**PartA**

PartA要求模拟Cache的访问，采用LRU原则进行替换，整体来说较为简单，只需要理解题目要求就可以了。输入会指定组索引位s、偏移块位b、每组行数E以及tarce文件。利用getopt函数完成对输入的处理，根据指定的s、b和E初始化cache，cache的本质是cache\_line的二维数组，需要用malloc根据s和b动态创建。cache\_line的结构为一个有效位、一个标记位和一个时间戳(记录最后一次使用的时间)。

trace文件中我们需要注意的只有L、S、M开头的指令，其中L和S需要访问一次内存，M需要访问两次内存。字母之后有对应需要访问的地址以及数据大小，因为只需要计算命中次数、未命中次数和替换次数，因此数据大小同样可以忽略。

cache查询逻辑很简单，给一个地址，拆分出组索引和标记位，首先在指定组中找是否存在标记为相同的行，如果存在则命中；如果不存在则未命中，并寻找空行写入记录；如果没有找到空行，则找最久未用的记录(时间戳最小的记录)替换，替换次数加一。

需要注意的是标记位的初始值应该设为负值，这样在寻找标记为相同的行时就不需要考虑有效位了，标记位为负的即表示无效，而负的标记位也不可能和任何地址的标记位相同。