Imagen que contiene Diagrama

Descripción generada automáticamente

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

Principios de Sistemas Operativos

**Segundo Proyecto – Empacador de Archivos**

Realizado por:

Jennifer Alvarado Brenes c. 2020124171

Kenneth Palacios Molina c.

Profesor:

Ing. Armando Arce Orozco

I Semestre 2024

**Contenido**

Introducción…………………….…………………………………………………………

Descripción del problema……………………………………………………………….

Definición de estructuras de datos………………………………………………………

Descripción detallada y explicación de los componentes principales del programa….............................................................................................................

Mecanismo de acceso a archivos……………………………………………..

Estructura de directorios internos (FAT)………………………………………

Estrategia de manejo de bloques libres……………………………………..

Procedimiento de desfragmentación del archivo……………………………..

Análisis de resultados de pruebas……………………………………………………..

Conclusiones……………………………………………………………………………..

1. **Introducción**

El proyecto se centra en el desarrollo de un empacador de archivos denominado "star" (Simple Tar), inspirado en el comando tar utilizado en entornos UNIX. Este programa permite almacenar múltiples archivos en un solo archivo, facilitando su transporte y gestión. A través de este proyecto, se busca ofrecer una alternativa eficiente y versátil para la manipulación de archivos en sistemas compatibles con UNIX.

El empacador de archivos "star" se implementa en el lenguaje de programación C y se ejecuta en un entorno UNIX. El programa proporciona funcionalidades básicas similares a las del comando tar, como la creación, extracción, listado y eliminación de archivos dentro de un archivo tar. Además, se integran características adicionales para el manejo eficiente de la estructura del archivo tar y la gestión de bloques libres, garantizando un rendimiento óptimo y una utilización eficaz del espacio en disco.

1. **Descripción del problema**

El objetivo de este proyecto consiste en programar un empacador de archivos. Este es el tipo de funcionalidad que provee el comando tar en ambientes UNIX.

El programa tar, es usado para almacenar múltiples archivos en un solo archivo. Dentro de los entornos Unix tar aparece como un comando que puede ser ejecutada desde la línea de comandos de una consola de texto o desde un simple terminal. El formato del comando tar es, comúnmente

**tar <opciones> <archivoSalida> <archivo1> <archivo2> ... <archivoN>**

donde <archivoSalida> es el archivo resultado y <archivo1>, <archivo2>, etc; son los diferentes archivos que serán "empaquetados" en <archivoSalida>.

Las opciones más comunes son las siguientes:

* -c, --create : crea un nuevo archivo
* -x, --extract : extraer de un archivo
* -t, --list: listar los contenidos de un archivo
* --delete: borrar desde un archivo
* -u, --update: actualiza el contenido del archivo
* -v, --verbose: ver un reporte de las acciones a medida que se van realizando
* -f, --file: empacar contenidos de archivo, si no está presente asume la entrada estándar.
* -r, --append: agrega contenido a un archivo
* -p, --pack: desfragmenta el contenido del archivo (no presente en tar )

Ejemplos:

Si queremos empacar un archivo llamado "index.html" y guardar los datos en "html-paq.tar", lo haríamos con la instrucción: **tar -cvf html-paq.tar index.html**

Si queremos desempaquetar todo el contenido de un archivo llamado xxx.tar podemos utilizar un comando como este: **tar -xvf xxx.tar**

Para archivar el contenido de tres archivos doc1.txt, doc2.txt y data.dat: **tar -cvf foo.tar doc1.txt doc2.txt data.dat**

Si ahora se desea eliminar el contenido del archivo data.dat se ejecutaría: **tar --delete -vf foo.tar data.dat**

Para agregar ahora un nuevo archivo test.doc a foo.tar se ejecutaría: **tar -rvf foo.tar test.doc**.

Los contenidos se desempacarán en el directorio actual.

**Programación**

* Se deberá programar el comando star ("simple tar", NO "estrella"), de tal forma que acepte los comandos básicos mostrados anteriormente. Para desarrollar su programa usted debe tomar en cuenta los siguientes aspectos:
* Para organizar la información de los archivos almacenados se deberá utilizar una estructura tipo "FAT" al inicio del archivo.
* El archivo debe almacenarse utilizando bloques de datos de tamaño 256K. La estructura "FAT" deberá guardar las posiciones de los diferentes bloques que componen cada archivo.
* Al crear un archivo empacado este se crea del tamaño necesario para almacenar los archivos agregados. Cuando se borra algún contenido, el archivo empacado no cambia de tamaño sino que se lleva registro de los espacios liberados. Si posteriormente se agrega unn nuevo contenido entonces se reutiliza el espacio libre. Si aún así el nuevo contenido no cabe, se hace crecer el archivo empacado. No debe utilizar ningún archivo auxiliar para hacer crecer el archivo.
* Tome en cuenta que un archivo que se agrega puede ya existir en el archivo empacado. Es decir, lo que se desea hacer es actualizar su contenido. Para esto existe la opción update (-u) que sobrescribirá el contenido de un archivo.
* Se debe llevar un control de los bloques libres en la estructura FAT, como si fuera un archivo normal.
* Este programa no utilizará los derechos de acceso, que normalmente almacenaría un archivo empacado tar en ambiente UNIX.
* La opción de desfragmentación (-p) no es estándar (no está presente en tar) y lo que hace es desfragmentar el contenido almacenado en el archivo empacado y liberar cualquier espacio sin utilizar. Es decir, con este comando se liberarán todos los bloques libres y el tamaño del archivo empacado se ajustará al contenido real existente. Note que no se debe utilizar un archivo temporal para realizar esta función, toda la desfragmentación se debe realizar sobre el contenido del mismo archivo.
* La opción -v muestra información sobre la operación ejecutada. Se puede aplicar dos veces -vv para ver información adicional.

**tar -rvf foo.tar test.doc**

1. **Definición de estructuras de datos**

El programa utiliza una estructura de datos tipo "FAT" (File Allocation Table) para organizar la información de los archivos almacenados dentro del archivo tar. Esta estructura se representa mediante la definición de una estructura llamada **FileHeader**, que contiene los siguientes campos:

* **name**: Nombre del archivo.
* **size**: Tamaño del archivo en bytes.
* **jump**: Desplazamiento relativo del contenido del archivo dentro del archivo tar.
* **deleted**: Bandera booleana que indica si el archivo ha sido marcado como eliminado.

La estructura **FileHeader** se utiliza para cada archivo almacenado en el archivo tar, permitiendo un acceso eficiente y una gestión adecuada de los archivos. Además, se define una constante **HEADER\_SIZE** que representa el tamaño en bytes de la estructura **FileHeader**, facilitando su manipulación dentro del programa.

1. **Descripción detallada y explicación de los componentes principales del programa**
   1. **Mecanismo de acceso a archivos**

El programa implementa un mecanismo de acceso a archivos para crear, extraer, listar y eliminar archivos dentro de un archivo tar. Utiliza la biblioteca estándar de C para manejar archivos.

**Crear un archivo tar:**

La función **create\_tar** toma un nombre de archivo tar de salida y una lista de archivos de entrada. Abre el archivo de salida en modo binario para escritura y luego itera sobre cada archivo de entrada. Para cada archivo de entrada, lee su contenido en memoria, crea una estructura de encabezado (**FileHeader**), escribe el encabezado seguido del contenido del archivo en el archivo tar. Utiliza las funciones **fopen**, **fwrite** y **fclose** para manejar los archivos.

**Extraer archivos de un archivo tar:**

La función **extract\_tar** toma un archivo tar de entrada y una lista opcional de archivos a extraer. Abre el archivo tar en modo binario para lectura y luego itera sobre cada entrada en el archivo tar. Si no se especifican archivos para extraer, extraerá todos los archivos del archivo tar. Utiliza **fread** y **fwrite** para leer y escribir datos en archivos.

* 1. **Estructura de directorios internos (FAT)**

El programa utiliza una estructura tipo "FAT" (File Allocation Table) para organizar la información de los archivos almacenados dentro del archivo tar. Esta estructura se implementa mediante la estructura **FileHeader**, que contiene el nombre del archivo, su tamaño, la ubicación de su contenido dentro del archivo tar y una bandera que indica si el archivo ha sido eliminado. La posición de los archivos dentro del archivo tar se calcula utilizando desplazamientos relativos basados en el tamaño del archivo.

* 1. **Estrategia de manejo de bloques libres**

El programa utiliza una estrategia simple para manejar bloques libres dentro del archivo tar. Cuando se elimina un archivo, se marca su entrada correspondiente en la estructura FAT como eliminada, pero el espacio ocupado por el archivo no se recupera inmediatamente. En su lugar, se mantiene un registro de los bloques liberados. Cuando se agrega un nuevo archivo y no hay suficiente espacio libre para contenerlo, se reutilizan los bloques liberados. Si aún así el nuevo archivo no cabe, el archivo tar se hace crecer dinámicamente.

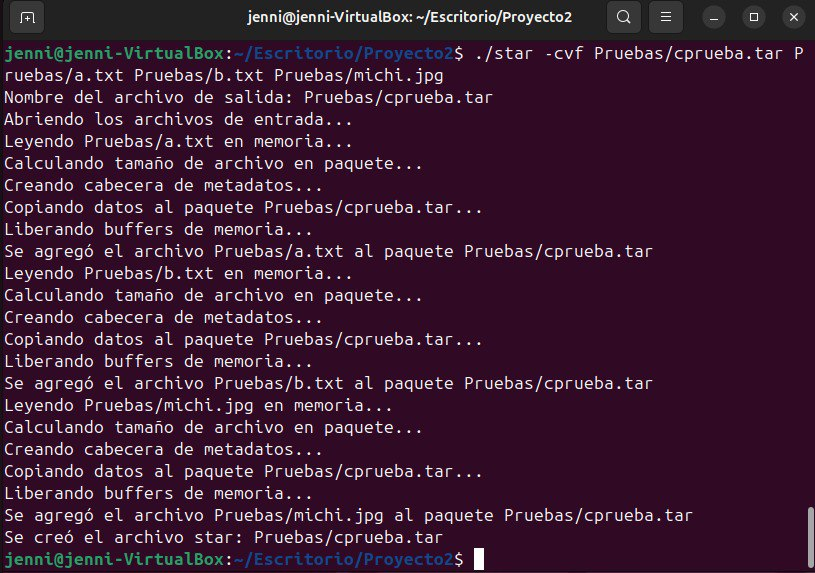
* 1. **Procedimiento de desfragmentación del archivo**

**COMPLETAR :D**

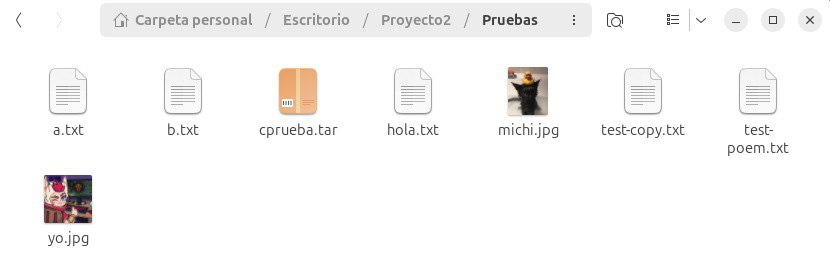
1. **Análisis de resultados de pruebas**

Para efectuar las pruebas necesarias se cuenta con una carpeta dentro del proyecto llamada Pruebas. Esta carpeta contiene 2 archivos formato jpg 5 archivos en formato txt, todos con contenidos y tamaños diferentes. Estos archivos fueron utilizados para probar los comandos que se muestran a continuación.

* -c, --create : crea un nuevo archivo



Al utilizar el comando **./star -cvf Pruebas/cprueba.tar Pruebas/a.txt Pruebas/b.txt Pruebas/michi.jpg,** se puede observar la creación del archivo **cprueba.tar,** que contiene a a.txt, b.txt y michi.jpg. A continuación, se muestra que efectivamente este archivo se creó dentro de la carpeta.

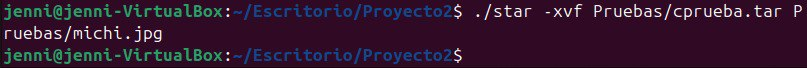


* -x, --extract : extraer de un archivo

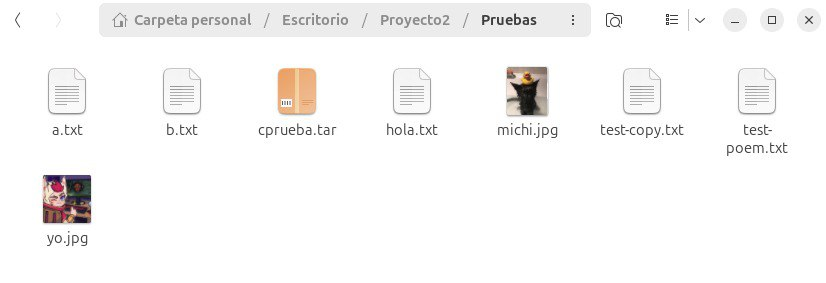
En esta prueba se utilizó el comando **./star -xvf Pruebas/cprueba.tar Pruebas/michi.jpg.** Este era el estado inicial de la carpeta Pruebas, en donde se eliminó michi.jpg para verificar visualmente su correcta extracción.



Posteriormente, se ejecutó el comando, visualizando que no se tuvo ningún problema.

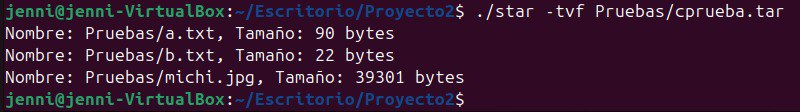


Y este fue el estado de la carpeta Pruebas luego de la ejecución:



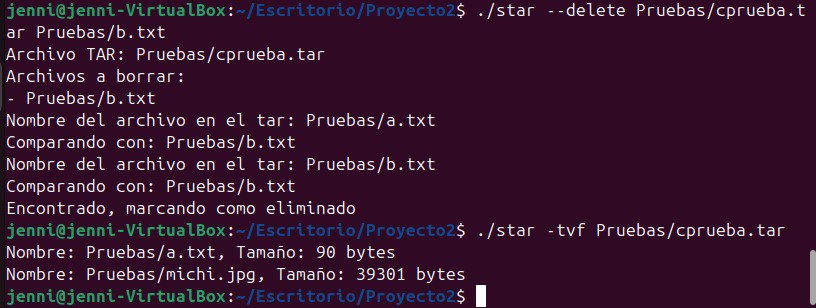
* -t, --list: listar los contenidos de un archivo

Para esta prueba se utilizó el comando **./star -tvf Pruebas/cprueba.tar,** dando como resultado un listado de los archivos dentro de la carpeta y sus tamaños correspondientes.



* --delete: borrar desde un archivo

Para esta prueba, se utilizó el comando **./star --delete Pruebas/cprueba.tar Pruebas/b.txt.** Para verificar su correcto funcionamiento, se ejecutó el comando para list, verificando que **b.txt** sí fue eliminado de **cprueba.tar.**



Además, en los mensajes impresos se logra visualizar la búsqueda del archivo a eliminar y cómo se efectúa su eliminación al ser encontrado.

* -u, --update: actualiza el contenido del archivo
* -v, --verbose: ver un reporte de las acciones a medida que se van realizando
* -f, --file: empacar contenidos de archivo, si no está presente asume la entrada estándar.
* -r, --append: agrega contenido a un archivo
* -p, --pack: desfragmenta el contenido del archivo (no presente en tar )

1. **Conclusiones**