Module 8 (Lectures 34-37) File systems

1. Create a text file called **Testfile** of size 5 KBytes on the hard disk of your system. Copy it to a flash memory stick. Write a program to display the contents of a file as follows: read the first 512Bytes and print them out to stdout. Then read the next 512Bytes and print them out to stdout, etc. You can structure this as a *for* loop that iterates 10 times. Measure the execution time of the program when used to display the file **Textfile** (i) from your hard disk, (ii) from your flash memory stick. What do you observe from the measurements?
2. Instrument the program that you wrote for question 6 to measure the execution time for each of the 10 iterations of the *for* loop. Use this instrumented program to display **TextFile** from the hard disk. What do you observe from the measurements? Now, modify the program so that instead of displaying the entire contents of the file, it displays the first 512Bytes of the file 10 times. Repeat the measurements for this modified program. What do you observe from the measurements?
3. Modify the program of question 6 to use memory mapped I/O instead of file system operations. How does this affect the program execution time if the file is (i) on hard disk, (ii) on flash memory stick.
4. The program **cat** available in Linux/UNIX provides the functionality required of the program for question 6. Study the implementation of cat. Is it superior to that of the programs that you have developed in the previous questions?
5. We have seen that files can be used as a means of Inter Process Communication (IPC). Write a program which uses *fork*( ) to create a child process. The child process prompts the user to type in a line of text, which it then reads in, writes into a file called **UserInput** and exits. The parent process, which has been waiting for the child process to exit, then reads the line of text from the file **UserInput** and prints it to stdout.