Homework 3

1.

磁盘块的大小8KB = 8192KB，表头有8B的模式指针，剩下的磁盘块的大小为8184B

由于插入每个定长记录还需要在偏移表当中增加2B，因此每个记录需要202B

由于删除的时候会用删除标记代替，因此除了第一天，每天会浪费4B的作为删除标记则需要的天数为

$$\lfloor (8184 - 808 + 400)/(400 + 4) \rfloor + 1 = 20$$

则用完需要20天

2.

(1)

- 方法一：
  - 使用一个空闲链表：使用一个空闲链表存储当前所有的空闲的frame，每当需要返回一个空闲的frame的时候直接从空闲链表当中获取
  - 时间负责度：每次从链表的表头获取空闲链表，时间复杂度为O(1)
- 方法二：
  - 维护一个bit map，一个位图来表示是否为空，如果bit为1,则表示frame不为空，否则表示frame为空
  - 时间负责度： 由于每次需要便利位图，因此时间复杂度为O(n)

(2)

- 方法一：
  - 使用哈希表：利用每个页的page_id作为标识，建立一个哈希表，每当收到一个页请求的时候，去查找这个页面是否存在。如果存在直接返回对应的frame地址，如果不存在则需要从磁盘加载对应的页
  - 时间复杂度：O(1)，由于在平均情况下哈希表的时间复杂度为O(1)
- 方法二：
  - 使用LRU缓存：利用 LRU 策略将缓存页面按使用时间顺序维护。可以通过一个双向链表和一个哈希表来联合使用。哈希表记录页号到链表节点的映射，而链表则记录页面的访问顺序。每次访问页面时，通过哈希表查找，找到页面对应的链表节点，移动该节点到链表头部。
  - 时间复杂度： O(1)

3.

(1) 可能导致大量的清理操作（垃圾回收）

- **解释**：CF-LRU 算法优先选择清洁页面进行置换，意味着它会把脏页面留在内存中，直到有足够的清洁页面可供替换。这会导致脏页面在内存中停留时间过长，最终可能需要通过较为复杂的清理操作将这些脏页面写回 SSD。随着时间的推移，系统可能需要频繁地执行垃圾回收（GC），即将大量的脏页面写回 SSD，尤其在内

存使用较为紧张时。这样一来，虽然算法减少了写入的频率，但在某些情况下却可能导致突发的、高强度的写操作，尤其是在内存空间有限或压力较大的情况下。

- **影响**：这种突发性的写操作可能会导致性能的严重波动，并且与连续的、平稳的写入相比，反而会增加 SSD 的写入压力，从而影响性能和寿命。

(2)忽略了页面的热度（访问频率）

- **解释**：CF-LRU 算法通过优先选择清洁页面来减少写操作，但它没有考虑页面的访问热度（频繁被访问的页面与不常被访问的页面）。如果一个页面在内存中经常被访问，但却是脏页面，CF-LRU 算法可能会拖延将该脏页面写回 SSD 的时机，甚至可能因为算法偏向清洁页面的置换而导致频繁的页面替换。换句话说，它忽略了“热”页面的存在，从而可能导致一些热点数据被频繁交换，增加了额外的系统开销。
- **影响**：忽略页面访问频率会导致缓存效率低下，尤其是对于那些热点页面。频繁访问的页面可能被错误地替换出去，增加了不必要的读写操作，从而降低了系统的整体性能。