README

FILE SYSTEM MANAGER

This project file system manager supports the file storing, retrieving, and deleting files from the storage device.

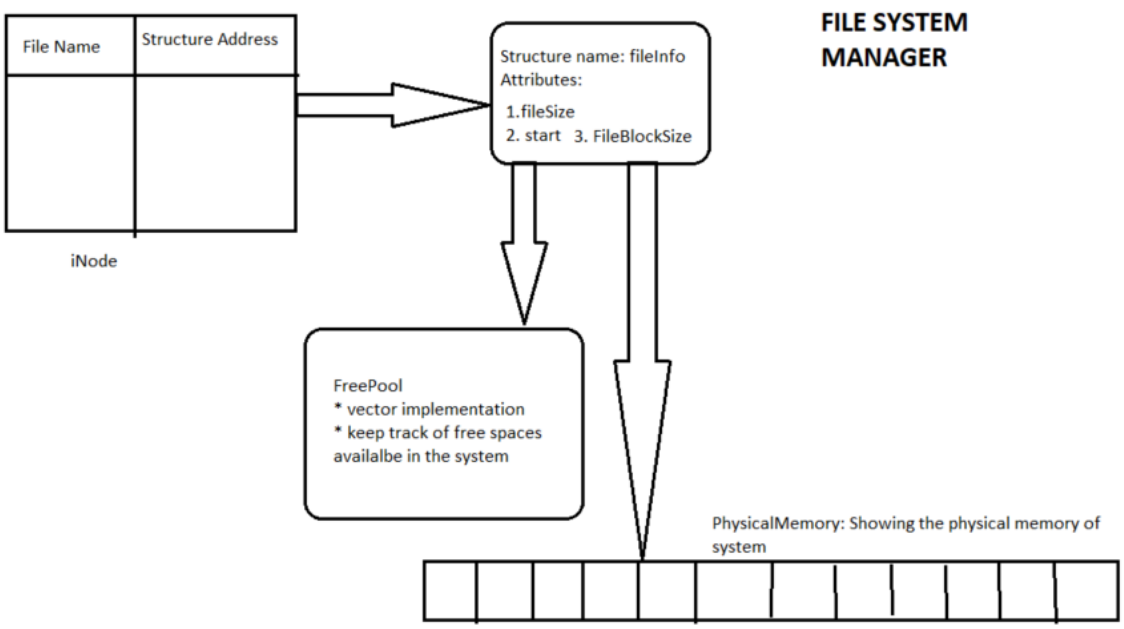
LANGUAGE: C++

DESIGN AND IMPLEMENTATION:

This is an object-oriented design implementation. In this project there is multiple class:

- Storage class: this class handles all things related to the memory location, adding, retrieving, and deleting the file.
- InputAdapter class: This class performs the input validation and input conversion from gb,mb to kb. NOTE: kb is considered as the lowest unit for this system.

The below image shows the actual design of the system.

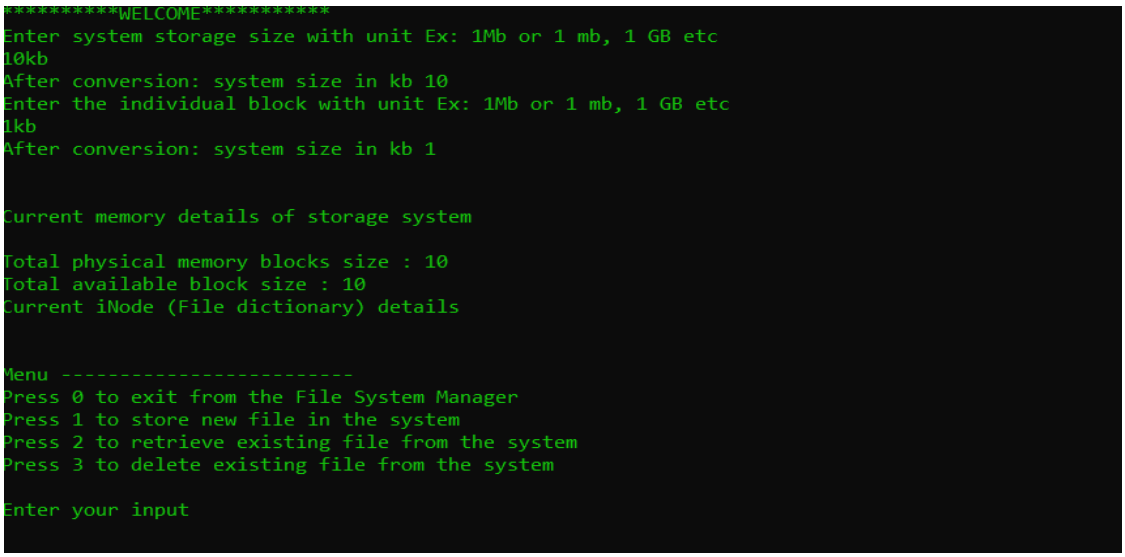


- Any file that has to be created its details are stored in iNode hash, where the key is filename and value is structure address(fileInfo).
- FileInfo is the structure that keeps the record of the individual file, that is file Size, block size and start address of a file in LinkedList.
- The linked list is the one that contains the actual data address. Address from the individual blocks from the physical memory is filled in the linked list and deleting from the free pool space.

FUNCTIONALITY:

1. **Input:** This project takes input in the form of a string (Ex: 1MB, 1 mb, or 1 Mb). The input handler class will internally break the string in the unit and actual value and will do the actual memory calculation (1MB = 1024kb). NOTE: kb is considered as the smallest unit in this system as byte may exceed data type size.
2. **Add FIle:** When the user enters file name and file size.
    a. Entry in an iNode hash is created with the filename as a key and value as a structure address.
    b. The structure will hold the file size and file start address of a linked List.
    c. The linked list is created with required blocks, that will store the address of the physical Memory blocks. Also keeping in mind that the block is available, free Pool vector will keep the record of available blocks.
3. **Retrieve file**: When the user enters the filename
    a. It is checked in the iNode hash, whether the file is available or not. If the filename is present in the iNode. The system will read the structure and display the linked list used for that file.
4. **Deletion:** When the user enters the file name
    a. Firstly, the linked list retrieve from the structure is deleted and stored in the free Pool space.
    b. Now the file detail from the structure and iNode is also deleted.

OUTPUT:



AddFile : Here a file name Mayank with size of 5kb is created.

```
Enter your input
1
Enter the file name
mayank
Enter the size of the file value and unit
5kb
After conversion: file size in kb 5
Initializing the file creation process
printing file size from the iNode 5
Total physical memory blocks size : 10
Total available block size : 5
Current iNode (File dictionary) details
File Name --> mayank   File Structure Address -->01442998
```

Retrieve file details: This will give the details of the file. Basically address which will be used to store data in file.

```
Menu -------------------------
Press 0 to exit from the File System Manager
Press 1 to store new file in the system
Press 2 to retrieve existing file from the system
Press 3 to delete existing file from the system

Enter your input
2
Enter the file name
mayank


printing address of the file
0143B854 0143B850 0143B84C 0143B848 0143B844


Total physical memory blocks size : 10
Total available block size : 5
Current iNode (File dictionary) details
File Name --> mayank   File Structure Address -->01442998
```

Delete File: Now deleting file Mayank will further add those address spaces in the free pool.

```
Enter your input
3
Enter the file name
mayank

Total physical memory blocks size : 10
Total available block size : 10
Current iNode (File dictionary) details


Menu -------------------------
Press 0 to exit from the File System Manager
Press 1 to store new file in the system
Press 2 to retrieve existing file from the system
Press 3 to delete existing file from the system
```