MACHINE PROBLEM 7

OBJECTIVE: To implement a simple file system.
BONUS OPTION 1 IS ATTEMPTED

IMPLEMENTATION:
The inode and free_block lists were used in the same way as already defined.
The following functions need to be implemented in the below file:

file_system.C-

FileSystem() - constructor for class FileSystem, all the variables in memory were initialized.
~FileSystem() - destructor for class FileSystem, inode and free_block lists were saved to disk.
Mount()- Associates a file system with a disk, reads the inode and free block list into memory from block 0 and block 1 respectively.
Format()- populates the disk with empty inode list, and a free block list with only block 0 and 1 marked as used.
LookupFile()- iterates over the inode list to find the matching ID file and return a class inode object of that file.
CreateFile()- checks if the file already exists, if yes then it throws a error otherwise finds a empty data block and initilizes an inode. Also marks the data block as used.
DeleteFile()- throws an error if the file doesn't exist, otherwise marks the inode and data block as free.
load_FL()- loads the free block list from disk from block 1. ->new function
load_IL()- loads the inode list from disk from block 0. ->new function
save_FL()- saves the free block list to disk to block 1. ->new function
save_IL()- saves the inode list to disk to block 0. ->new function
get_block_no()- finds the block no from a given file ID. ->new function

file.C-
File() - constructor for class File, opens the file, maintains position and other variables and reads the data block from disk to block_cache.
~File() - writes the block_cache to disk.
Read() - reads _n characters starting at the current position from the block cache and returns the number of characters read.
Write() - writes _n characters to the block cache starting from the current position and returns the number of characters written.
Reset()- sets the current position to the beginning of the file.
EoF()- checks if the current position is the end of the file.

Other files changed:-
file_system.H- added block_no variable in class inode and the new functions in file_system.C in class FileSystem.
file.H- added variables for storing current position etc.

BONUS OPTION 1 explanation:

Changes in class Inode:

1) Instead of a single variable block_no to store the single block of each file, it needs to be changed to an array, it can be fixed size array of length 128(64kb/512) or we can declare a pointer and dynamically allocate it. we can also maintain a variable size indicating the number of blocks allocated to this file. The order of the blocks in the array will determine how the files will be written, first data block written to the first block entry in the array and so on.

Changes in class FileSystem:

1) In CreateFile() function, we need to pass a size parameter indicating the size of the file, so that we can allocate required number of blocks from free_blocks list.

Changes in class File:

1) We need to maintain a current block variable also in addition to the position variable we are already maintaining.
2) We need to increase the block cache size to hold more than one block.
3) While reading and writing we need to change our loops to take care of reading/writing from multiple blocks.
4) Our Reset function will be modified to set the current block also to zero.
5) Our EOF function will be modified to check if the current block is also last, this we can get to know from the size of the file.

TESTING - testing is done using kernel.C, no assertion error was observed.

```
Please choose one: [6] 6
00000000000i[      ] installing x module as the Bochs GUI
00000000000i[      ] using log file bochsout.txt
Installed exception handler at ISR <0>
Allocating Memory Pool... done
Installed interrupt handler at IRQ <0>
Installed interrupt handler at IRQ <14>
In file system constructor.
Hello World!
formatting disk
mounting file system from disk
creating file with id:1
creating file with id:2
Opening file.
Opening file.
writing to file
writing to file
Closing file.
Closing file.
Opening file.
Opening file.
resetting file
reading from file
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
resetting file
reading from file
```


checking for EoF
resetting file
reading from file
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
checking for EoF
Closing file.
Closing file.
deleting file with id:1
deleting file with id:2
creating file with id:1
creating file with id:2
Opening file.
Opening file.
writing to file
writing to file
Closing file.
Closing file.
Opening file.
Opening file.