I.

1.
(8ms * .3 + 20ms * .7 + 100ns)*p + (1-p)100ns = 200ns.
Solve for p.
Page fault rate has to be 6.09756√ó10^-6 %.

2.
a.

LRU

Ref	:	7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1
Page 1:		7	7	7	1		1	3	3	3	7		7	7	5	5	5	2	2	2	1
Page 2:			2	2	2		2	2	4	4	4		1	1	1	4	4	4	3	3	3
Page 3:				3	3		5	5	5	6	6		6	0	0	0	6	6	6	0	0

Total Faults 18

FIFO

Ref	:	7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1
Page 1:		7	7	7	1		1		1	6	6		6	0	0	0	6	6	6	0	0
Page 2:			2	2	2		5		5	5	7		7	7	5	5	5	2	2	2	1
Page 3:				3	3		3		4	4	4		1	1	1	4	4	4	3	3	3

Total Faults 17

Optimal

FIFO

Ref	:	7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1
Page 1:		7	7	7	1		1		1	1	1			1		1	1	1	1		
Page 2:			2	2	2		5		5	5	5			5		4	6	2	3		
Page 3:				3	3		3		4	6	7			0		0	0	0	0		

Total Faults 13

3.
    a. Yes you could recreate the list by brute force. You would have to look at all saved files and where they are stored, and use that data to derive were the free blocks are.

    b. 7 I/O Operations. Two for each directory (current, a, then b) and then one for the file its self (c).

    c. Log structure where all operations are first written to a log before they are performed and remove after they complete. If a system failure happens, the log can re-create what should happen.

#II.

1.
```
  time ./a.out
  real      0m0.165s
```

```
  user      0m0.126s
  sys 0m0.012s

  time ./a.out > X
  real      0m0.100s
  user      0m0.094s
  sys 0m0.003s
```

  Writing to the disk took less time. But I did get some inconsistent results.

2.

```
  time ./a.out
  real      0m0.157s
  user      0m0.123s
  sys 0m0.011s

  time ./a.out > Y
  real      0m0.098s
  user      0m0.091s
  sys 0m0.004s
```

  Printf and write are really close. Once again writting to disk took less time.

3. Writting the commands back to the console through the system to be the difference between printing and writing to disk. Writting to disk seemed to be faster because it had to deal with the system less. On this machine the difference the system call and the lib call were very small.