1. (4 pt) Consider the Megatron 777 disk with the following characteristics:
10 surfaces with 10,000 tracks each; each sector is 512 bytes and each track on average has 1000 sectors; 20% of each track is used for gaps; the disk rotates at 10,000 rounds per minute; the time to move the head *k* tracks is 1 + 0.001*k* ms. Answer the following questions about the Megatron 777.

(1) Determine the capacity of each cylinder of the disk.
(2) If a block were 32 sectors, how long would it take to transfer two consecutive blocks?
(3) What is the average time to update two blocks of data on the same track? Assume the read/write head is on this track.
(4) Suppose the disk head is at track 2500 that is 1/4 of the way across the tracks. If the next request is for a block on a random track, calculate the average time to read this block.

Solution.

1. 10 tracks * 1,000 sectors/track * 512 bytes/sector = $5.12 * 10^6$ bytes

2. each sector: 360degrees * (80%) / 1,000 = 0.288 degree

   each gap: 360*(20%) / 1,000 = 0.072 degree

   2 blocks: 64 sectors + 63 gaps = 0.288*64+0.072*63=22.968 degrees

   (22.968/360)* (60/10,000) = $3.83 * 10^{-4}$ seconds

3. An update is a read followed by a write. Assume all operations are performed without interrupt and we need to read first both blocks then write them.

Blocks b1 and b2 are not consecutive. The approximate update time is

t(update) =

+t(half rotation) //time to wait for the beginning of b1

+t(rotation) //time to read b1, wait for the beginning of b2, read b2, and wait for the beginning of b1 in order to start writing (full circle).

+t(half rotation) //time to write b1, wait for the beginning of b2, write b2

= $3*10^{-3} + 6*10^{-3} + 3*10^{-3}$

4 Suppose the track is *i*. The probability that the next required track is on the left is (n-i)/n, and the probability that the it is on the right is i/n. If it is on the left, on average the head has to travel half way, and the same reasoning applies on it is on the right. Hence the average number of tracks to traverse is:

(i/2) * (i/n) + ((n-i)/2) * ((n-i)/n)

for i=2500 and n=10,000

$2500/2 * 2500/10000 + 7500/2 * 7500/10000 = 312.5 + 2812.5 = 3125$ tracks

The average seek time is: $1 + 3125 * 0.001 = 4.125$ms.
We add here the average rotational delay and the transfer time for a total of about 7.3 ms.

2. (3 pt) Suppose the relation R described in the slides grows to have as many tuples as can be sorted using 2PMMS on the machine described in the slides as well. Also assume that the disk grows to accommodate R, but all other characteristics of the disk, machine, and relation R remain the same. How long would it take to sort R?

Solution.

With the following parameters:

M=100MB = 100,000,000 Bytes = 10^8 Bytes ($100*2^{20}$ more precisely)
B = 16,384 Bytes
R = 160 Bytes

we can sort
M^2/RB = (100*2^20)^2 / (160 * 16,384) ≅ 4.2 billion tuples.

In terms of number of blocks, this is ((100*2^20)^2 / (160 * 16,384)) / (16,384/160) = 40,960,000 blocks.

If 11 ms is the time for one read or write, 2PMMS will take
4 * 40,960,000 * 11 / (60*60) ≅ 500 hours

Of course, this is a pessimistic estimate assuming that the disk head will move arbitrarily after reading each block.

3. (3 pt) Describe the steps taken to recover from the following failures using RAID level 6 scheme with seven disks:

a) Disks 1 and 7.
b) Disks 1 and 4.
c) Disks 3 and 6.

Solution.

(a) In the matrix of the figure on the slides, we see that columns 1 and 7 differ in both the first and second rows. In each case, column 1 has bit 1 and column 7 has bit 0. Let's use the first row to recover disk 1. That step requires us to take the modulo-2 sum of disks 2, 3, and 5, the other disks where row 1 has bit 1.
Next, we use row 3, the only row where column 7 has a 1, to recover disk 7 by taking the modulo-2 sum of disks 1, 3, and 4.

(b) and (c) are similar.

Also, solutions with a different matrix are acceptable.

4. (3 pt) Suppose we have eight data disks numbered 1 through 8, and three redundant disks: 9. 10, and 11. Disk 9 is a parity check on disks 1 through 4, and disk 10 is a parity check on disks 5 through 8. If all pairs of disks are equally likely to fail simultaneously, and we want to maximize the probability that we can recover from the simultaneous failure of two disks, then on which disks should disk 11 be a parity check?

The parity checks of disks 9 and 10 can be represented by the matrix

1	2	3	4	5	6	7	8	9	10	11
1	1	1	1	0	0	0	0	1	0	0
0	0	0	0	1	1	1	1	0	1	0
?	?	?	?	?	?	?	?	0	0	1

Before we add a third row representing the parity check of disk 11, recall that **we can recover from two disk failures if and only if the two disks have different columns.**
We want to maximize the number of pairs of disks that no longer have identical columns.
As is now, 1,2,3,4 have identical columns, and also 5,6,7,8 have identical columns. Where to put 1 for "?" and where 0, so that we get the greatest number of non-identical columns?
A solution would be to make disk 11 be a parity check on disks 3 through 6, but there are many other correct answers.