

# INFO198 - SISTEMAS OPERATIVOS

## Prueba 3 - parte práctica

2023-02 (Dr. Luis Veas C.)

La prueba 2 consta de 2 partes, una teórica y una práctica.

- La parte teórica se realizará el día jueves 30/11/2023 en horario de clases
- La parte práctica se entrega a los alumnos el día viernes 17/11/2023, y se debe entregar al profesor a más tardar:
  - Jue 30/11/2023 a las 23:59 horas, para optar al 7
  - Domingo 03/12/2023 a las 23:59 horas, para optar al 6
- Esta sección debe realizarse en grupos de dos personas.
- La forma de entrega es vía email, bajo el siguiente formato:

asunto: INFO198 - SISTEMAS OPERATIVOS - Prueba 3 - práctica

cuerpo: nombre de los integrantes

adjunto: archivo zip comprimido con el siguiente formato

⇒ si son dos integrantes nombre-apellido1\_nombre-apellido2.zip

La ponderación de las pruebas es la siguiente:

- parte teórica 58% del total de la calificación, **para revisar la parte práctica debe tener por lo menos una calificación 2.5 de 7 en la parte teórica**
- parte práctica 42% del total de la calificación

## Problemas a resolver

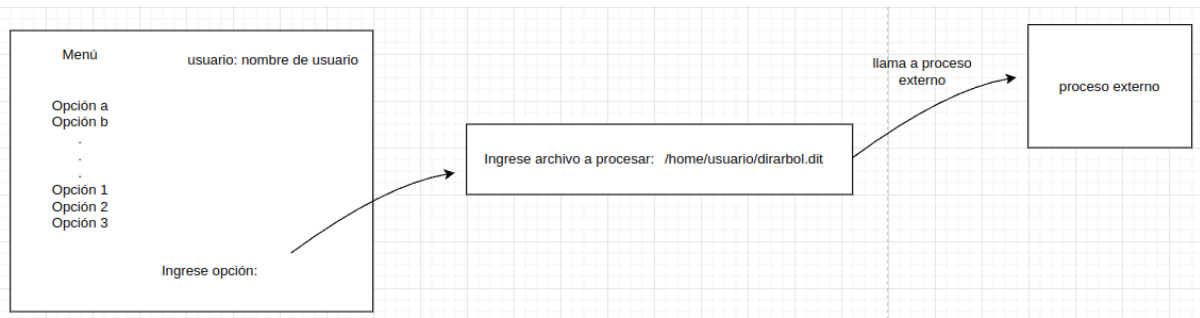
- 1) Trabajar sobre el sistema de menú y agregar tres opciones nuevas, las cuales deben llamar a tres procesos externos (en c++):

Opción 1 ⇒ Crear sistema de directorio basado en árbol

Opción 2 ⇒ Crear sistema de directorio basado en lista circular

Opción 3 ⇒ Procesamiento gráfico

- 2) seleccionar la opción “Crear sistema de directorio basado en árbol”, el sistema debe abrir una interfaz que permite ingresar el path de un archivo \*.dit



3) Debe validar la existencia del archivo y extensión

4) Debe validar el formato del archivo, el cual es el siguiente

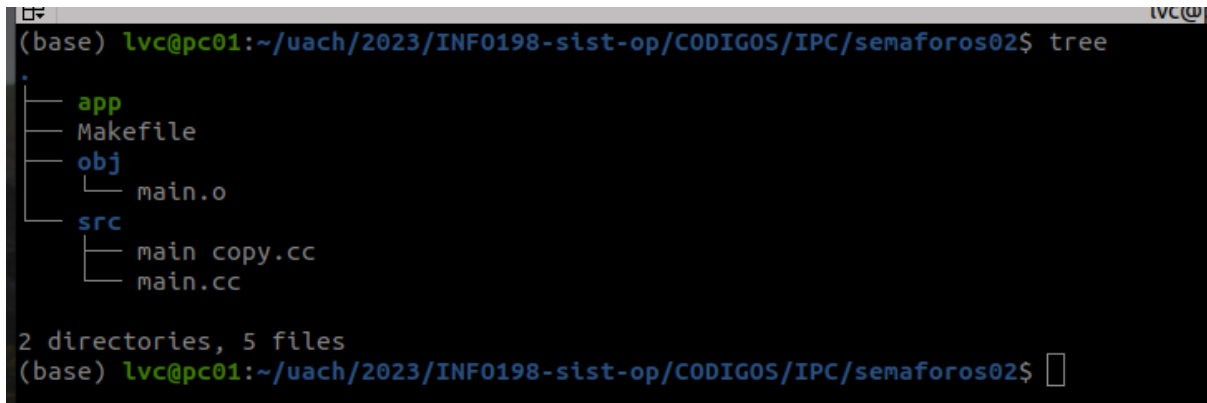
```

{
  "dirBase": "/home/usuario/objetos/",
  "objetos": [
    {
      "tipo": "directorio",
      "nombre": "dir1",
      "objetos": [
        {
          "tipo": "archivo", "nombre": "file1", "contenido": "la la", "permisos": ["escritura", "lectura"]},
        {
          "tipo": "archivo", "nombre": "file2", "contenido": "casa", "permisos": ["lectura"]},
        {
          "tipo": "archivo", "nombre": "file3", "contenido": "nnnn ccc", "permisos": ["escritura"]},
        {
          "tipo": "directorio",
          "nombre": "dir1.1",
          "objetos": [
            {
              "tipo": "archivo", "nombre": "file4", "contenido": "4 ccc", "permisos": ["escritura"]}
          ]
        }
      ]
    },
    {
      "tipo": "directorio",
      "nombre": "dir2",
      "objetos": []
    }
  ]
}
  
```

Es un json con un arreglo de objetos (los objetos pueden ser ingresados de forma anidada), los cuales pueden ser:

- 5) El dirBase, debe crear un directorio base en el cual se crearán todos los objetos.
- 6) Si el tipo de objeto es directorio, implica que deben crear un directorio, debe establecer el nombre y una lista de objetos anidados, los cuales pueden ser archivos, directorios o puede estar vacío
- 7) Si el tipo de objeto es archivo, implica que deben crear un archivo, agregarle el contenido y establecer los permisos señalados

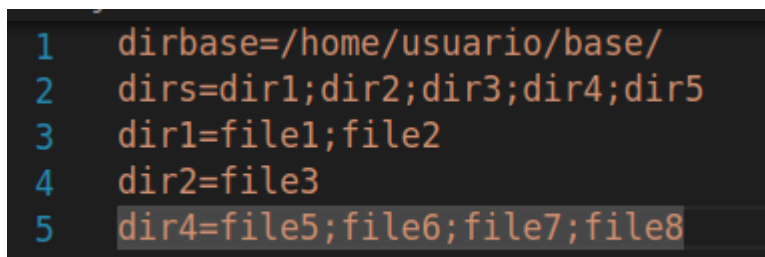
- 8) La creación de los objetos debe realizarse de forma iterativa e incremental, es decir, si encuentra objetos anidados debe crearlos y seguir el recorrido
- 9) Debe mostrar el resultado de la creación de objetos (puede instalar “tree” es un programa de consola de linux, con este programa puede mostrar el árbol de directorios y archivos anidados)



```
(base) lvc@pc01:~/uach/2023/INF0198-sist-op/CODIGOS/IPC/semaforos02$ tree
.
├── app
├── Makefile
├── obj
│   └── main.o
└── src
    ├── main.copy.cc
    └── main.cc

2 directories, 5 files
(base) lvc@pc01:~/uach/2023/INF0198-sist-op/CODIGOS/IPC/semaforos02$
```

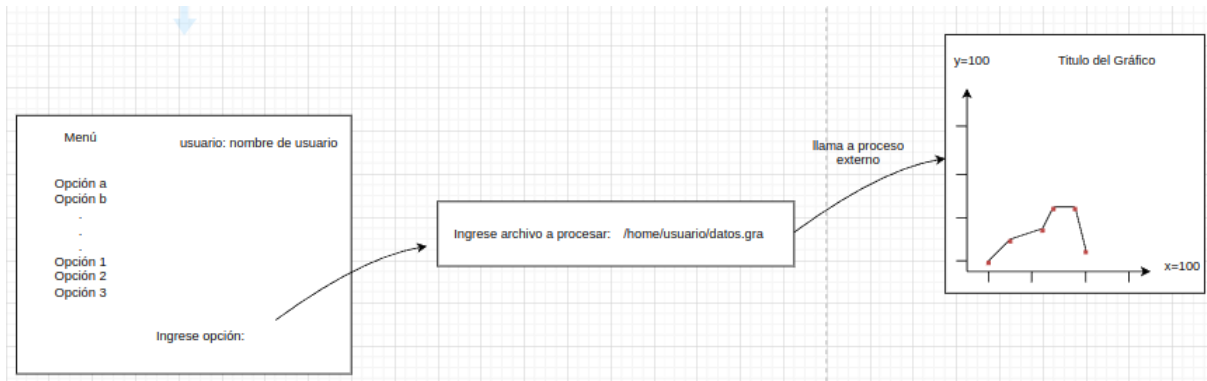
- 10) seleccionar la opción “Crear sistema de directorio basado en lista circular”, el sistema debe abrir una interfaz que permite ingresar el path de un archivo \*.dre
- 11) Debe validar la existencia del archivo y extensión
- 12) Debe validar el formato del archivo, el cual es el siguiente



```
1  dirbase=/home/usuario/base/
2  dirs=dir1;dir2;dir3;dir4;dir5
3  dir1=file1;file2
4  dir2=file3
5  dir4=file5;file6;file7;file8
```

- 13) dirbase, debe crear un directorio base en el cual se crearán todos los subdirectorios de dirs.
- 14) dirs, debe crear todos los directorios señalados (pueden ser N), cuando llegue al último directorio debe hacer un enlace virtual al primer directorio (revise la instrucción “ln -s”)
- 15) dirX, permite crear archivos de texto plano en el directorio señalado (pueden ser R archivos)
- 16) Debe mostrar el resultado de lo realizado (también puede usar “tree”)

- 17) Validaremos el enlace virtual, es decir, cuando entremos al último directorio, éste debe llevarnos al primer directorio
- 18) seleccionar la opción “Procesamiento gráfico”, el sistema debe abrir una interfaz que permite ingresar el path de un archivo \*.gra
- 19) Debe validar la existencia del archivo y extensión



- 20) Debe validar el formato del archivo, el cual es el siguiente

```
1  titulo:"gráfico de linea"
2  x:1,y:5
3  x:3,y:6
4  x:4,y:8
5  x:10,y:9
6  x:32,y:15
```

- 21) El sistema debe crear un gráfico computacional que contenga un plano cartesiano de 100x100
- 22) El gráfico debe tener “título”, este texto debe ser mostrado en la parte superior del gráfico. Este gráfico no debe ser una imagen, debe ser dibujado con píxeles en la pantalla, es decir, deben activar el modo gráfico de c++.
- 23) El sistema debe ser capaz de leer los puntos (x, y) del archivo y graficar una línea entre cada punto consecutivo