Natalia Mariel Calderón Echeverría
202200007
Estructuras de Datos
Primer semestre 2024

MANUAL TÉCNICO

Requerimientos básicos:

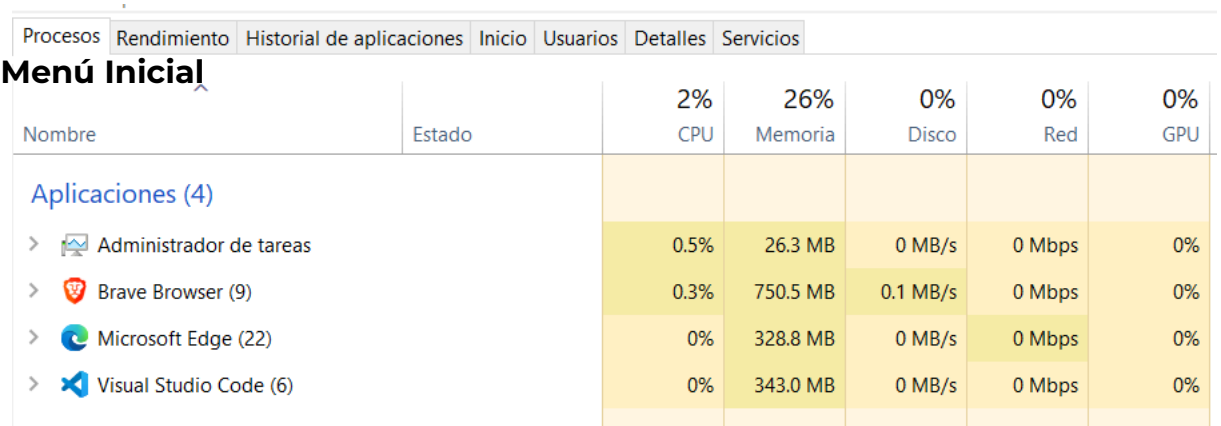
- **Lenguaje programado:** FORTRAN
- **Compilador: GFORTRAN**
- **Librerías:**
    - **Json module**

      Json module forma parte fundamental del siguiente proyecto ya que gracias a esta librería es posible leer los archivos json, que son los que contienen la información necesaria para llenar las estructuras de datos disponibles.
    - **Graphviz**

      Graphviz hace posible que crear las representación gráfica de dichas estructuras, gracias a esta librería es posible generar los árboles, las matrices y más importante las mismas imágenes.

Requerimientos del equipo:

-Menú Inicial



```
-----------------------------------
FASE 2 EDD - 202200007
-----------------------------------
MENU PRINCIPAL
1. INICIO DE SESION
2. REGISTRAR USUARIO
3. Salir
-----------------------------------
Ingrese una opcion:
```

***INICIAR SESIÓN**

- **MODO ADMINISTRADOR**

  De este modo entran únicamente aquellos que introducen las credenciales propias del administrador. El modo administrado es capaz de realizar una carga masiva de clientes a comparación del simple sistema de registro del menú principal.

  La credenciales propias del administrado son:

  - ***admin***
  - ***EDD2024***

```
-----------------------------------
PROGRAM MENU - ADMINISTRADOR
1. CARGA MASIIVA DE CLIENTES
2. ARBOL B - CLIENTES
3. REPORTES
4. INSERTAR CLIENTE
5. MODIFICAR CLIENTE
6. ELIMINAR CLIENTE
7. salir del modo administrador
-----------------------------------
Ingrese una opcion:
```

El menú del modo administrado le brinda al administrador la capacidad de:

1. realizar la carga masiva de usuario (a través de una archivo json)

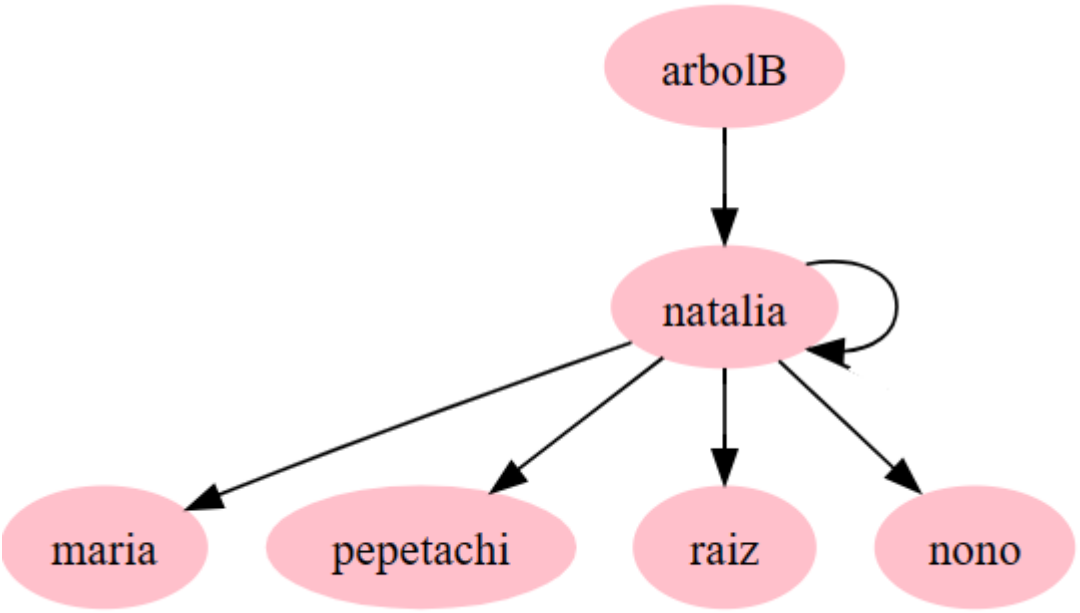
```
[
    {
        "dpi":"075551011",
        "nombre_cliente":"natalia",
        "password":"nataliacontra"
    },
    {
        "dpi":"075545765",
        "nombre_cliente":"maria",
        "password":"mariacontra"
    },
    {
        "dpi":"08005964",
        "nombre_cliente":"pepe",
        "password":"pepecontra"
    },
    {
        "dpi":"09005964",
        "nombre_cliente":"tachi",
        "password":"tachicontra"
    },
    {
        "dpi":"08005984",
        "nombre_cliente":"nono",
        "password":"nonocontra"
    },
```

El archivo JSON para clientes esta construido con la siguiente estructura:
Cada objeto dentro del arreglo representa un cliente y tiene tres atributos: "dpi" que representa el Documento Personal de Identificación del cliente, "nombre_cliente" que indica el nombre del cliente y "password" que contiene la contraseña asociada a ese cliente. Estos datos probablemente se utilizarán para construir una estructura de datos eficiente para la gestión de clientes en un sistema.

La subrutina json_cliente, implementada en Fortran, se encarga de cargar un archivo JSON que contiene información detallada sobre clientes, incluyendo su DPI, nombre y contraseña.
Utilizando las capacidades de procesamiento de JSON del módulo, la subrutina extrae estos datos y los asigna a objetos del tipo user_type, representando así a cada cliente individualmente. Posteriormente, estos objetos se insertan en una estructura de datos tipo árbol B mediante la subrutina insert, permitiendo una gestión eficiente de los clientes en el sistema.

```
subroutine json_cliente(filename)
    use json_module
    use btree
    use user
    implicit none

    character(len=*), intent(in) :: filename
    type(json_file) :: json
    type(json_value), pointer :: listPointer, itemPointer, dpiPointer, nombrePointer, passwo
    type(json_core) :: jsonc
    type(user_type) :: user

    integer :: i, size, dpi_int
    logical :: found
    character(:), allocatable :: dpi, nombre, password

    call json%initialize()
    call json%load(filename=filename)

    call json%get_core(jsonc)
    call json%get('', listPointer, found)
    !print *, 'Root found:', found

    if (found) then
        call json%info('', n_children=size)
        !print *, 'Number of children:', size
        do i = 1, size

            call jsonc%get_child(listPointer, i, itemPointer, found)
            print *, '-------------------'
            print *, '-------------------'
            if (found) then
                user = user_type(0,"","")
                call jsonc%get_child(itemPointer, 'dpi', dpiPointer, found)
                if (found) then
                    call jsonc%get(dpiPointer, dpi)
                    read(dpi, *) dpi_int
                    print *, 'dpi:', trim(dpi), ', dpi_int:', dpi_int
                    user%dpi = dpi_int
                end if

                call jsonc%get_child(itemPointer, 'nombre_cliente', nombrePointer, found)
                if (found) then
                    call jsonc%get(nombrePointer, nombre)
                    print *, 'nombre_cliente:', trim(nombre)
                    user%name = trim(nombre)
                end if

                call jsonc%get_child(itemPointer, 'password', passwordPointer, found)
                if (found) then
                    call jsonc%get(passwordPointer, password)
                    print *, 'password:', trim(password)
                    user%password = trim(password)
                end if
                call insert(user)
                print *, '--------lista-------'
                call lista_clientes(root)
                print *, '---------------'
```

2. visualizar el árbol b, propio de la estructura



3. reportes de usuarios
   Realiza un recorrido por niveles de los usuarios dentro del arbolB, también permite buscar información sobre un usuario en particular.

```
3
 REPORTES
 -----recorrido niveles------
natalia,maria,pepe,tachi,raiz,nono, ----------------------------
 ----------------------------
 ingrese dpi a buscar--------
9005964
 nombre: tachi
 dpi:      9005964
 password: tachicontra
 ----------------------------
```

```
 ----------------------------
 Ingrese una opcion:
4
 INSERTAR CLIENTE
 ingrese dpi:
9007000
 ingrese nombre:
prueba
 ingrese contrasena:
pruebacontra
 ******************************
 ******usuario ingresado********
 nombre: prueba
 dpi:      9007000
 password: pruebacontra
 ******************************
```

4. Ingresar usuarios

Esta opción permite registrar nuevos usuarios, se encarga de solicitar las siguientes credenciales:
-dpi
-nombre
-password

- **MODO USUARIO**
  El modo usuario es al cual ingresa cualquier usuario registrado que no sea el administrador.

```
 ----------------------INICIO DE SESION----------------------
 INGRESE DPI -en caso de administrador las credenciales respectivas
9005964
 Ingrese la contrase▒a:
tachicontra
 buscando en el sistema
     9005964
 ----------------------------------
 PROGRAM MENU - CLIENTE
 1. CARGA MASIIVA DE CAPAS
 2. CARGA MASIVA DE IMAGENES
 3. CARGA MASIVA DE ALBUMES
 4. VISUALIZAR ESTRUCTURAS
 5. GENERAR IMAGENES
 6. REPORTES
 7. salir del MODO CLIENTE
 ----------------------------------
```

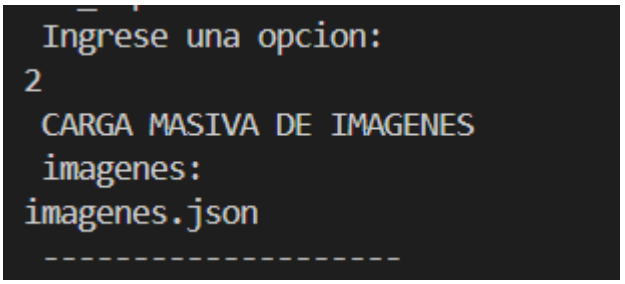
  Este permite que el usuario genere, manipule y administre sus: capas, imágenes y álbumes. Entre sus funcionalidades está:

  1. Carga masiva de capas
     En este apartado se introduce el archivo con extensión .json que desea que sea el que cargue la información.

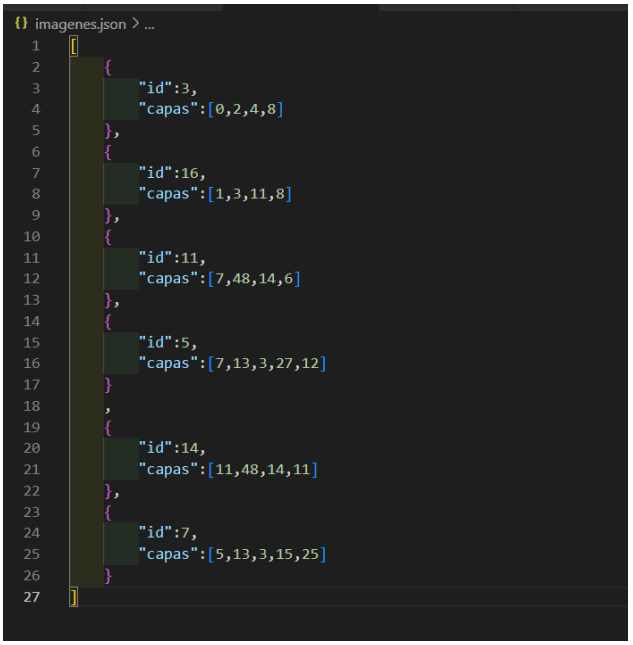
```
 Ingrese una opcion:
1
  CARGA MASIIVA DE CAPAS
 capas:
capas.json
```

Este archivo JSON describe una estructura de capas para la representación de una imagen. Cada elemento de la lista representa una capa, identificada de manera única por su id_capa, junto con una lista de píxeles asociados a esa capa. Cada píxel se caracteriza por su posición en fila y columna, así como por su color expresado en formato hexadecimal.

2. Carga masiva de imágenes.
   En este apartado se introduce el archivo con extensión .json que desea que sea el que cargue la información sobre las imágenes.



   El archivo JSON propio de las imágenes proporciona una estructura de datos que organiza diferentes conjuntos de capas para la representación de imágenes. Cada elemento en la lista corresponde a un conjunto identificado por un id único. Dentro de cada conjunto, se especifican las capas que lo componen mediante una lista de identificadores de capa.



3. Carga masiva de álbumes
   En este apartado se introduce el archivo con extensión .json que desea que sea el que cargue la información propia de los álbumes

```
3
 CARGA MASIVA DE ALBUMES
 album:
album.json
 --------------------
```

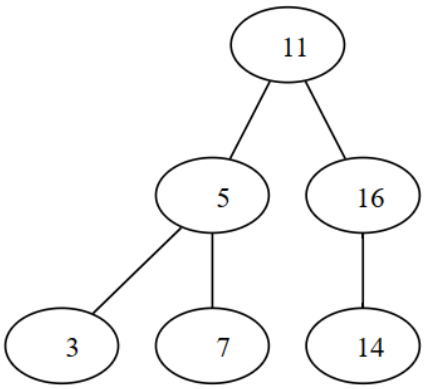
El archivo json de albumes esta compuesto por colecciones de imágenes organizadas por álbumes. Cada elemento de la lista representa un álbum identificado por su nombre_album y acompañado de una lista de índices que corresponden a las imágenes dentro del álbum.

album.json

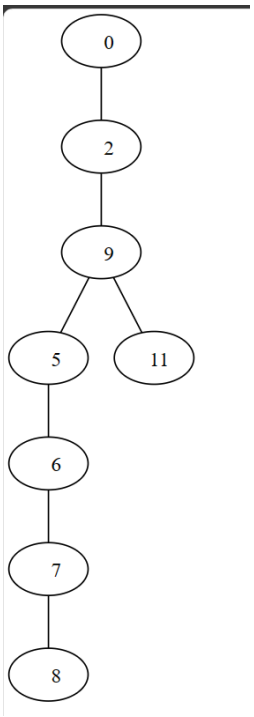
```json
[
    {
        "nombre_album":"album1",
        "imgs":[0,2,4,8]
    },
    {
        "nombre_album":"album4",
        "imgs":[1,3,11,8]
    },
    {
        "nombre_album":"album5",
        "imgs":[7,2,9,10,13,5]
    }
]
```

4. Visualización de las estructuras

- **ARBOL AVL - IMÁGENES**

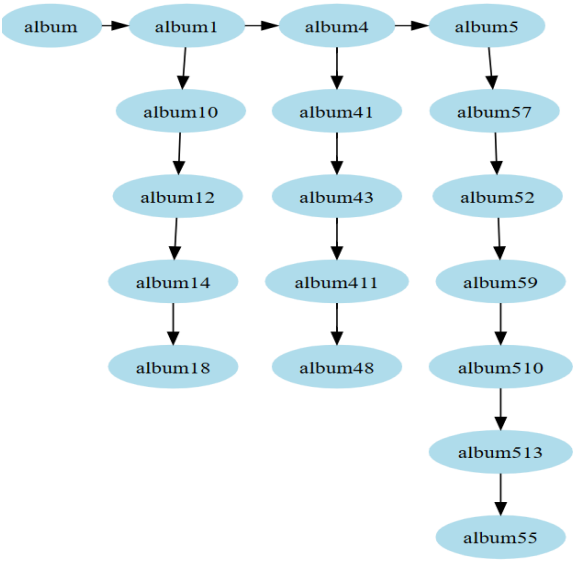


- **ARBOL BINARIO**

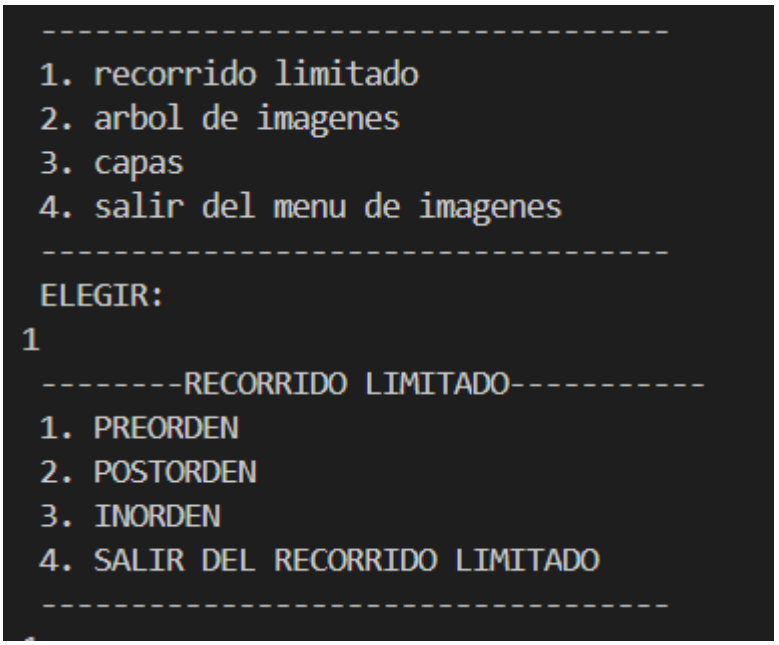


- **MATRIZ CAPAS**

  – **Lista album**



5. Generacion de imagenes



   - por recorrido limitado

     En este apartado se le indica al programa que tipo de recorrido desea realizar: preorden, postorden e inorden. Posteriormente se indica la cantidad de capas que desea graficar, lo que generará una imágenes a medida que recorre el árbol según el recorrido indicado y las apila. Nótese que las capas, una sobre otra de acuerdo al recorrido.

Explicación:

Dicho recorrido se realiza a través de apilar capas, es decir se insertan una sobre otra. Para lograr esta funcionalidad se hizo uso de una cola y de la lógica propia del recorrido específico. Primero se realiza el recorrido y se llena la cola propia de este recorrido, debido a la funcionalidad de la cola - FIFO (First In, First Out) - se van sacando cada uno de los elementos y al tiempo que se van sacando se agrega a la construcción de una capa, esto se hace dependiendo de la cantidad de capas que el usuario haya especificado.

```
---> recorrido limitado
--> PREORDEN
-> ingrese cantidad de capas
3
preorden -->
 0 ----------------------------------
 2 ----------------------------------
 9 ----------------------------------
 5 ----------------------------------
 6 ----------------------------------
 7 ----------------------------------
 8 ----------------------------------
11 ----------------------------------
-------------------------------------------
```

IMAGEN

```
2
POSTORDEN
---> recorrido limitado
--> POSTORDEN
-> ingrese cantidad de capas
3
postorden -->
 8  7  6  5 11  9  2  0 ---------------
```

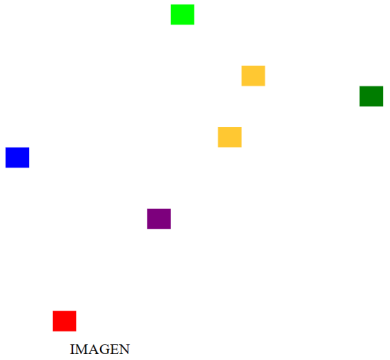
IMAGEN

```
3
INORDEN
---> recorrido limitado
--> INORDEN
-> ingrese cantidad de capas
3
inorden -->
 0  2  5  6  7  8  9 11 -----------
```

IMAGEN

- por capas
  En este apartado se indica la cantidad de capas que se desea graficar y posteriormente se van ingresando segun su id.

```
 ELEGIR:
3
 3. capas
 numero de capas que desea graficar
3
          3
 -> ingrese numero de vcapa
11
 BUSCANDO CAPA ID:          0
 BUSCANDO CAPA ID:          2
 BUSCANDO CAPA ID:          9
 BUSCANDO CAPA ID:         11
 -------generando imagen-------
 Graphviz file generated: imagen.dot.png
 <<<<<
          2
 -> ingrese numero de vcapa
2
 BUSCANDO CAPA ID:          0
 BUSCANDO CAPA ID:          2
 -----actualizando imagen----
 Graphviz file generated: imagen.dot.png
 capa agregada --> exitosamente
          1
 -> ingrese numero de vcapa
9
 BUSCANDO CAPA ID:          0
 BUSCANDO CAPA ID:          2
 BUSCANDO CAPA ID:          9
 -----actualizando imagen----
 Graphviz file generated: imagen.dot.png
 capa agregada --> exitosamente
```



IMAGEN

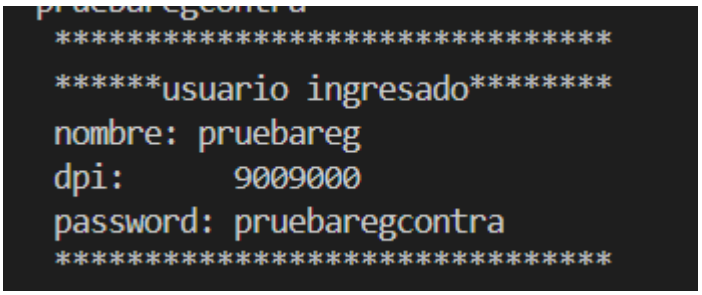
Al igual que en la construcción de imágenes por recorrido limitado, en esta función también se van apilando imágenes una sobre otra. En esta función no fue necesario el uso de una estructura auxiliar, debido a la naturaleza de la creación de imágenes (se indica que capas se desean graficar)

6. REPORTES
   Este apartado presenta los reportes relacionados con los recorridos y las imágenes propias de los clientes.
7. SALIR
   Esta opción le permite al usuario cerrar sesion y regresar al menu principal

***REGISTRAR USUARIO**

Esta opción permite registrar nuevos usuarios, se encarga de solicitar las siguientes credenciales:

- dpi
- nombre
- password


****
******usuario ingresado********
nombre: pruebareg
dpi: 9009000
password: pruebaregcontra
****

***SALIR**