CSCE 310: Database Systems

Homework 7

Raymond Zhu

923008555

Problem 1:

a)

Capacity of a disk = # surface \* # track \* # sector \* sector size

Capacity of a disk = 10 \* 100,000 \* 1000 \* 1024

Capacity of a disk = 1,000,000,000 \* 1024

Capacity of a disk = 109 \* 1024 bytes

c)

Maximum seek time = 1 + 0.0002\*n where n = # track

Maximum seek time = 1 + 20 = 21 milliseconds

d)

Maximum rotational latency = time (seconds) / rotations per minute

Maximum rotational latency = 60 seconds / 10,000 rpm

Maximum rotational latency = 0.006 seconds

e)

Given a block size of 65,546 bytes (64 sectors)

Transfer time = # sectors for size of block \* (rotational latency / # sectors)

Transfer time = 64 \* (0.006 / 1000)

Transfer time = 64 \* 6.0 x 10-6 seconds

f)

Average seek time = Maximum seek time / 3

Average seek time = 21 / 3

Average seek time = 7 milliseconds

g)

Average rotational latency = half of maximum rotational latency

Average rotational latency = 0.006 / 2

Average rotational latency = 0.003 seconds

Problem 2:

If we are using a Megatron 747 disk with an average seek time of 6.46, rotational latency of 4.17, and transfer time of 0.13, we can calculate completion times given the sum of rotational latency, transfer time, and the incurred seek time.

a)

Initially reading down to 8000 from 32000, and then by that time 4000 has arrived so we seek 4000. 40000 hasn’t arrived yet so we seek to 48000. Lastly, we then seek to 40000

|  |  |
| --- | --- |
| Cylinder of Request | Time Completed (ms) |
| 8000 | 11.3 |
| 4000 | 17.6 |
| 48000 | 33.9 |
| 40000 | 41.2 |

b)

|  |  |
| --- | --- |
| Cylinder of Request | Time Completed (ms) |
| 8000 | 11.3 |
| 48000 | 26.6 |
| 4000 | 39.9 |
| 40000 | 54.2 |

Problem 3:

a) 00111011 – 5 ones

Even parity = 001110111 – adding 1 to make the # of 1-bits even

Odd parity = 001110110 – adding 0 to keep the # of 1-bits odd

b) 0000000 – 0 ones

Even parity = 00000000 – adding 0 to keep the # of 1-bits even

Odd parity = 00000001 – adding 1 to make the # of 1-bits odd

c) 10101101 – 5 ones

Even parity = 101011011 – adding 1 to make the # of 1-bits even

Odd parity = 101011010 – adding 0 to keep the # of 1-bits odd

Problem 4:

a) Given 01010110 11000000 00111011, and 11111011

Take the #1-bits (mod 2) where even = 0 and odd = 1

1st bit = [0101] = 0

2nd bit = [1101] = 1

3rd bit = [0011] = 0

4th bit = [1011] = 1

5th bit = [0011] = 0

6th bit = [1000] = 1

7th bit = [1011] = 1

8th bit = [0011] = 0

The recovered block is: 01010110

b) Given 11110000 11111000, 00111111, and 00000001

1st bit = [1100] = 0

2nd bit = [1100] = 0

3rd bit = [1110] = 1

4th bit = [1110] = 1

5th bit = [0110] = 0

6th bit = [0010] = 1

7th bit = [0010] = 1

8th bit = [0011] = 0

The recovered block is: 00110110

Problem 5:

a) Fields can start at any byte

# bytes = 15 (character string of length 15) + 2 (size of integer) + 3 (size of date) + 5 (size of time)

# bytes = 15 + 2 + 3 + 5

# bytes = 25 bytes

b) Fields must start at a byte that is a multiple of 4. Need to pad spacing to values that aren’t multiples of four

# bytes = [15 + [1] (16 mod 4)] + [2 + [2] (4 mod 4)] + [3 + [1] (4 mod 4)] + [5 (last field)]

# bytes = 16 + 4 + 4 + 5

# bytes = 29 bytes

c) Fields must start at a byte that is a multiple of 8. Need to pad spacing for values that aren’t multiples of eight

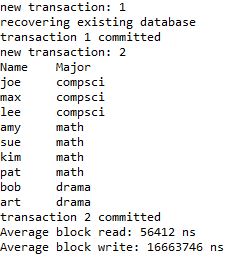
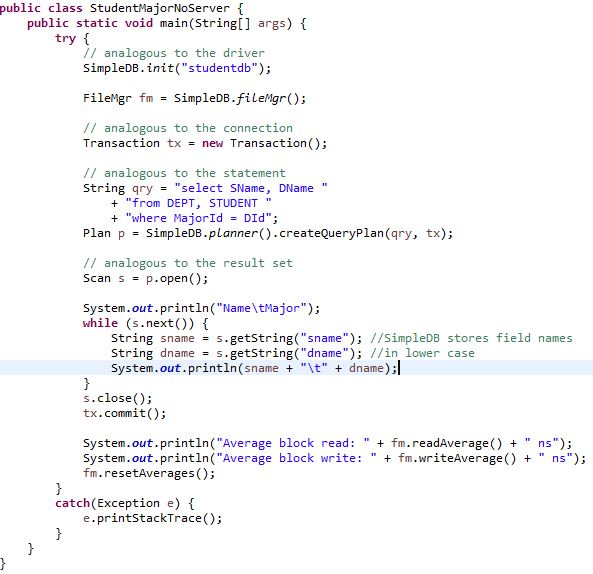
# bytes = [15 + [1] (16 mod 8)] + [2 + [6] (8 mod 8)] + [3 + [5] (8 mod 8)] + [5 (last field)]

# bytes = 16 + 8 + 8 + 5

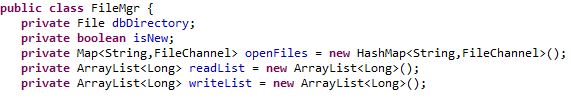
# bytes = 37 bytes

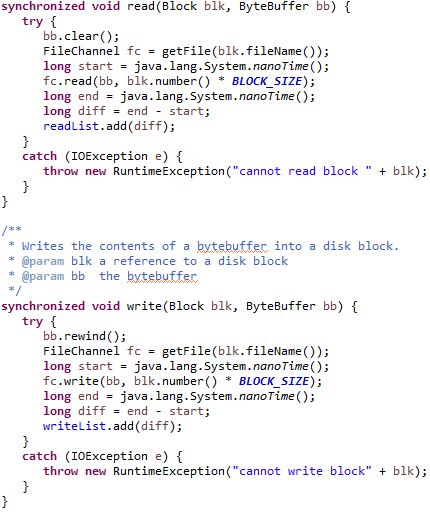
Problem 6:

StudentMajorNoServer

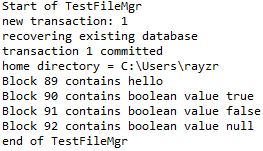


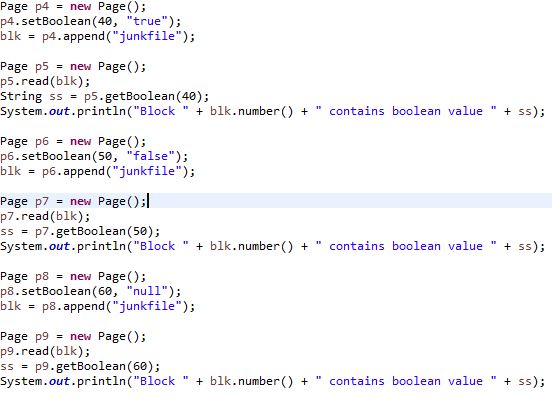
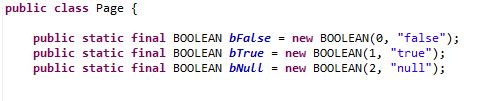
FileMgr



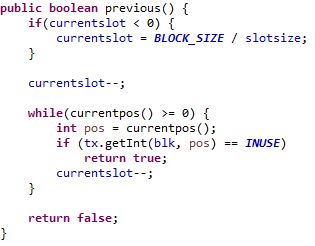
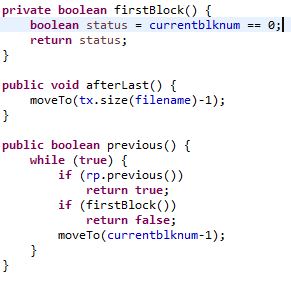


Problem 7:





Problem 8:



An aggie does not lie, cheat, steal, or tolerate those who do.