Comp 3500 Homework 1

Has0027

1. **a)** The method in figure 1, contiguous allocation, places files in order from block to block. The method in figure 2, chained allocation, uses a pointer at the end of each block to point to the block containing the next file. The major problem with figure 1 is external fragmentation. When files are deleted and extra blocks are made available for use, the new files that can be placed within those new blocks must be small enough to fit or else you cannot use that space. This leaves us with many unused blocks floating around the hard drive that we cannot use because they are not big enough to fit a new file size. In figure 2 we can avoid this since the blocks to not need to be in sequential order in the hard drive. For chained allocation all we need is one open block for the previous block to point to.

   b) **Contiguous allocation:** The main advantage with this method is that it is fast and easy. This is due to the fact that it does not take much computation to place files in one block after the other. The downside to using this method is the external fragmentation that is caused from deleting files. When the hard drive becomes full and there are many unused single blocks scattered around, we have to defragment the hard drive which can be a slow process.
   Chained allocation: The main advantage with this method is the fact that we no longer need to worry about external fragmentation since we are using pointers to locate the block containing the next part of the file. The disadvantages to this method are that we have to lose some space on each block for a pointer and this method does not work nearly as fast as contiguous allocation.

   c) Internal fragmentation occurs inside of an allocated block within the hard drive. This happens when a file is placed inside of a block, but it does not fill the entire space that is allocated.

2. The method being used in figure 3 is indexed allocation, where block 24 contains the index table that contains an ordered list of blocks where each part of a file is located. The method being used in figure 4 is indexed allocation with variable length partitions, where block 24 contains the index table that contains a list of start blocks and how many blocks follow after that start block.

3. A bit table keeps track of which blocks are allocated and which are unallocated. This is done by assigning each block a bit being either 0 (for unallocated) or 1 (for allocated). Since we will be continually checking the bit table to see if given blocks are free, it would be best to store it in the main memory for quicker access.

4. **Smaller** block size will have much less internal fragmentation since we will be able to fill each block much easier, but the performance will be much worse since we will have to manage so many more blocks.

Larger block size will be much better performance since we will have less blocks to deal with, but this will also have much more internal fragmentation since the blocks will be able to hold much larger files.

5. **FAT** stands for File Allocation Table and this is used to store file properties such as file names, file indexes, and file size.
FST stands for Free Space Table and this is used to store a starting free block and how many blocks after it are free.