参数	值
字节偏移	
缓存索引	
缓存标记	
缓存命中？（是/否）	
返回的缓存字节	

** 9.14 假设有一个输入文件 hello.txt，由字符串“Hello, world!\n”组成，编写一个 C 程序，使用 mmap 将 hello.txt 的内容改变为“Jello,world!\n”。

* 9.15 确定下面的 malloc 请求序列得到的块大小和头部值。假设：1)分配器保持双字对齐，使用隐式空闲链表，以及图 9-35 中的块格式。2)块大小向上舍入为最接近的 8 字节的倍数。

请求	块大小（十进制字节）	块头部（十六进制）
malloc(3)		
malloc(11)		
malloc(20)		
malloc(21)		

* 9.16 确定下面对齐要求和块格式的每个组合的最小块大小。假设：显式空闲链表、每个空闲块中有四字节的 pred 和 succ 指针、不允许有效载荷的大小为零，并且头部和脚部存放在一个四字节的字中。

对齐要求	已分配块	空闲块	最小块大小（字节）
单字	头部和脚部	头部和脚部	
单字	头部，但是没有脚部	头部和脚部	
双字	头部和脚部	头部和脚部	
双字	头部，但是没有脚部	头部和脚部	

** 9.17 开发 9.9.12 节中的分配器的一个版本，执行下一次适配搜索，而不是首次适配搜索。

** 9.18 9.9.12 节中的分配器要求每个块既有头部也有脚部，以实现常数时间的合并。修改分配器，使得空闲块需要头部和脚部，而已分配块只需要头部。

* 9.19 下面给出了三组关于内存管理和垃圾收集的陈述。在每一组中，只有一句陈述是正确的。你的任务就是判断哪一句是正确的。

1) a) 在一个伙伴系统中，最高可达 50%的空间可以因为内部碎片而被浪费了。
   b) 首次适配内存分配算法比最佳适配算法要慢一些(平均而言)。
   c) 只有当空闲链表按照内存地址递增排序时，使用边界标记来回收才会快速。
   d) 伙伴系统只会有内部碎片，而不会有外部碎片。

2) a) 在按照块大小递减顺序排序的空闲链表上，使用首次适配算法会导致分配性能很低，但是可以避免外部碎片。
   b) 对于最佳适配方法，空闲块链表应该按照内存地址的递增顺序排序。
   c) 最佳适配方法选择与请求段匹配的最大的空闲块。
   d) 在按照块大小递增的顺序排序的空闲链表上，使用首次适配算法与使用最佳适配算法等价。

3) Mark&Sweep 垃圾收集器在下列哪种情况下叫做保守的：
   a) 它们只有在内存请求不能被满足时才合并被释放的内存。
   b) 它们把一切看起来像指针的东西都当做指针。
   c) 它们只在内存用尽时，才执行垃圾收集。
   d) 它们不释放形成循环链表的内存块。