UASLP
Universidad Autónoma
de San Luis Potosí

FACULTAD DE
INGENIERÍA

Práctica 5. Implementación de comandos básicos en Nachos.

Sistemas operativos B.
Profesora: M.I. Ortiz Hernández Marcela.
grupo: 2402-03.
Semestre: 2019-2020/II.

Cantu Olivares Pedro de Jesus..

27/Mayo/2020.

1.- Resumen:

En esta práctica se modificó el sistema de archivos para añadir ciertas funcionalidades. Aprendiendo así la manera de trabajar del sistema de archivos de nachos.

2.- Instrucciones:

La estructura de directorios implementada de forma original en nachos es de un solo nivel con soporte para 10 archivos, el objetivo de la práctica es que tenga capacidad para manipular hasta 20 archivos en el directorio raíz y se le agreguen loa siguientes comandos al sistema:
1. Comando -sfd para desplegar el número de los sectores de disco duro que están libres (se debe imprimir el número de cada sector libre en el disco duro separados por comas). La sintaxis debe ser: $./nachos -sfd
}
2. Comando -sf para desplegar los sectores que tiene asignado un archivo. La sintaxis debe ser: $./nachos -sf nombre_archivo Se debe realizar las validaciones correspondientes y marcar los errores que se presenten, como por ejemplo: que el nombre_archivo no existe, que la sintaxis es incorrecta, etc.

3. Comando -rf para renombrar un archivo. La sintaxis debe ser: $./nachos -rf nombre_archivo nombre_nuevo Se debe realizar las validaciones correspondientes y marcar los errores que se presenten, como por ejemplo: que el nombre_archivo no existe, nombre duplicado, etc.

4. Comando -man va a visualizar en terminal un manual de ayuda general, donde se muestre el nombre de cada comando(los que tiene en la versión original más los que se están agregando), la descripción de lo que hace el comando, la sintaxis y un ejemplo de como ejecutarlo. La sintaxis debe ser: $./nachos -man

5. Comando -help para mostrar en terminal la ayuda de un comando específico. Debe de visualizarse la descripción de lo que hace el comando, la sintaxis y un ejemplo de como ejecutarlo. La sintaxis debe ser: $./nachos -help nombre_del_comando

6. Comando -inf para visualizar en terminal la siguiente información a) Nombres de los integrantes del equipo b) Nombre de la materia c) Fecha de entrega de la

práctica d) Semestre: 2019-2020/II e) Nombre del profesor La sintaxis debe ser: $./nachos -inf

3. Descripción del sistema de archivos de nachos sobre la técnica de asignación de espacio en DD forma original (estructuras de datos, soporte que tiene, tamaño del disco duro, cantidad y tamaño del sector, etc.)
¿Qué estructura de directorios(de los vistos en clase) tiene implementada nachos?
nachos tiene implementado una estructura de directorios de un solo nivel.
¿Qué estructura(s) de datos se maneja(n) en la implementación de la estructura de directorios?. Escriba el archivo, la(s) clase(s) y los campos que justifiquen su respuesta.
Encontrado en : nachos/filesys/directory.h y nachos/filesys/directory.cc
class DirectoryEntry: Esta es una clase que representa a un archivo. contiene un booleano que indica si está en uso. Un entero que indica la locación del sector donde están guardados sus datos. y una cadena de caracteres de tamaño 10, lo que hace que el tamaño máximo del nombre del archivo se limite a 9 caracteres.

class Directory: Esta clase representa el directorio como tal. Tiene un constructor al que se le manda el parámetro de su tamaño, indicado como el número de archivos que podrá contener. Así como métodos para el manejo de los archivos(agregar , remover, listar, etc). Sus atributos son tablesize(número de archivos). y un apuntador de tipo directoryEntry, que basicamente sera un arreglo de DirectoryEntries(arreglo de archivos). Aquí se puede observar que la clase Directory no puede contener más directorios dentro por lo que se concluye que es de un solo nivel.

Encontrado en : nachos/filesys/filesys.h y nachos/filesys/filesys.cc
El sistema de archivos de nachos que se encarga del manejo del directorio, su creación, eliminación,etc.

soporte que tiene:
Soporta originalmente 10 archivos el directorio , y es de un solo nivel. cada sector tiene un espacio de 128 Bytes.

¿De qué tamaño es el disco duro?. Escriba el nombre de los archivos y las líneas de código que justifiquen su respuesta.
Las estructuras de datos para emular un disco físico(almacenamiento secundario) se encuentran en el archivo nachos/machine/disk.h y nachos/machine/disk.cc. En este último se define el tamaño del disco con la siguiente expresión:
#define DiskSize (MagicSize + (NumSectors * SectorSize))
Que es lo mismo que:

DiskSize = MagicSize + (NumSectors * SectorSize))
Para lo que necesitamos saber:
MagicSize = size(int) = 4 bytes.
SectorsPerTrack = 32.
NumTracks = 32.
SectorSize = 128 bytes.
NumSectors = (SectorsPerTrack * NumTracks) = 32 * 32 = 1024 sectores.
Entonces...
DiskSize = 4 + (1024 * 128) = **131,076 bytes.**

4. Descripción del sistema de archivos de nachos sobre la estructura de directorios de forma original (estructuras de datos, soporte que tiene, etc.).

filesys.h, filesys.cc | interfaz de nivel superior para el sistema de archivos.

directory.h, directory.cc | traduce nombres de archivo a encabezados de archivo de disco; La estructura de datos del directorio se almacena como un archivo.

filehdr.h, filehdr.cc | gestiona la estructura de datos que representa el diseño de los datos de un archivo en el disco. Este es el equivalente de NachOS a un i-nodo en UNIX.

openfile.h, openfile.cc | traduce lecturas y escrituras de archivos a lecturas y escrituras del sector del disco.

synchdisk.h, synchdisk.cc | proporciona acceso sincrónico al disco físico asincrónico, de modo que los subprocesos se bloquean hasta que se completen sus solicitudes.

Algunas de las estructuras de datos en el sistema de archivos NachOS se almacenan tanto en la memoria como en el disco. Para proporcionar cierta uniformidad, todas estas estructuras de datos tienen un procedimiento \ FetchFrom "que lee los datos del disco y en la memoria, y un procedimiento \ WriteBack" que almacena los datos nuevamente en el disco. Hay que tener en cuenta que las representaciones en memoria y en disco no tienen que ser idénticas.

5. Descripción de las modificaciones realizadas para que el sistema de archivos en nachos pueda soportar el manejo de 20 archivos en el directorio raíz. Además, anexar el código.

Se hizo el cambio del valor para la constante NumDirEntries en ./nachos/filesys/filesys.cc de 10 a 20.

```
#define NumDirEntries          20
```

6. Para cada uno de los comandos -sfd, sf, rf, -man,- help y inf realizar lo siguiente

a) Descripción de las modificaciones realizadas para su implementación.

b) Anexar el código comentado.

c) Captura de pantalla donde se aprecie la ejecución del comando.

1.-Aquí el primer cambios son muy parecidos:

Archivo : ./nachos/threads/main.cc.

Descripción:

llamando a funciones externas en ./nachos/threads/main.cc

Código:

```
/************************ Practrica 5*********************/
extern void Manual();
extern void Help(char* comando);
extern void Info();
```

2.-Archivo : ./nachos/threads/main.cc.

Descripción:

Especificación para cada comando añadido a el sistema de archivos.

Código:

para la comparación del comando(bandera).

```
#ifdef FILESYS
    if (!strcmp(*argv, "-cp")) {        // copy from UNIX to Nachos
        fileSysNoExiste = true;
        ASSERT(argc > 2);
        Copy(*(argv + 1), *(argv + 2));
        argCount = 3;
    } else if (!strcmp(*argv, "-p")) {  // print a Nachos file
        fileSysNoExiste = true;
        ASSERT(argc > 1);
        Print(*(argv + 1));
```

```
        argCount = 2;
    } else if (!strcmp(*argv, "-r")) {  // remove Nachos file
        fileSysNoExiste = true;
        ASSERT(argc > 1);
        fileSystem->Remove(*(argv + 1));
        argCount = 2;
    } else if (!strcmp(*argv, "-l")) {  // list Nachos directory
        fileSysNoExiste = true;
            fileSystem->List();
    } else if (!strcmp(*argv, "-D")) {  // print entire filesystem
        fileSysNoExiste = true;
            fileSystem->Print();
    } else if (!strcmp(*argv, "-t")) {  // performance test
        fileSysNoExiste = true;
            PerformanceTest();
    }
/************************************************************************
Practica 5. para las nuevas implementaciones en el sistema de archivos
FileSystem.
************************************************************************
/
     else if(!strcmp(*argv, "-sfd"))   {   //imprime los sectores libres
del disco.
        fileSysNoExiste = true;
            fileSystem->Print_LibresSectores();
    }
     else if(!strcmp(*argv, "-sf"))   {    // imprime los sectores del
archivo especificado.
        fileSysNoExiste = true;
        //ASSERT(argc > 1);
        if(argc > 1){
            fileSystem->Print_ArchivoSectores(*(argv + 1));
        }
        else{// si solo hay un argumento, lo hace notar al usuario
                        printf("\nNo  se  especifico  ningun  nombre  de
archivo...\n\n");
            printf("Sintaxis correcta: ./nachos -sd nombre_archivo\n");
            interrupt->Halt();
        }
    }
```

```
      else if(!strcmp(*argv, "-rf"))    { // cambia el nombre de un
archivo.
        fileSysNoExiste = true;
        if(argc > 2){
            fileSystem->Renombra(*(argv + 1) , *(argv + 2));
        }
        else{
                 printf("\nNo se especificaron los nombres de archivo
correctamente...\n");
                printf("Sintaxis correcta: ./nachos -rf nombre_archivo
nombre_nuevo\n");
            interrupt->Halt();
        }
    }
     else if(!strcmp(*argv, "-inf")){ // imprime informacion del equipo
y materia.
        fileSysNoExiste = true;
            Info();
            interrupt->Halt();
        }
         else if(!strcmp(*argv, "-man")){ // imprime informacion general
de los comandos de nachos.
            fileSysNoExiste = true;
             Manual();
             interrupt->Halt();
        }
           else if(!strcmp(*argv, "-help")){ //imprime informacion del
comando especificado.
            fileSysNoExiste = true;
            if(argc > 1){
                Help(*(argv + 1));
                interrupt->Halt();
            }
            else{
                    printf("\nSintaxis incorrecta intenta con: ./nachos
-help nombre_del_comando ");
                interrupt->Halt();
            }
        }
        else if(fileSysNoExiste == false && strcmp(*argv, "-f")){
```

```
            printf("\n\nEl comando %s no fue encontrado en el sistema
de archivos.\n\n",*argv);
             printf("\nIntenta con el comando -man para saber cuales
comandos estan disponibles..\n\tSintaxis:  ./nachos -man\n\n");
           fileSysNoExiste = true;
       }

#endif // FILESYS
```

NOTA: podemos clasificar las implementaciones en las de ftest y otras del filesystem como tal. En ftest hay tres funciones para la impresión de información y ayudar al usuario, como con el manual(-man).

2.-Archivo : ./nachos/filesys/ftest.cc.
Descripción:
Definición de funciones, Info(), man() y help().Para imprimir información para el usuario.
Código:

```
/************************************************************************
*************
Practica 5: implementacion para el manual y el despliegue de
informacion de comandos
************************************************************************
************/
void Manual()
{
     printf("\n\n°° MANUAL DE COMANDOS PARA EL SISTEMA DE ARCHIVOS
°°\n\n");


printf("\n\n-------------------------------------------------------------
-------------------\n\n");
       printf("\n\nNOMBRE:\n\t%s.","-f");
       printf("\n\nSINTAXIS:\n\t./nachos -f ");
         printf("\n\nDESCRIPCION:\n\tFormatea el disco de nachos para
ser usado.\n\n");
```

```
printf("\n\n-------------------------------------------------------------
--------------------\n\n");


printf("\n\n-------------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s.","-cp");
             printf("\n\nSINTAXIS:\n\t./nachos  -cp  Nombre_Archivo_Unix
Nombre_Archivo_Nachos.");
            printf("\n\nDESCRIPCION:\n\tCopia un archivo de UNIX a el
directorio de nachos como un archivo de nachos.");
          printf("\n\nNOTA:\n\tEl nombre del archivo de nachos no debe
exceder los 9 caracteres.\n\n");

printf("\n\n-------------------------------------------------------------
--------------------\n\n");


printf("\n\n-------------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-p");
        printf("\n\nSINTAXIS:\n\t./nachos -p Nombre_Archivo_Nachos");
           printf("\n\nDESCRIPCION:\n\tImprime las caracteristicas del
archivo indicado.\n\n");

printf("\n\n-------------------------------------------------------------
--------------------\n\n");



printf("\n\n-------------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-r");
        printf("\n\nSINTAXIS:\n\t./nachos -r Nombre_Archivo_Nachos");
          printf("\n\nDESCRIPCION:\n\tElimina del directorio el archivo
indicado.\n\n");

printf("\n\n-------------------------------------------------------------
--------------------\n\n");
```

```
printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-l");
        printf("\n\nSINTAXIS:\n\t./nachos -l");
           printf("\n\nDESCRIPCION:\n\tImprime todos los archivios del
directorio de nachos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");



printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-D");
        printf("\n\nSINTAXIS:\n\t./nachos -D");
         printf("\n\nDESCRIPCION:\n\tImprime todo el sistema de archivos
actual(instanciado) de nachos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");



printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-t");
        printf("\n\nSINTAXIS:\n\t./nachos -t");
          printf("\n\nDESCRIPCION:\n\tPrueba que funcione correctamente
el sistema de archivos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");

/**************************************************************
Practica 5. para las nuevas implementaciones en el sistema de archivos
FileSystem.
```

```
**********************************************************************
/

printf("\n\n----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-sfd");
        printf("\n\nSINTAXIS:\n\t./nachos -sfd");
          printf("\n\nDESCRIPCION:\n\tImprime todos sectores libres del
sistema de archivos.\n\n");

printf("\n\n----------------------------------------------------------
--------------------\n\n");



printf("\n\n----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-sf");
        printf("\n\nSINTAXIS:\n\t./nachos -sf Nombre_Archivo_Nachos");
          printf("\n\nDESCRIPCION:\n\tImprime todos sectores usados por
el archivo indicado.\n\n");

printf("\n\n----------------------------------------------------------
--------------------\n\n");



printf("\n\n----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-rf");
            printf("\n\nSINTAXIS:\n\t./nachos  -rf  Nombre_Archivo_Nachos
Nuevo_Nombre");
             printf("\n\nDESCRIPCION:\n\tCambia  el  nombre  del  archivo
indicado a el nuevo nombre.");
          printf("\n\nNOTA:\n\tEl nombre del nuevo archivo de nachos no
debe exceder los 9 caracteres.\n\n");

printf("\n\n----------------------------------------------------------
--------------------\n\n");
```

```
printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-man");
        printf("\n\nSINTAXIS:\n\t./nachos -man");
                printf("\n\nDESCRIPCION:\n\tDespliega este manual con
informacion de todos los comandos\n\tdel sistema de archivos.");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");


printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-help");
        printf("\n\nSINTAXIS:\n\t./nachos -help Comando.");
            printf("\n\nDESCRIPCION:\n\tDespliega informacion sobre el
comando especificado.");
         printf("\n\nNOTA:\n\tEl comando debe existir para el sistema de
archivos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");


printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s","-inf");
        printf("\n\nSINTAXIS:\n\t./nachos -info");
            printf("\n\nDESCRIPCION:\n\tDespliega informacion sobre el
equipo de trabajo.");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");

}

void Help(char* comando)
{
    if (!strcmp(comando, "-f")) {
```

```
printf("\n\n-------------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s.",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -f ");
          printf("\n\nDESCRIPCION:\n\tFormatea el disco de nachos para
ser usado.\n\n");

printf("\n\n-------------------------------------------------------------
--------------------\n\n");
    }
    else if (!strcmp(comando, "-cp")) {

printf("\n\n-------------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s.",comando);
             printf("\n\nSINTAXIS:\n\t./nachos  -cp  Nombre_Archivo_Unix
Nombre_Archivo_Nachos.");
            printf("\n\nDESCRIPCION:\n\tCopia un archivo de UNIX a el
directorio de nachos como un archivo de nachos.");
          printf("\n\nNOTA:\n\tEl nombre del archivo de nachos no debe
exceder los 9 caracteres.\n\n");

printf("\n\n-------------------------------------------------------------
--------------------\n\n");
    } else if (!strcmp(comando, "-p")) {  // print a Nachos file

printf("\n\n-------------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -p Nombre_Archivo_Nachos");
           printf("\n\nDESCRIPCION:\n\tImprime las caracteristicas del
archivo indicado.\n\n");

printf("\n\n-------------------------------------------------------------
--------------------\n\n");
    } else if (!strcmp(comando, "-r")) {  // remove Nachos file

printf("\n\n-------------------------------------------------------------
--------------------\n\n");
```

```
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -r Nombre_Archivo_Nachos");
          printf("\n\nDESCRIPCION:\n\tElimina del directorio el archivo
indicado.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
    } else if (!strcmp(comando, "-l")) {  // list Nachos directory

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -l");
           printf("\n\nDESCRIPCION:\n\tImprime todos los archivios del
directorio de nachos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
    } else if (!strcmp(comando, "-D")) {  // print entire filesystem

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -D");
         printf("\n\nDESCRIPCION:\n\tImprime todo el sistema de archivos
actual(instanciado) de nachos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
    } else if (!strcmp(comando, "-t")) {  // performance test

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -t");
          printf("\n\nDESCRIPCION:\n\tPrueba que funcione correctamente
el sistema de archivos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
```

```
    }
/*****************************************************************
Practica 5. para las nuevas implementaciones en el sistema de archivos
FileSystem.
*****************************************************************
/
    else if(!strcmp(comando, "-sfd"))   {  //imprime los sectores libres
del disco.

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -sfd");
          printf("\n\nDESCRIPCION:\n\tImprime todos sectores libres del
sistema de archivos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
    }
     else if(!strcmp(comando, "-sf"))   {    // imprime los sectores del
archivo especificado.

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -sf Nombre_Archivo_Nachos");
          printf("\n\nDESCRIPCION:\n\tImprime todos sectores usados por
el archivo indicado.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
    }
      else if(!strcmp(comando, "-rf"))   { // cambia el nombre de un
archivo.

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
            printf("\n\nSINTAXIS:\n\t./nachos  -rf  Nombre_Archivo_Nachos
Nuevo_Nombre");
```

```
            printf("\n\nDESCRIPCION:\n\tCambia el nombre del archivo
indicado a el nuevo nombre.");
         printf("\n\nNOTA:\n\tEl nombre del nuevo archivo de nachos no
debe exceder los 9 caracteres.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
    }
      else if(!strcmp(comando, "-man"))   { // cambia el nombre de un
archivo.

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -man");
          printf("\n\nDESCRIPCION:\n\tDespliega informacion sobre todos
los comandos del sistema de archivos.");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
    }
     else if(!strcmp(comando, "-help"))   { // cambia el nombre de un
archivo.

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -help Comando.");
            printf("\n\nDESCRIPCION:\n\tDespliega informacion sobre el
comando especificado.");
         printf("\n\nNOTA:\n\tEl comando debe existir para el sistema de
archivos.\n\n");

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
    }
    else if(!strcmp(comando, "-inf"))  { //info

printf("\n\n-----------------------------------------------------------
--------------------\n\n");
```

```
        printf("\n\nNOMBRE:\n\t%s",comando);
        printf("\n\nSINTAXIS:\n\t./nachos -info");
            printf("\n\nDESCRIPCION:\n\tDespliega informacion sobre el
equipo de trabajo.");

printf("\n\n------------------------------------------------------------
--------------------\n\n");
    }
    else
    {
          printf("\n\nEl comando %s no fue encontrado en el sistema de
archivos.\n",comando);
             printf("\nIntenta con el comando -man para saber cuales
comandos estan disponibles y su sintaxis...\n\tSintaxis:   ./nachos
-man\n\n");
    }
}

void Info()
{
            printf("\n\n----------------      Informacion del equipo
-------------------------\n\n");

    printf("\nIntegrantes:\n");
    printf("\tCantú Olivares Pedro de Jesus.\n");

    printf("\nMateria:\n");
    printf("\tSistemas Operativos B\n");

    printf("\nFecha de entrega:\n");
    printf("\t17/Mayo/2020\n");

    printf("\nSemestre:\n");
    printf("\t2019-2020-II\n");

    printf("\nMaestra:\n");
    printf("\tMarcela Ortiz Hernández\n");

printf("\n\n------------------------------------------------------------
---------\n\n");
```

```
}
```

NOTA: Aqui esta la parte que toma forma dentro del sistema de archivos como tal.

3.-Archivo : ./nachos/filesys/filesys.h.

Descripción:

Declaración de los métodos implementadas en la práctica 5, para la clase Filesys.

Código:

```
/**************************************************************************
**************
      Practica 5: definiciones de los metodos implementados en la
practica 5 para la clase Filesystem.
**************************************************************************
***************/
    void Renombra(char* nombreArchivo, char* nuevoNombre);//metodo para
renombrar un archivo.
      void Print_LibresSectores();//metodo para imprimir los sectores
libres del disco.
       void  Print_ArchivoSectores(char  *nombreArchivo);//metodo  para
imprimir los sectores utilizados por un archivo especificado.
```

4.-Archivo : ./nachos/filesys/filesys.h.

Descripción:

Definición de los métodos implementadas en la práctica 5, para la clase Filesys.

Código:

```
/************************************************************
*
*   Practica 5: implementacion de los metodos.
*
************************************************************/
//-----------------------------------------------------------------------
-
// FileSystem::Remombra
//  Renombra un archivo especificado a un nombre nuevo dado
//  llamando al metodo de renombra del directorio.
//-----------------------------------------------------------------------
-
```

```
void FileSystem::Renombra(char* nombreArchivo, char* nuevoNombre)
{
    Directory *directory = new Directory(NumDirEntries);
    FileHeader *hdr = new FileHeader();
    directory->FetchFrom(directoryFile);
    int sector = directory->Find(nombreArchivo);

    if (sector != -1)
    {
        hdr->FetchFrom(sector);

        if(directory->Renombra(nombreArchivo,nuevoNombre))
        {
                    printf("\nEl archivo %s ah sido renombrado como %s
exitosamente\n\n",nombreArchivo,nuevoNombre);
            hdr->WriteBack(sector);
            directory->WriteBack(directoryFile);
            delete directory;
            return;
        }
            printf("El archivo %s no cambio...\n",nombreArchivo);

    }

}

//----------------------------------------------------------------------
-
// FileSystem::Print_LibresSectores
//  imprime los sectores libres del disco.
//----------------------------------------------------------------------
-
void FileSystem::Print_LibresSectores()
{
    int sectoresLibres = 0;//contador para los sectores libres.

    BitMap *map;
    map = new BitMap(NumSectors);
    map->FetchFrom(freeMapFile);
```

```
    printf("\nSectores vacios en el disco:\n");
    for (int i = 0; i < map->getNumBits(); i++)
    {
        if (!map->Test(i))
        {
            printf("%d, ", i);
            sectoresLibres ++ ;
        }
    }
    printf("\nSectores Totales: %d", NumSectors);
    printf("\nSectores Libres: %d",sectoresLibres);
    printf("\nSectores Ocupados: %d", NumSectors-sectoresLibres);
    delete map;
}

//----------------------------------------------------------------------
-
// FileSystem::Print_LibresSectores
//  imprime los sectores utilizados por un archivo especificado.
//----------------------------------------------------------------------
-
void FileSystem::Print_ArchivoSectores(char *nombreArchivo)
{
    Directory *directory = new Directory(NumDirEntries);
    FileHeader *hdr = new FileHeader();
    directory->FetchFrom(directoryFile);
    int sector = directory->Find(nombreArchivo);

    if (sector != -1)
    {
        printf("\nSector del i-nodo: %d\n", sector);
        hdr->FetchFrom(sector);
        printf("\nSectoresLibres:\n");
        hdr->Print_Sectores();
        printf("\n");
        delete directory;
    }
    else
    {
        printf("\nNo se encontro el archivo especificado\n\n");
```

```
    }
}
```

5.-Archivo : ./nachos/filesys/directory.h.
Descripción:
Declaración del método para renombrar un archivo dentro de un directorio.
Código:

```
/************************************************************************
**********
          Practica5: Definicion de los metodos agregados a la clase
Directory.

************************************************************************
************/
     bool Renombra(char* nombreArchivo, char* nuevoNombre);//definicion
del metodo para renombrar un archivo dentro del directorio.
```

6.-Archivo : ./nachos/filesys/directory.cc.
Descripción:
Definición del método para renombrar n archivo dentro de un directorio.
Código:

```
/************************************************************************
******
    Practica 5: Metodo para renombrar un archivo dentro del directorio.
************************************************************************
******/
bool Directory:: Renombra(char* nombreArchivo, char* nuevoNombre)
{
    bool exitoRes = FALSE;
    int indiceDelArchivo = FindIndex(nombreArchivo);

    if(indiceDelArchivo >= 0)// el archivo fue encontrado.
    {
        int nuevoNombreTam = strlen(nuevoNombre);
         if(nuevoNombreTam > 9)//el nuevo nombre no cumple la condicion
de tamaño del nombre.
        {
```

```
                printf("\nEl nuevo nombre %s exede el tamaño maximo de
nombre de archivo\n\n",nuevoNombre);
                exitoRes = FALSE;
            }
            else// nos ponemos a trabajar.
            {
                                                printf("\nRenombrando    %s
...\n\n",table[indiceDelArchivo].name);
                        strncpy(table[indiceDelArchivo].name,nuevoNombre,
FileNameMaxLen);
                exitoRes = TRUE;
            }
        }
        else//el archivo no fue encontrado, informamos al usuario.
        {
            printf("El archivo no existe en el directorio\n");
            exitoRes = FALSE;
        }
        return exitoRes;
}
```

7.-Archivo : ./nachos/filesys/filehdr.h.
Descripción:
Declaración del método para imprimir los sectores de un archivo..
Código:

```
/************************************+
Practica 5:
*********************************/
  void Print_Sectores();
```

8.-Archivo : ./nachos/filesys/filehdr.cc.
Descripción:
Definición del método para imprimir los sectores de un archivo..
Código:

```
/************************************+
Practica 5:
*********************************/
//-----------------------------------------------------------------------
-
// FileHeader::Print_Sectores()
//  Imprime los sectores pertenecientes al archivo.
//-----------------------------------------
void
FileHeader::Print_Sectores()
{
    for(int i = 0 ; i< numSectors; i++)
    {
        printf("%d, ",dataSectors[i]);
    }
printf("\n");
}
```

9.-Archivo : ./nachos/userprog/bitmap.h.
Descripción:
Declaración del método get,para obtener el número de bits del mapa de bits.
Código:

```
  /************* Practica 5**********************/
int getNumBits();// para obtener el numero de bits.
```

10.-Archivo : ./nachos/userprog/bitmap.cc.
Descripción:
Método get para obtener el número de bits, del mapa de bits.
Código:

```
/************************
Practica 5: Obtiene el numero de bits en el mapa de bits
*******************/
int BitMap::getNumBits()
{
    return this->numBits;
}
```

7. Problemas presentados y soluciones durante el desarrollo de la práctica.
Entender como hacer para imprimir todo lo necesario.No pude imprimir un aviso para cuando un archivo ya existía en el directorio de nachos.

8. Conclusiones:
Con la realización de esta práctica nos familiarizamos con la estructura de archivos. Asi mismo se concreto el conocimiento que se acumulo en las clases sobre los inodos y otros temas importantes sobre archivos y directorios.