**请求分页分配方式**

**沈忆珠 103013**

**要求：**

* 基本任务
* 假设每个页面可存放10条指令，分配给一个作业的内存块为4。模拟一个作业的执行过程，该作业有320条指令，即它的地址空间为32页，目前所有页还没有调入内存。
* 模拟过程
  + 在模拟过程中，如果所访问指令在内存中，则显示其物理地址，并转到下一条指令；如果没有在内存中，则发生缺页，此时需要记录缺页次数，并将其调入内存。如果4个内存块中已装入作业，则需进行页面置换。
  + 所有320条指令执行完成后，计算并显示作业执行过程中发生的缺页率。
  + 置换算法可以选用FIFO或者LRU算法
  + 作业中指令访问次序可以按照下面原则形成：

50%的指令是顺序执行的，25%是均匀分布在前地址部分，25％是均匀分布在后地址部分。

* 具体实施方法如下
  + 在0－319条指令之间，随机选取一个起始执行指令，如序号为m
  + 顺序执行下一条指令，即序号为m+1的指令
  + 通过随机数，跳转到前地址部分0－m-1中的某个指令处，其序号为m1
  + 顺序执行下一条指令，即序号为m1+1的指令
  + 通过随机数，跳转到后地址部分m1+2~319中的某条指令处，其序号为m2
  + 顺序执行下一条指令，即m2+1处的指令。
  + 重复跳转到前地址部分、顺序执行、跳转到后地址部分、顺序执行的过程，直到执行完320条指令。

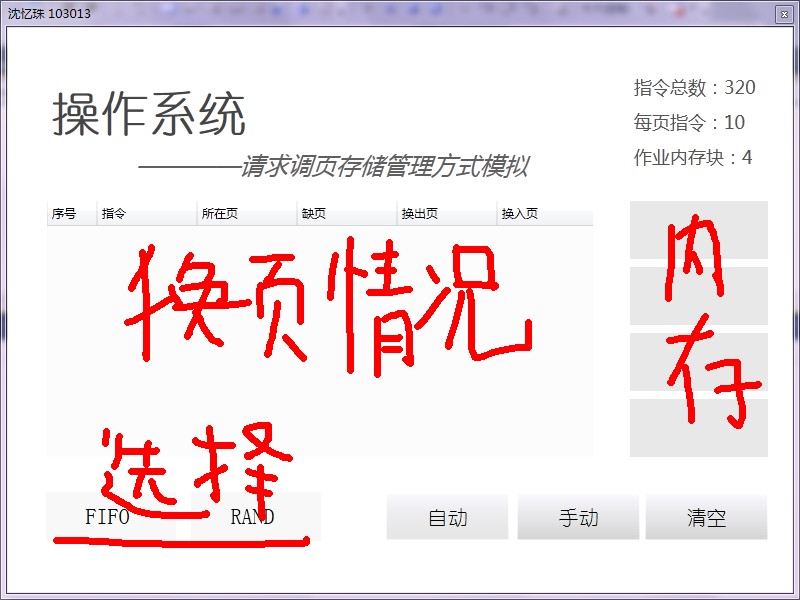
**目的：**

* + 页面、页表、地址转换
  + 页面置换过程
  + 加深对请求调页系统的原理和实现过程的理解。

**设计：**

* 先进先出法（First In First Out）：
* 该算法总是淘汰最先进入内存的页面，既选择在内存中驻留时间最久的页面予以淘汰。
* 在该算法的模拟过程中，设立一个范围在0-3的变量fifo，用来访问相对应的内存，每当页面被置换进入内存时，将fifo自动加1，更好地模拟了先进先出法；
* 最近最久未使用（Least Recently Used）：
* 该算法以最近的过去作为不久将来的近似， 将过去最长一段时间里不曾被使用的页面置换掉。
* 在该算法的模拟过程中，给每个内存块设立一个access，当改内存块开始被访问时，每一次访问内存access都将自动加1，且将被访问的内存相对应的access值设为0。通过判定access值的大小来决定该访问的内存块。

运行：



**总结：**

通过本次操作系统实验，使我们对操作系统这门课程有了更进一步的认识和了解，通过模拟实现请求页式存储管理的几种基本页面置换算法，了解虚拟储技术的特点。通过对页面、页表、地址转换和页面置换过程的模拟，加深对请求调页系统的原理和实现过程的理解。

本项目的难点之一在于如何用c#语言按要求模拟生成随机指令，即50%的指令是顺序执行的，25%的指令是均匀分布在前地址部分，25%的指令是均匀分布在后地址部分，花费了大量时间研究该算法，最终将这个过程拆分为4步，通过一个随机函数并限制数的大小予以实现。对于随机产生的数字可能出现重复的问题，则设了一个320大小的数组对应1-320个数，每抽取一条指令则删除这个内容。

在本次设计过程中，用到了许多C#的基本知识和操作系统的基本原理，是对平时所学知识的一次考验，尽管这些知识都学过，但运用到实际时，却不知从何下手，而且错误不断，往往为了找一个错误而花了大量的时间，这是专业知识掌握不够，缺乏实践动手能力的表现。在设计的过程中我们发现了许多自己的不足之处，对以前所学过的知识理解得不够深刻，掌握得不够牢固，以后还要多加努力。

总之，通过该实验，使我学到很多知识以及认识到实践的重要性。也使我了解到编写程序不是首要任务，而是一种实现手段。我们最重要的是如何做好需求分析和理清思路，做出正确、简洁的流程设计，这样可以达到事半功倍的效果。