Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

**“Búsqueda de apellidos entre un rango de identificadores (RUT)”**

Taller de Estructuras de Datos y Algoritmos INFO088

Prof. Héctor Ferrada

Ingeniería Civil en Informática

**Joaquin Flores**

**Maximiliano Medina**

**Felipe Lizama**

**Introducción**

Los sistemas actuales constan de muchas estructuras de datos para almacenar información de distintas formas, esto nos da la capacidad de solucionar un problema de varias maneras (unas más eficientes que otras), eso es lo que vamos a demostrar a continuación.

En este informe se realizará un seguimiento y análisis del código para la tarea dada, cuyo problema a abordar es la búsqueda de apellidos entre un rango de identificadores. Se realizarán 3 soluciones con distintas estructuras de datos cada una. Se harán hipótesis iniciales de estas soluciones y finalmente se implementarán para experimentar, y de esta forma confirmar o negar nuestras hipótesis iniciales.

Durante el proceso se hará una explicación y justificación de la lógica usada, además de detallar la experimentación junto con gráficas de esta.

**Resumen**

Para la Búsqueda de apellidos entre un rango de identificadores (RUT) Se utilizarán 3 distintos métodos con distintas estructuras de datos cada una, las cuales son “Búsqueda en una lista enlazada”, “Árbol binario de búsqueda” e “Implementación y búsqueda en un AVL”.

Luego de su implementación y experimentación se verificará cual es el más eficiente y porqué.

**Hipótesis iniciales:**

* Se espera que la solución 3 “implementación y búsqueda en un AVL” sea el método más eficiente de todos.
* Caso contrario, la solución 1 “Búsqueda en una lista enlazada” debería ser el método más ineficiente.
* El tiempo de respuesta debería ser 0,1s aproximadamente para la solución 1, para las demás correspondería un tiempo inferior.

**Metodología**

1. Búsqueda en una lista enlazada

|  |
| --- |
| **Inicio**  p <- perOrdenRut  Para i<-0 hasta personas.size()  p <- nuevo nodo  p->pos <- i  p <- p->next  Fin para  Para i<-0 Hasta i<-personas.size()-1 Hacer  Para j<-0 Hasta j<-personas.size()-i-1 Hacer  Si personas[p->pos].rut > personas[p->next->pos].rut Entonces  aux <- p->pos  p->pos <- p->next->pos  p->next->pos <- aux  Fin Si  Fin Para  Fin Para  **Fin** |