Assign3 : Disk Virtualization (Coding Exercises)

3.1. **Disk Virtualization (Consolidation & Partitioning)**

- Create two arrays for storing Blocks of disks A and B.
- Block size is 100 bytes. Disk A has 200 blocks and Disk B has 300 Blocks
- Provide an API to a programmer so that it appear as one disk D of 500 blocks
- APIs could be Write / Read (block_No, block_inf) where block_no is 1..500
- Test it with random read / write of blocks

- Now create Disk of arbitrary size using an API CreateDisk(ID, No of blocks)
- Now this ID is used to read and write blocks.
- APIs could be Read/Write (ID, Block_No, Block_info,)
- It should return success if Block no within defined limit, else error
- Should be able to Create multiple Disk of arbitrary size, till no more space is left
- DeleteDisk(ID) would delete the disk and space is released , which can then be used in subsequent Create Disk.
- Test it using various creates/delete/read/write operations

- Arbitrary Create and Delete disk will create holes in the Disk array and you may have to use a different way to store blocks of array in case of fragmentation

- Document your Design and give a demo

3.2 **Disk Virtualization : Block Replication**

- This is an extension of 3.1. Modify the code for 3.1 to do Block Replication for providing reliable storage

- Write each block in two locations, so that in case one copy is in error, block can be read from the 2nd copy.
- The read operation should read from the first copy. In case it is in error, read from the 2nd copy.
- Before returning the value to the user, create additional copy in some other location (donot rewrite the error block. Flag it as bad block and never use it again).
- To simulate random read error, before reading a block generate a random number in the range 1-100. If the value is less than 10, assume reading the first copy has given an error.
- Test the system by doing large no of read and writes and see what is going on in the background where blocks are stored.
- Document your design and Give a demo

3.3. **Disk Virtualization : SnapShoting**

- This is an extension of 3.1. Modify the code for 3.1 to take a Snap Shot of the current version of the virtual Disk and roll back if needed.

- This feature lets the user create a checkpoint and then continue using the disk.
- Any number of checkpoints may be created. The user may return to any checkpoint at any time.
- API could be
    - Checkpoint (ID, No) No is the Checkpoint ID returned for later use.
    - Rollback (ID, Checkpoint_No) : Rolls back the virtual disk ID

- Test by doing multiple checkpoints and rolling back to anyone.
- Document your design and Give a Demo