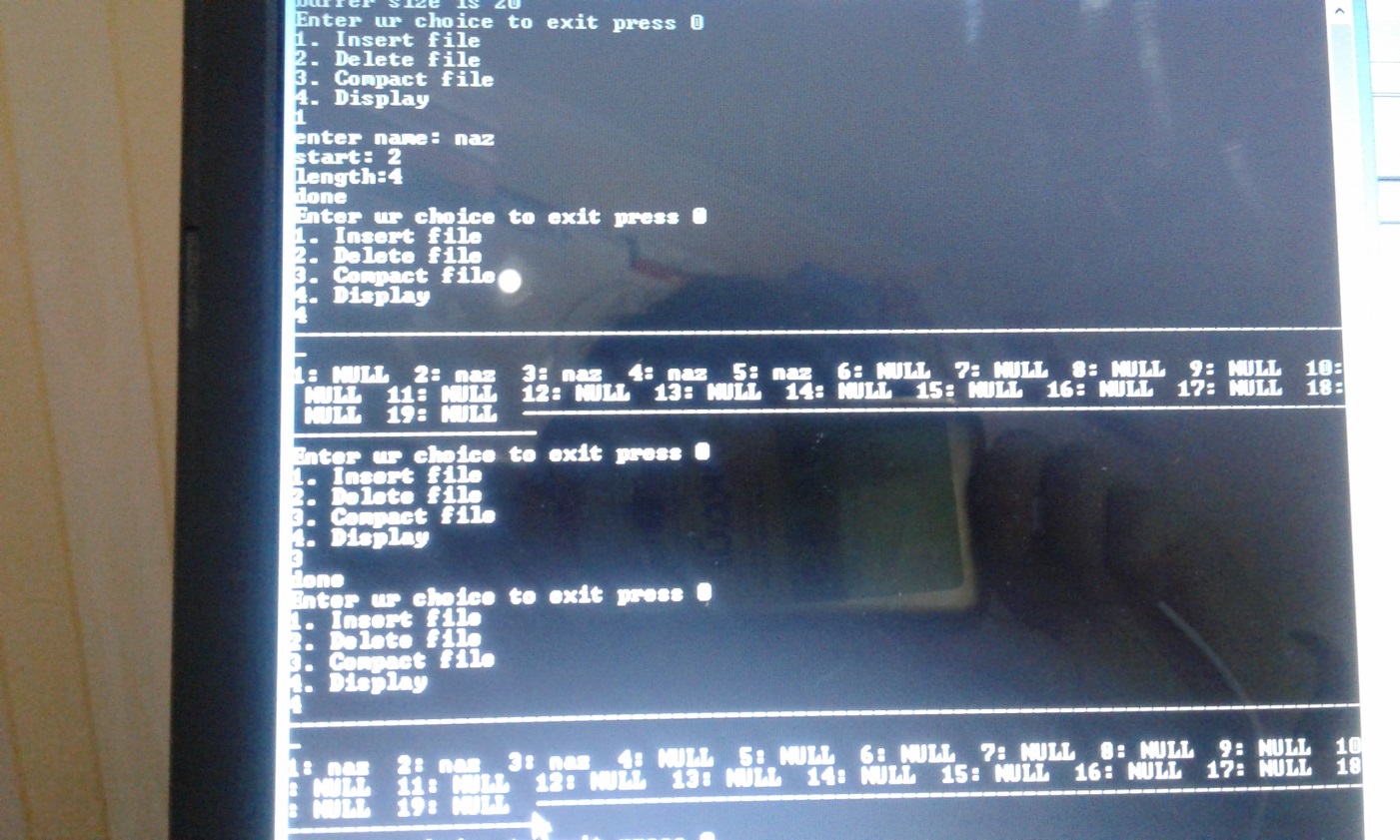
At all, file allocation methods define how files are stored in the disk blocks. So here we have three of them:

1)Contiguous allocation:

Here each file will occupy set of blocks(one by one) on the disk. In the program we have provide for customer many 4 choices written in the menu() function and each time we did some option we call this function continuously. So we have insert() function which will need from customer file name and its start location. Then it will need length, then within in loop until the given length we will write for each element of our vector<string> buf(20, "NULL") our file name. If user enters the delete option, it will check typed filename with names in our buffer if they are matched it will write there NULL. Then if user needs to compact, it will locate each not null element of buffer to free spaces .(just swapping function) . Also we have display option which shoes overall our buffer view and exit option. Though this method provides fast direst access, it suffers from both internal and external fragmentation. Result as follows:

2)Linked List allocation: 

As the name says here each file will be a linked list which is not necessarily one by one. As the directory entry can contain starting and ending file block we in the program created some structure for containing data and index, next node.Here we also have 4 options as add\_node, delete,display and concatenate. This is very flexible related to file size,as it can be easily increased. Here we distribute file blocks randomly(int j = rand() % 20;) on the disk. Because of this large number of seeks are needed to access every block individually. So it is much slower.

3)Indexed allocation:

Its program works similarily to LL, but here we do not nead enter starting and ending points, here customer just types buffer size and index block. Index block will contain pointers to all the blocks occupied by a file. So because of this, compared to LL when we write function void display() we need two parameters and the first one is our this index block . The ith entry in the index block will contain the disk address of the ith file block, as each file has its own index block. Though it provides fast access to the file blocks, its pointer overhead is much grater.