

Proyecto 2

Muñoz Tamés María Ángel

Tafolla Rosales Esteban

Para el desarrollo de este proyecto se han utilizado las bibliotecas:

- struct para realizar la conversión de números en formato little endian.
- os para obtener la información de los directorios del equipo del usuario.
- time y datetime para obtener las fechas en el formato deseado según las especificaciones dadas para el programa.
- math para realizar una división entera que devuelva un entero mayor o igual al resultado de la división (para la asignación de clusters).

Información de archivos

Para la extracción y escritura de datos dentro de la imagen se han realizado cinco funciones que permiten codificar y decodificar las cadenas a bytes y enteros a little endian y al contrario para ambos casos.

```
def sacaDatos(inicio, tamaño):  
    global imagen  
    imagen.seek(inicio)  
    return imagen.read(tamaño)  
def sacaDatosAscii(inicio, tamaño):  
    global imagen  
    imagen.seek(inicio)  
    return imagen.read(tamaño).decode("ascii")  
def datoUnpack(inicio, tamaño):  
    global imagen  
    imagen.seek(inicio)  
    dato = imagen.read(tamaño)  
    return struct.unpack('<i', dato)[0]  
def meteDatosAscii(inicio, dato):  
    global imagen  
    imagen.seek(inicio)  
    dato = dato.encode("ascii")  
    return imagen.write(dato)  
def meteDatoPack(inicio, dato):  
    global imagen  
    imagen.seek(inicio)  
    dato = struct.pack('<i', dato)  
    return imagen.write(dato)
```

Se extrae la información básica de nuestro sistema de archivos y es almacenado dentro de variables la cuales serán utilizadas

```
tamañoCluster= datoUnpack(40,4) #Da 1024  
clustersDirectorio = datoUnpack(45,4) #Da 4  
clustersUnidad = datoUnpack(50,4) #Da 720
```

Para almacenar la información de cada uno de los archivos se ha creado una clase que almacene la información básica de los archivos contenida en el directorio del sistema de archivos.

```
class archivo:
    global tamañoCluster
    def __init__(self, nombre, tamaño, clusterInicial, fechaCreacion, fechaModificacion):
        self.nombre = nombre.replace(" ", "")
        self.tamaño = tamaño
        self.clusterInicial = clusterInicial
        self.fechaCreacion = fechaCreacion
        self.fechaModificacion = fechaModificacion
        self.numClusters = math.ceil(tamaño/tamañoCluster)
```

Para obtener la información en el directorio se a creado una función la cual obtiene los datos de los archivos almacenados en el directorio y crea un objeto de tipo archivo o lo retorna.

```
def sacarDatosArchivo(posicion):
    inicial = 1024 + (posicion * 64)
    if(sacarDatosAscii(inicial+1,14) != "-----"):
        nombre = sacarDatosAscii(inicial+1, 14)
        tamaño = datoUnpack(inicial+16, 4)
        clusterInicial = datoUnpack(inicial+20,4)
        fechaCreacion = sacarDatosAscii(inicial+24, 14)
        fechaModificacion = sacarDatosAscii(inicial+38, 14)
        archivoAux = archivo(nombre, tamaño, clusterInicial, fechaCreacion, fechaModificacion)
        return archivoAux
```

De esta forma podemos realizar la lectura de la información del directorio y almacenarla en una lista de objetos de tipo archivo.

Mapecto de la imagen

Para conocer cuales clusters son los que se encuentran vacíos se a creado una lista con el tamaño de clusters que contiene el sistema de archivos en la cual se almacena el sistema de archivos el cual será modificado cada que se elimine o agregue un nuevo archivo. Esta lista contiene ceros y unos. Un uno representa que el cluster esta ocupado y un cero representa que un cluster está disponible. Para actualizar nuestro mapa se obtiene la información de nuestra lista de archivos.

```
def iniciaMapa():
    global mapaAlmacenamiento
    for x in range(5):
        mapaAlmacenamiento.append(1)
    while (len(mapaAlmacenamiento) != 720):
        mapaAlmacenamiento.append(0)

#Actualiza la informacion en el mapa a partir de la lista de archivos
def actualizaMapa():
    global mapaAlmacenamiento
    global archivos
    for x in range(720):
        mapaAlmacenamiento[x]=0
    for x in range(5):
        mapaAlmacenamiento[x]=1
    for archivoActual in archivos:
        aux = archivoActual.numClusters
        for j in range(aux):
            mapaAlmacenamiento[archivoActual.clusterInicial+j] = 1
```

Copiar archivos a la computadora

Se verifica la existencia del archivo en el sistema a partir de la información de la lista de archivos y en caso de existir crea una copia del archivo en la ruta especificada por el usuario

```
#Método que copia un archivo desde el sistema de archivos a la computadora
def copiaArchivoAComputadora(nombreArchivo, ruta):
    global archivos
    global imagen
    desfragmentar()
    #busca si el nombre del archivo se encuentra en nuestro sistema de archivos.
    posicion = buscaArchivoNombre(nombreArchivo)
    if(posicion != -1):
        auxiliar = sacaDatos(archivos[posicion].clusterInicial*1024, archivos[posicion].tamaño)
        if(os.path.exists(ruta) and not os.path.exists(ruta+"/"+nombreArchivo)):
            print("Archivo copiado exitosamente.")
            ArchivoNuevo = open(ruta+"/"+nombreArchivo, "bw")
            ArchivoNuevo.write(auxiliar)
            ArchivoNuevo.close()
        else:
            print("No se pudo copiar el archivo")
    else:
        print("No se pudo copiar el archivo")
```

Desfragmentar

Para desfragmentar la memoria se hace uso del mapa de clusters, este se recorre hasta encontrar un espacio que no esté siendo utilizado, después buscar el siguiente archivo y lo recorre hasta que no haya espacios entre los archivos almacenados en el sistema. Esto de forma sucesiva hasta que se termine de recorrer el mapa.

Borra archivo

Busca el archivo dentro de la lista de archivos y los elimina, actualiza el mapa de los cluster del sistema de archivos y por último modifica el directorio para borrar la información del archivo contenida en él.

```
#Método que borra un archivo de nuestro sistema de archivos
def borraArchivo(nombreArchivo, aux):
    global archivos
    for x in archivos:
        if(x.nombre==nombreArchivo):
            archivos.pop(archivos.index(x))
            actualizaMapa()
            for y in range(64):
                inicial = 1024 + (y * 64)
                nombre = sacaDatosAscii(inicial+1, 14).replace(" ", "")
                print(nombre)
                if(nombre == nombreArchivo):
                    borraEnDirectorio(y)
                    return
            elif(aux==2):
                return
    print("El archivo para borrar no existe")
    return
```

Copiar archivo a sistema de archivos

Desfragmenta los archivos almacenados en el sistema, verifica que cupla los requisitos básicos para que el sistema puede almacenar el nuevo archivo.

Tamaño del nombre, el archivo aun no esta almacenado, la ruta y el archivo que se van a copiar existen y hay espacio disponible para ser almacenado, en caso de ser así busca un espacio dentro del directorio para almacenar el archivo y lo almacena dentro del sistema.

```
#Método que copia un archivo desde nuestra computadora a nuestro sistema de archivos.
def copiaArchivoAlmacen(rutaOrigen):
    global archivos
    desfragmentar()
    #verifica que el archivo exista, que el tamaño del nombre sea de 14 o menos y que el tamaño del archivo sea de menos de 737,140 bytes (715 clusters)
    if(os.path.exists(rutaOrigen) and len(os.path.split(rutaOrigen)[1])<15 and os.stat(rutaOrigen).st_size < 737140):
        #verificamos que no exista un archivo en el sistema con el mismo nombre
        if(buscaArchivoNombre(os.path.split(rutaOrigen)[1].replace(" ", ""))!=-1):
            nombre=agregaEspacios(os.path.split(rutaOrigen)[1])
            tamaño = os.stat(rutaOrigen).st_size
            #Busca un espacio en el cual pueda almacenar el nuevo archivo
            clusterInicial = asignaCluster(tamaño)
            #En caso de no haber espacio para el archivo informa al usuario
            if (clusterInicial == -1):
                print("No hay espacio disponible para almacenar el archivo dentro del directorio")
                return
            fechaCreacion = datetime.datetime.strptime(time.ctime(os.stat(rutaOrigen).st_ctime), "%a %b %d %H:%M:%S %Y").strftime("%Y%m%d%H%M%S")
            fechaModificacion = datetime.datetime.strptime(time.ctime(os.stat(rutaOrigen).st_mtime), "%a %b %d %H:%M:%S %Y").strftime("%Y%m%d%H%M%S")
            archivoAux = archivo(nombre, tamaño, clusterInicial, fechaCreacion, fechaModificacion)
            #agrega el nuevo archivo a la lista
            archivos.append(archivoAux)
            #actualiza el directorio
            #verifica si hay espacio en el directorio
            if (agregaADirectorio(archivoAux)==-1):
                return
            agregaArchivoAlmacen(rutaOrigen, archivoAux)
            actualizaMapa()
        else:
            print("Ya existe un archivo con el mismo nombre. Porfavor intentele nuevamente.")
```

Pruebas

Al compilar el programa nos presentara un menú con 5 opciones

```
Opciones:
1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos
```

Primero listaremos el contenido dentro del sistema de archivos tal cual esta en la imagen dada para realizar las pruebas en el sistema

```
1
README.org          31078 bytes
logo.png            188055 bytes
mensajes.png        296326 bytes
```

Ahora realizaremos la copia de los tres archivos contenidos dentro del sistema de archivos a la carpeta “Nuevos archivos”

```
Opciones:
1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos

2
Ingresa el nombre del archivo que quieres copiar: README.org
README.org
Ingresa la ruta en la cual lo quieres copiar: C:\Users\steph\Downloads\Proyecto2\NuestroProyecto\Nuevos archivos
Archivo copiado exitosamente.




Opciones:
1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos

2
Ingresa el nombre del archivo que quieres copiar: logo.png
logo.png
Ingresa la ruta en la cual lo quieres copiar: C:\Users\steph\Downloads\Proyecto2\NuestroProyecto\Nuevos archivos
Archivo copiado exitosamente.

Opciones:
1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos

2
Ingresa el nombre del archivo que quieres copiar: mensajes.png
mensajes.png
Ingresa la ruta en la cual lo quieres copiar: C:\Users\steph\Downloads\Proyecto2\NuestroProyecto\Nuevos archivos
Archivo copiado exitosamente.

Opciones:
1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos
```

Descargas > Proyecto2 > NuestroProyecto > Nuevos archivos					Buscar el
Nombre	Fecha de modificación	Tipo	Tamaño		
 logo	06/01/2023 01:10 a. m.	Archivo PNG	184 KB		
 mensajes	06/01/2023 01:10 a. m.	Archivo PNG	290 KB		
 README.org	06/01/2023 01:10 a. m.	Archivo ORG	31 KB		

Ahora copiaremos un archivo de nuestro equipo al sistema. Es importante que la ruta la mandemos con el nombre del archivo

```
3
Ingresa la ruta del archivo:C:\Users\steph\Downloads\Proyecto2\NuestroProyecto\Examen.cc

Opciones:
1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos
```

Para verificar su correcto almacenamiento listaremos los archivos almacenados en el sistema

```
Opciones:
1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos

1
README.org          31078 bytes
logo.png            188055 bytes
mensajes.png        296326 bytes
Examen.cc           2348 bytes

Opciones:
1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos
```

Ahora eliminaremos el archivo almacenado anteriormente

Opciones:

1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos

4

Ingresa el nombre del archivo que quieres eliminar: Examen.cc

README.org

Examen.cc

Opciones:

1. Listar archivos
2. Copiar uno de los archivos a tu computadora
3. Copiar un archivo de tu computadora hacia FiUnamFS
4. Eliminar un archivo del FiUnamFS
5. Cerrar el sistema de archivos

1

README.org 31078 bytes

logo.png 188055 bytes

mensajes.png 296326 bytes