)
        return raiz

    def buscar(self, raiz, valor):
        if raiz is None or raiz.valor == valor:
            return raiz
        if valor < raiz.valor:
            return self.buscar(raiz.izquierda, valor)
        return self.buscar(raiz.derecha, valor)

bst = BST()
valores = [50, 30, 70, 20, 40, 60, 80]
for v in valores:
    bst.raiz = bst.insertar(bst.raiz, v)

resultado = bst.buscar(bst.raiz, 60)
print("Encontrado:", result.valor if resultado else "No encontrado")
```

Conclusiones

El BST permite búsquedas rápidas y organización automática de datos. Sin embargo, si se desbalancea, pierde eficiencia.