COMPUTER SCIENCE TRIPOS Part IA – 2012 – Paper 2

4 Operating Systems (IML)

Consider the following scheme for structuring a file from a set of disk blocks. A disk block contains 4096 bytes and a block address is 32 bits. The first block of the file contains the following information:

control information:	1024 bytes
direct block pointers:	1024 bytes
indirect block pointer:	4 bytes
double indirect block pointer:	4 bytes
immediate data:	2040 bytes

The data bytes of the file start at the beginning of the immediate data. After the immediate data, the file data is found on the block addressed by the first direct block pointer and then carries on in a fashion similar to the structure defined by a Unix inode. We consider the first byte of the file to be byte 0, then byte 1, etc.

(*a*) For each of the following describe the actions taken to fetch the indicated byte of a file, and state how many disk blocks may need to be read:

(*i*) byte 70 of the file [1 mark]

(*ii*) byte $2^{20} + 2044$ [1 mark]

(*b*) How large can a file be if it is to be guaranteed that only three disk blocks need to be read in order to access any given byte of the file? [4 marks]

(*c*) Information about a file can be stored in a directory that references the file or in the control part of the first block of the file (i.e. inode in Unix). Which of these is used in the Unix file system to store the following information and why?

(*i*) time of creation [3 marks]

(*ii*) file name [3 marks]

(*iii*) file access rights [3 marks]

(*d*) Another way to structure files on a disk is to use physically contiguous blocks (with contiguous addresses), so that if the first block of a file is block b, then the next block of the file is $b + 1$. Suppose we use this method, retain the control information on the first block, but include the first 3 KBytes of the file in the first block. Comment on the performance of such a system, considering reading, writing, and creating files. [5 marks]