1.（1）采用 Bitmap 位图管理法，用每个 bit 表示一个盘块的使用情况（0 表示空，1 表示占用）。16384 个磁盘块一共需要 16384 / 8 = 2048 个 Byte = 2KB，恰好可以放在 2KB 的内存空间里。
（2）采用 CSCAN 算法，依次访问磁道 120，30，50，90。
所以访问中移动的磁道总数= 20 + (120-30) + 20 + 40 = 170, 所以总的寻道时间是 170*1ms = 170ms。磁面转速 6000RPM，故平均旋转延迟 = 60*1000ms/（6000*2）= 5ms，共访问四个磁道，所以总的旋转延迟= 4*5ms = 20ms。每个磁道 100 个扇区，所以读取一个扇区需要（60*1000ms/6000）/100 = 0.1ms，总的传输时间 = 4*0.1ms = 0.4ms
所以总时间=170+20+0.4 = 190.4ms
（3）用 FCFS 调度策略，因为 Flash 的结构不用考虑寻道时间和旋转延迟，可以直接按 IO 请求的先后顺序服务。

2.（1）写放大为 22 个 Page
（2）共发生 6 次擦除操作
（3）发生 3 次 full merge
（4）Log Table：2000->0
Block Table: 1000->8, 2000->4

0	1	2	3		4	5	6	7		8	9	10	11		12	13	14	15
e					e	f	g	h		a	b	c	d					
Block 0					Block 1					Block 2					Block 3			

3.
fd2=4, child: c=b
fd2=4, c=a
fd2=4, c=r
或
fd2=4, c=b
fd2=4, child: c=a
fd2=4, c=r

4. 固定大小 data block：文件对应 5MB/4KB = 1280 个 data block。由于 1024+12<1280，所以需要使用 12 个 direct pointers 和 2 个 indirect point，共 1280+2=1282 个指针，对应的存储空间=1282*8 = 10256 bytes
extent: 文件分为 5 个连续的部分, 每个连续的部分需要一个指针和 extent size, 对应 12 bytes, 所以存储空间= 5*12 = 60 bytes

5. （1）在 bar 文件内，首先 superblock 要读一次，inode bitmap 和 data bitmap 分别读写一次，inode 要读写，读取 data block 时，100KB 对应 25 个 page，需要读 25 个指针。每个指针追加写入 1MB 数据，共 25MB。其中有 8MB-52KB 的数据由 indirect point 指向，写入时需要 2 次 IO；剩余 17MB+52KB 数据则由 double indirect pointer 指向，写入时需要 3 次 IO。
而在访问 bar 文件时，要读一次根目录的 inode，读一次根目录的 data block，读写 foo 目录的 inode，读写 foo 目录的 data block。

综上，共需要 1+2+2+2+25+8*2+18*3+2+2*2 = 108 次

（2）该命令的第二个参数是文件，会实现重命名的功能，所以读写根目录的 inode，读写根目录的 data block，读写 foo 目录的 inode，读写 foo 目录的 data block（因为文件的名字存储在它的父目录中，所以需要修改的是 foo 的 data block 而不是 bar 文件）