



**操作系统课程设计实验报告**

——实验三：虚存管理

负责人姓名：姜铭

学号：14061167

日期：2016.5.7

**小组成员**

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 姓名 | 学号 | 实验分工 |
| 1 | 于东方 | 14061149 | 实验四 |
| 2 | 牛钰浩 | 14061155 | 实验二 |
| 3 | 常涛 | 14061161 | 实验一 |
| 4 | 姜铭 | 14061167 | 实验三 |

目录

[1 实验目的 5](#_Toc451023347)

[2 需求说明 5](#_Toc451023348)

[2.1 基本要求 5](#_Toc451023351)

[2.2 提高要求 5](#_Toc451023352)

[2.3 完成情况 6](#_Toc451023353)

[3 设计说明 7](#_Toc451023354)

[3.1 流程图 7](#_Toc451023358)

[3.2 基本要求实现说明 8](#_Toc451023359)

[3.1.1 模拟存储 8](#_Toc451023364)

[3.1.2 一级页表 8](#_Toc451023365)

[3.1.3 产生访存请求 8](#_Toc451023366)

[3.1.4 响应请求 9](#_Toc451023367)

[3.2 提高要求实现说明 10](#_Toc451023368)

[3.2.1 实现二级页表 10](#_Toc451023369)

[3.2.2 多道程序的存取控制 10](#_Toc451023370)

[3.2.3 通过FIFO进行进程间通信 12](#_Toc451023371)

[3.2.4 页面老化算法 13](#_Toc451023372)

[3.2.5 LRU算法 14](#_Toc451023373)

[3.2.6 实现辅存和实存内容的打印 15](#_Toc451023374)

[4 问题及解决方法 15](#_Toc451023375)

[4.1. 使用指针遇到的问题 15](#_Toc451023380)

[4.2. 源码中存在的问题 16](#_Toc451023381)

[4.3. 一些细节带来的问题 16](#_Toc451023382)

[5 收获和感想 16](#_Toc451023383)

1. 实验目的
   1. 了解Linux的内存管理机制。
   2. 掌握页式虚拟存储技术，理解虚地址到实地址的定位过程。
   3. 掌握“最不频繁使用淘汰算法”，即LFU页面淘汰算法。
2. 需求说明
4. 1. 基本要求

本实验要求在了解Linux系统下页式存储管理机制的基础下，实现一个简单的虚存管理模拟程序。

具体要求如下：

* 设计并实现一个虚存管理模拟程序，模拟一个单道程序的页式存储管理，用一个一维数组模拟实存空间，用一个文本文件模拟辅存空间。
* 建立一个一级页表
* 程序中使用一个函数do\_request()随机产生访存请求，访存操作包括读取、写入、执行三种类型
* 实现一个函数do\_response()响应访存请求，完成虚地址到实地址的定位及读/写/执行操作，同时判断并处理缺页中断
* 实现LFU页面淘汰算法
  1. 提高要求

在基础要求完成的基础上对程序的功能和性能进行改进，具体要求如下：

* 建立一个多级页表
* 实现多道程序的存储控制
* 将do\_request()和do\_response()实现在不同进程中，通过进程间通信（如FIFO）完成访存控制的模拟
* 实现其它页面淘汰算法：如页面老化算法、最近最久未使用淘汰算法（LRU）、最优算法（OPT）等
  1. 完成情况

所有基本要求已经完成，提高要求的完成情况如下：

* 建立了一个的二级页表。
* 通过修改请求的数据结构和do\_response()等函数，实现了多道程序的存取控制。
* 通过FIFO完成进程间通信，将do\_request()和do\_response()实现在不同进程中。
* 实现了页面老化算法和LRU算法。

程序运行截图如下：

1. 设计说明

4. 1. 流程图

结束

开始

更新页表

装入页面

产生缺页中断

获取块号，计算实地址

写回至辅存

产生访存请求，写入管道

获取页表

根据页号和页内偏移查页表

更新页表

返回访存结果

返回错误信息

淘汰页面

环境初始化

地址越界

页在实存

有权访问

页有修改

有空闲块

程序结束

是

是

是

是

是

是

否

否

否

否

否

否

* 1. 基本要求实现说明

源程序基本实现了所有基本要求，说明如下:

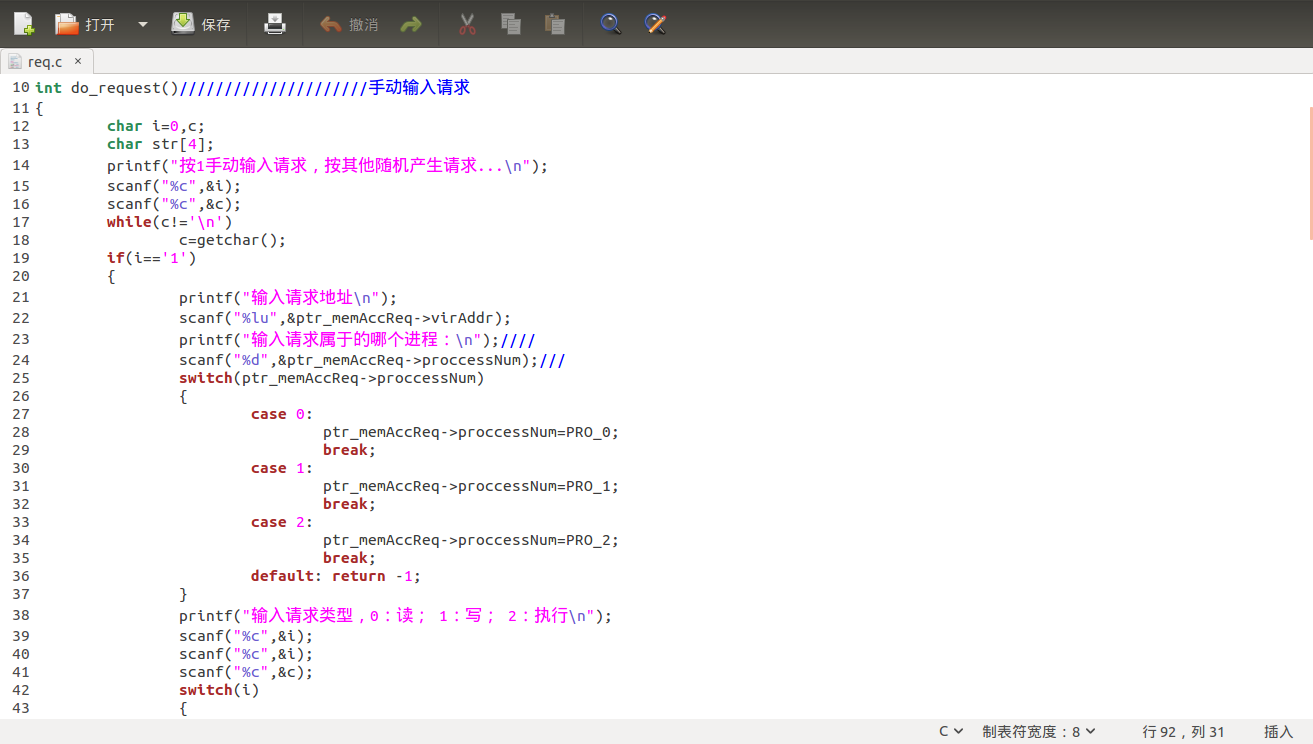
3. 1. 1. 模拟存储

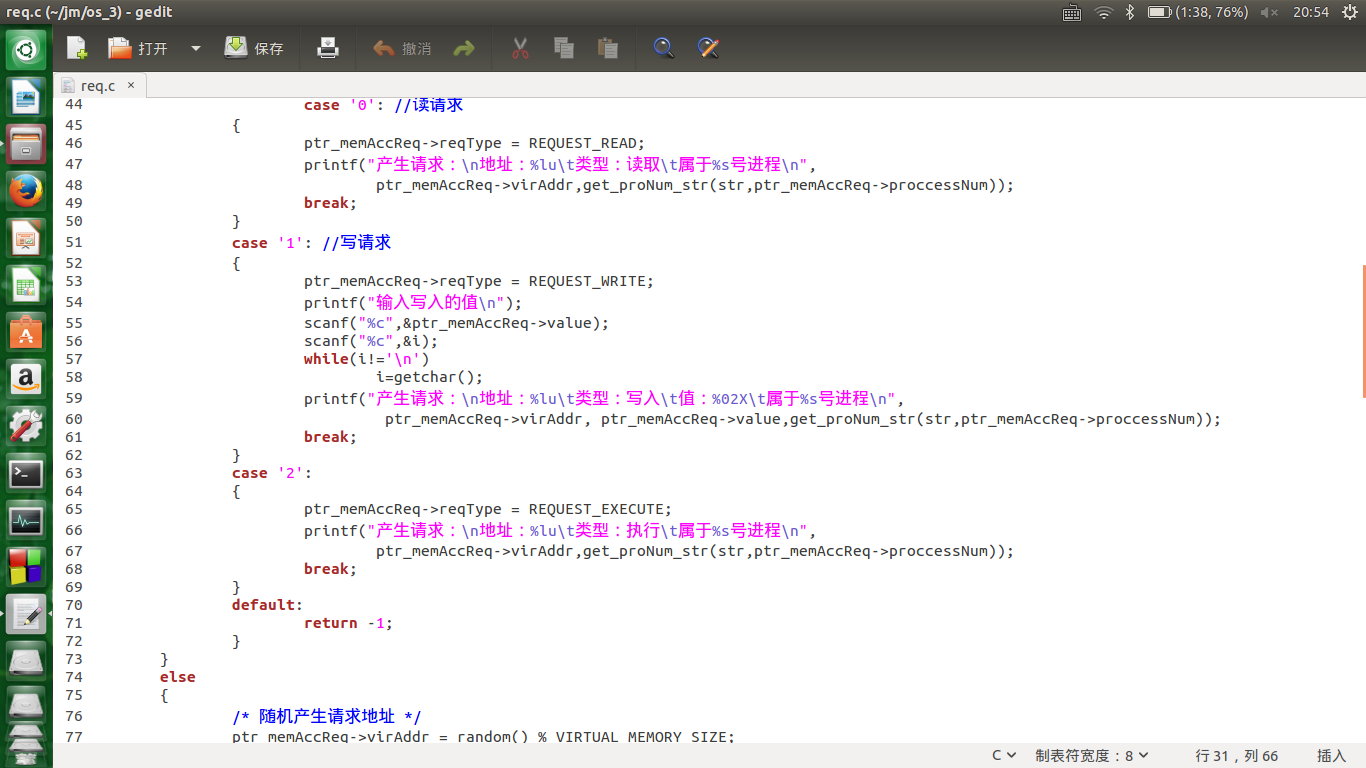
实验用一维数组模拟实存空间，用文本文件模拟辅存空间。通过文件操作实现实存与辅存中的数据读取与写入。

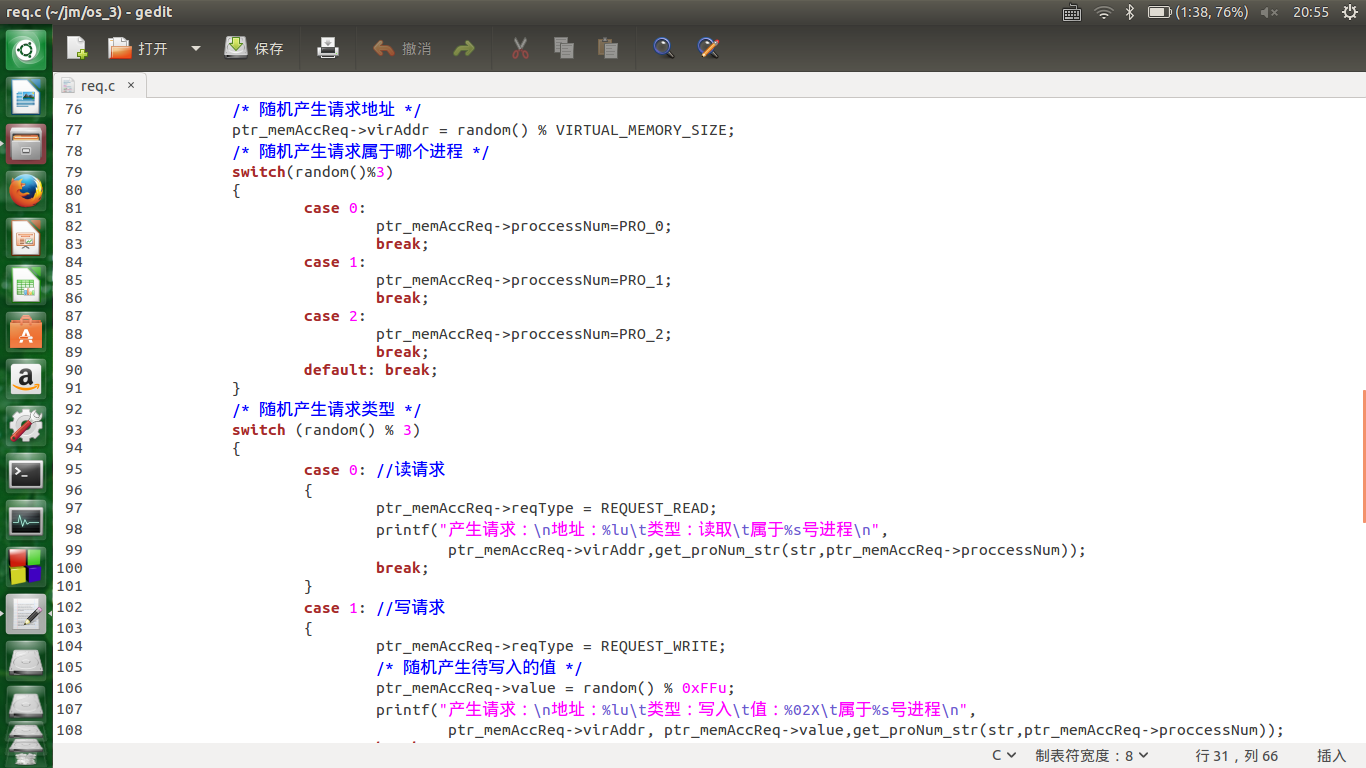
* + 1. 一级页表

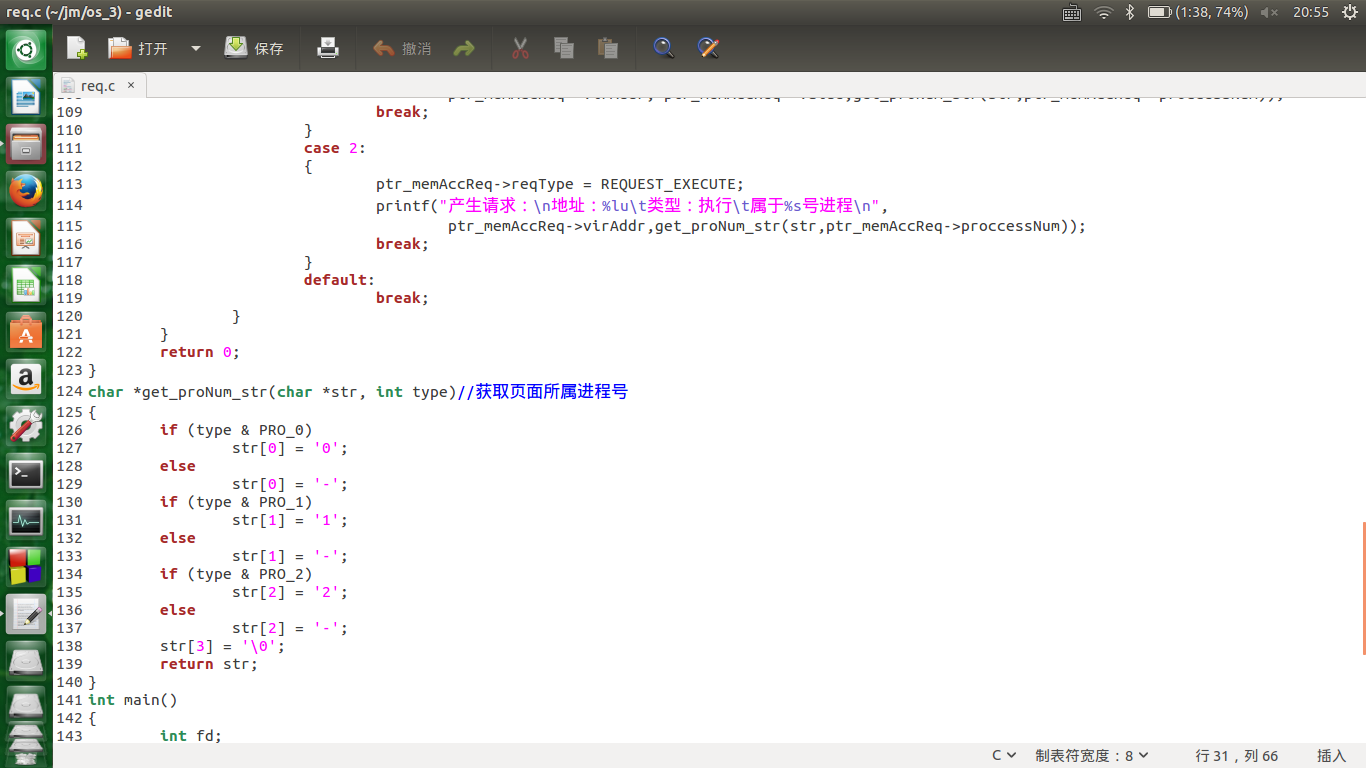
实验中采用一维数组作为页表项，计算页号后通过数组下标查找页表。

* + 1. 产生访存请求

源程序通过随机数随机产生请求，我在此基础上实现了手动输入请求。代码如下：







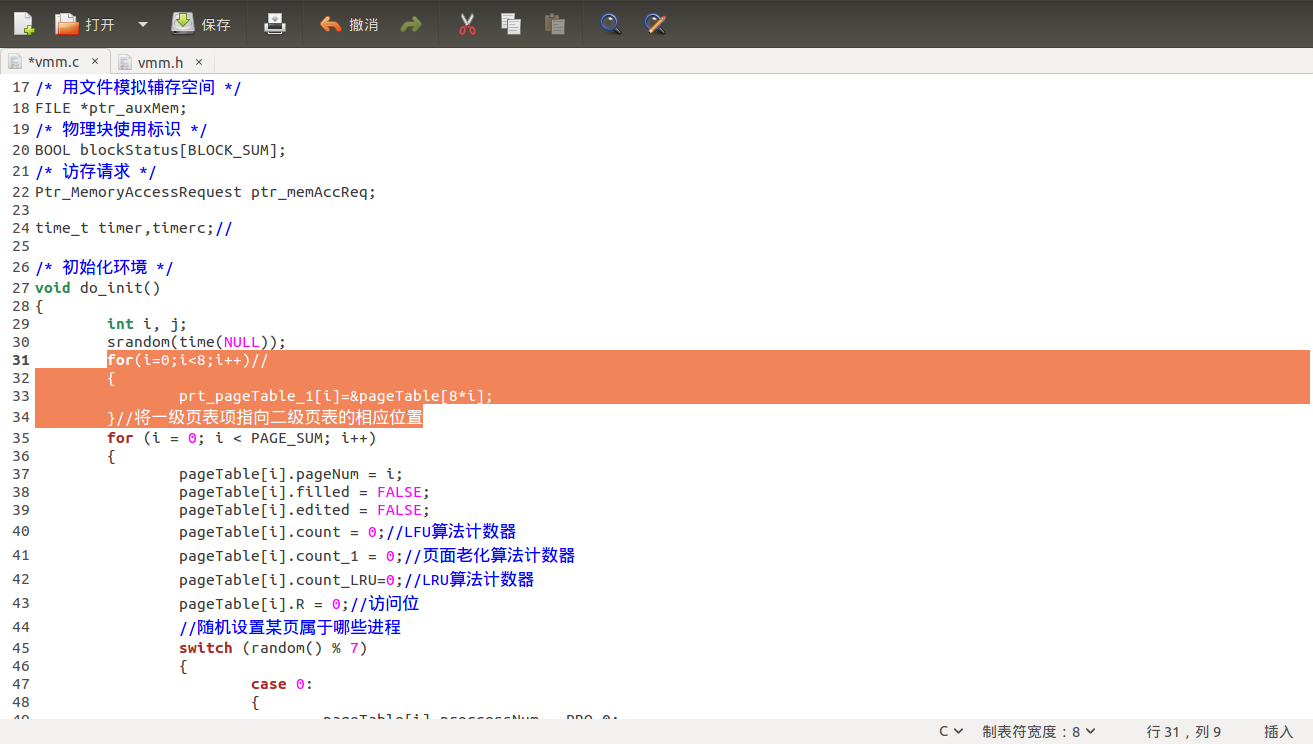
* + 1. 响应请求

先根据请求的地址计算页号和页内偏移值，获取对应页表项。再看该页面是否已装入内存，若没有装入则产生缺页中断，将页面调入内存。然后检查页面的访问权限，判断该请求是否可以访问。若不能访问则输出错误信息，否则响应请求。

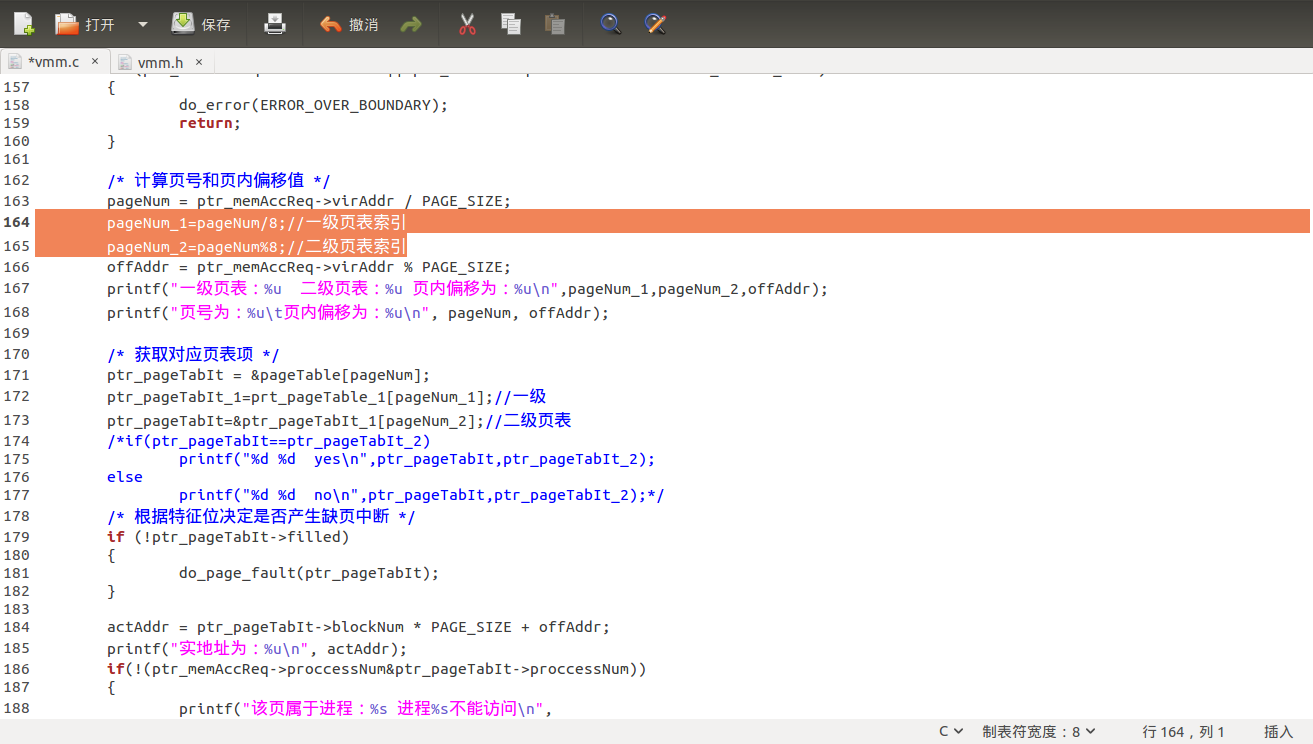
* 1. 提高要求实现说明
     1. 实现二级页表

实现思路：本实验中将原来的64个二级页表分为8份，用8个一级页表指向相应地址。在处理请求时分别计算一级索引和二级页表索引。实现代码如下。

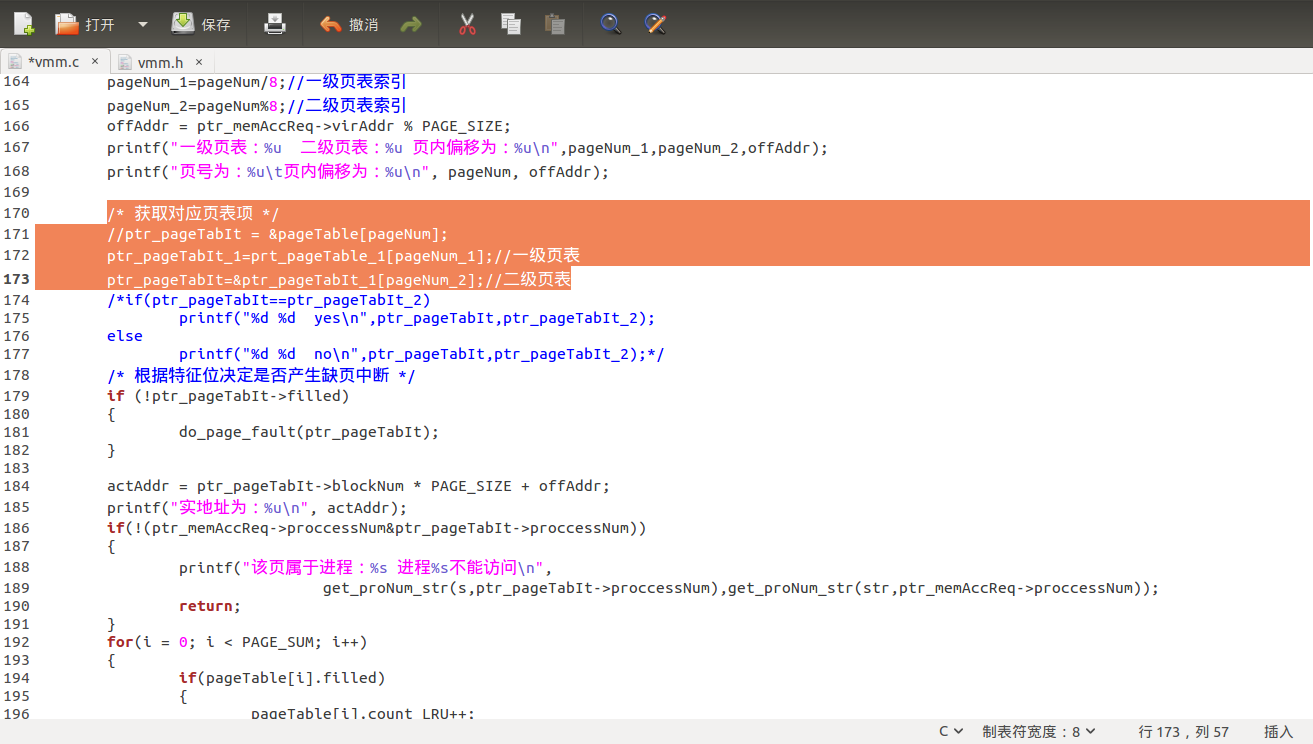
初始化：



计算一级页表和二级页表索引：



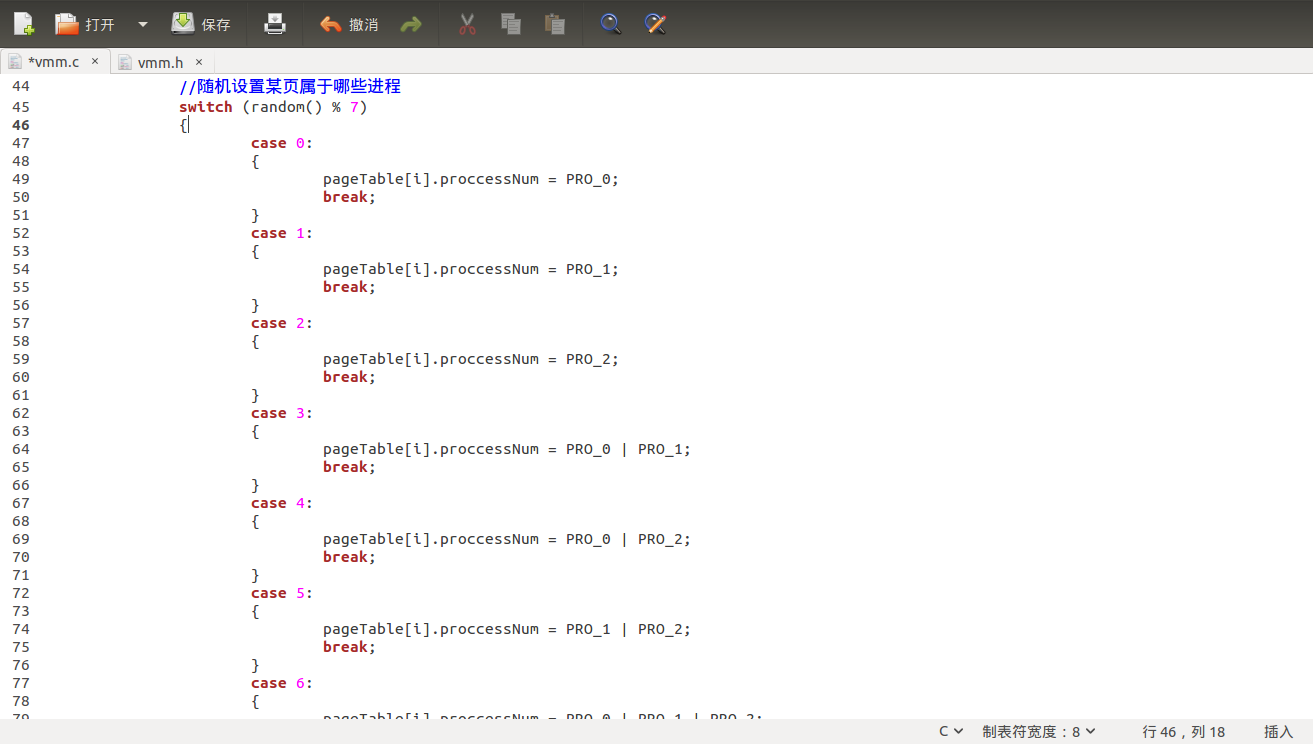
根据索引找到对应的页表项:



* + 1. 多道程序的存取控制

本次实验中实现的是3道进程的存取控制。实现思路为：为每个页表设置属于哪些进程，产生请求时标记该请求所属的进程。程序处理请求时先检查访问的权限，若不能访问则输出提示信息，能访问则继续处理。具体实现过程如下。

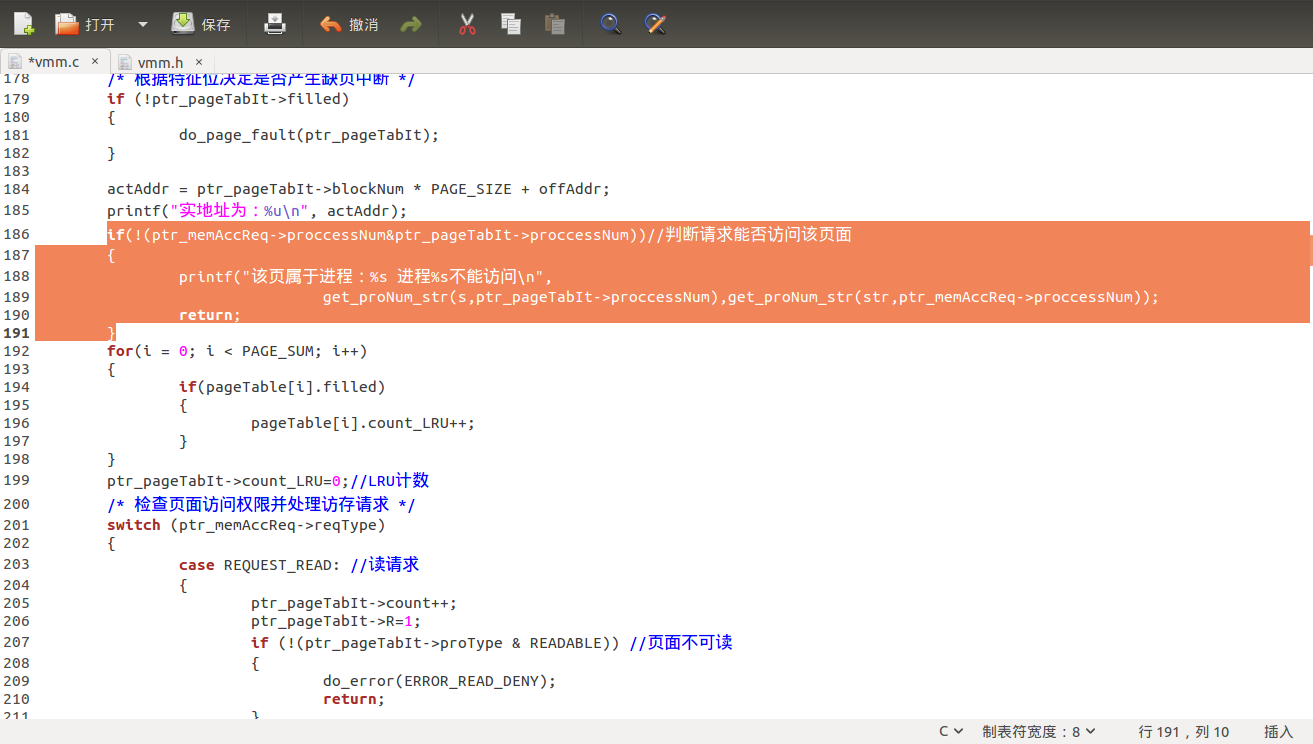
1.随机设置页表属于哪些进程：



2.在产生请求时标记该请求属于哪个进程



3.在访问内存时进行权限判断



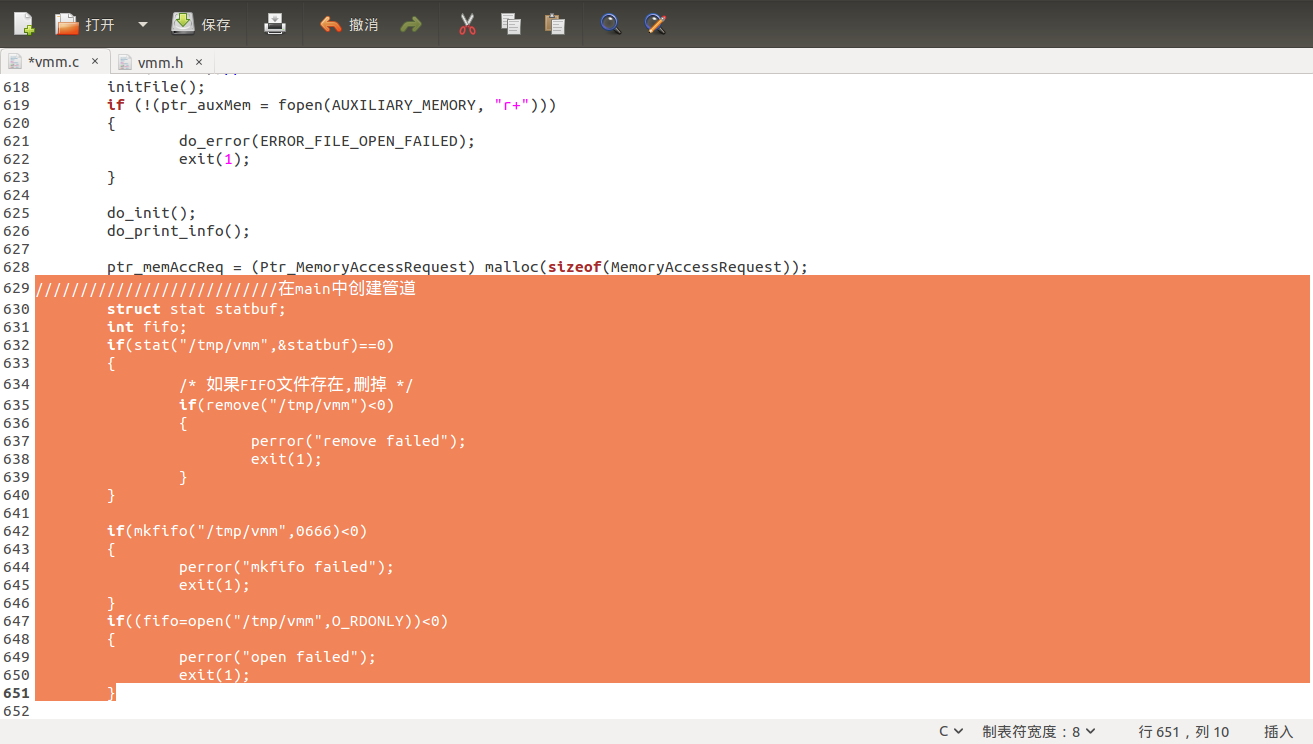
4.为便于查看，实现了打印页面所属进程号的函数



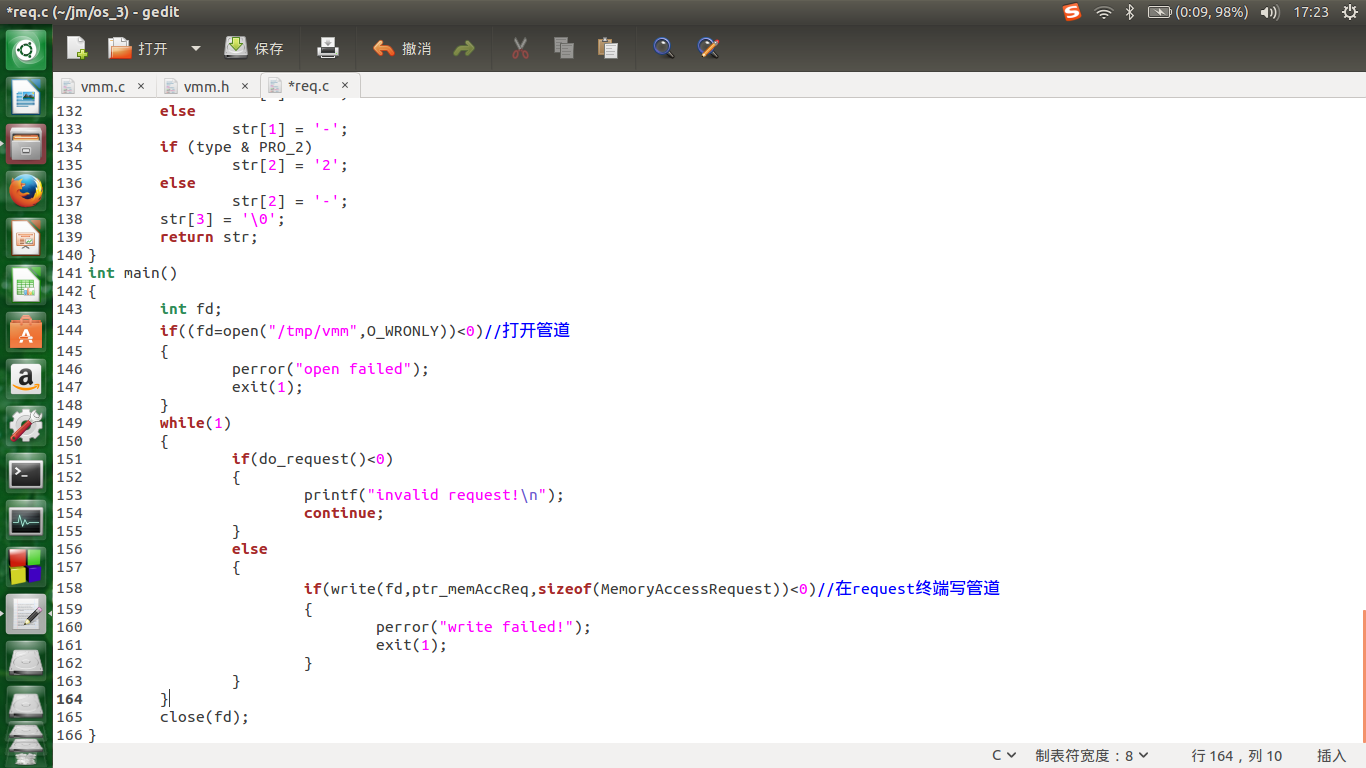
* + 1. 通过FIFO进行进程间通信

实现思路：利用命名管道进行进程间通信，实现do\_request和do\_response异终端。一个进程向管道写请求，另一个进程从管道中读取请求并处理。实现过程如下。

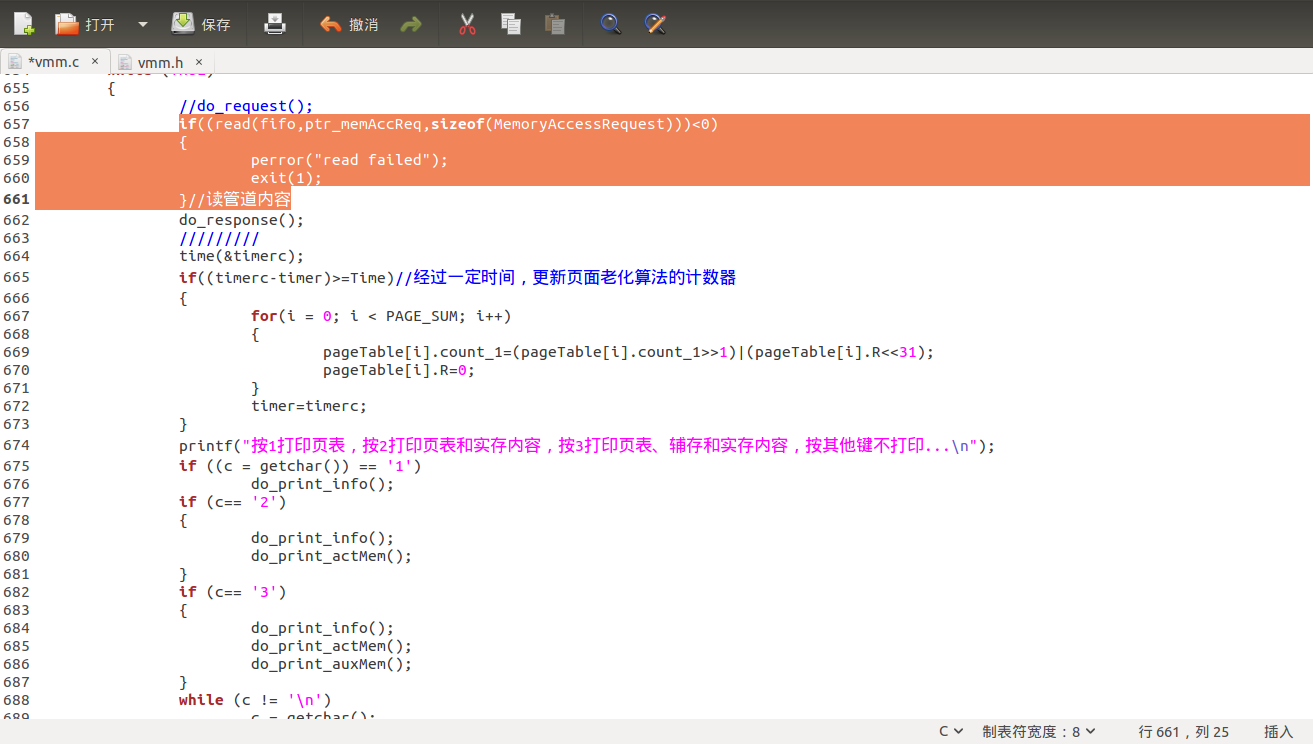
1.在main函数中创建管道



2.在request进程中打开管道，并将请求写入管道



3.在main函数中打开管道并读取请求内容



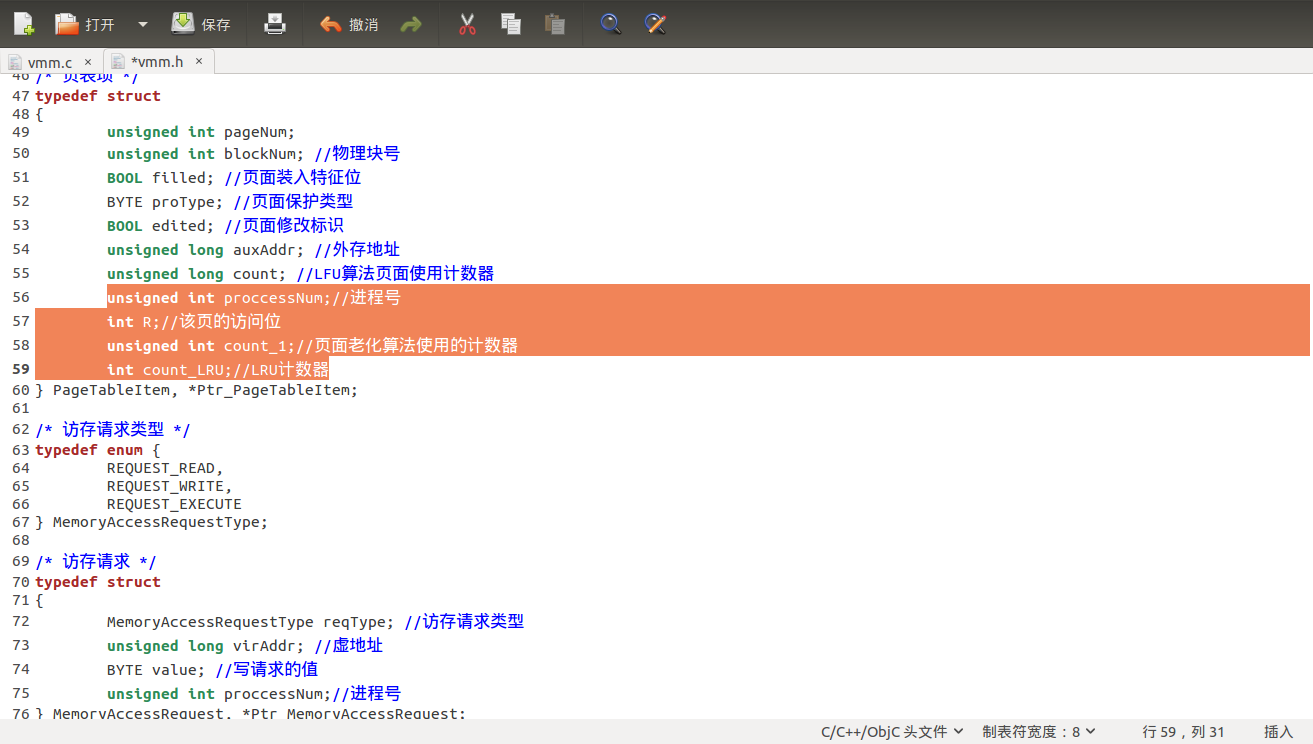
* + 1. 页面老化算法

1. 算法的基本思想与实现方法介绍

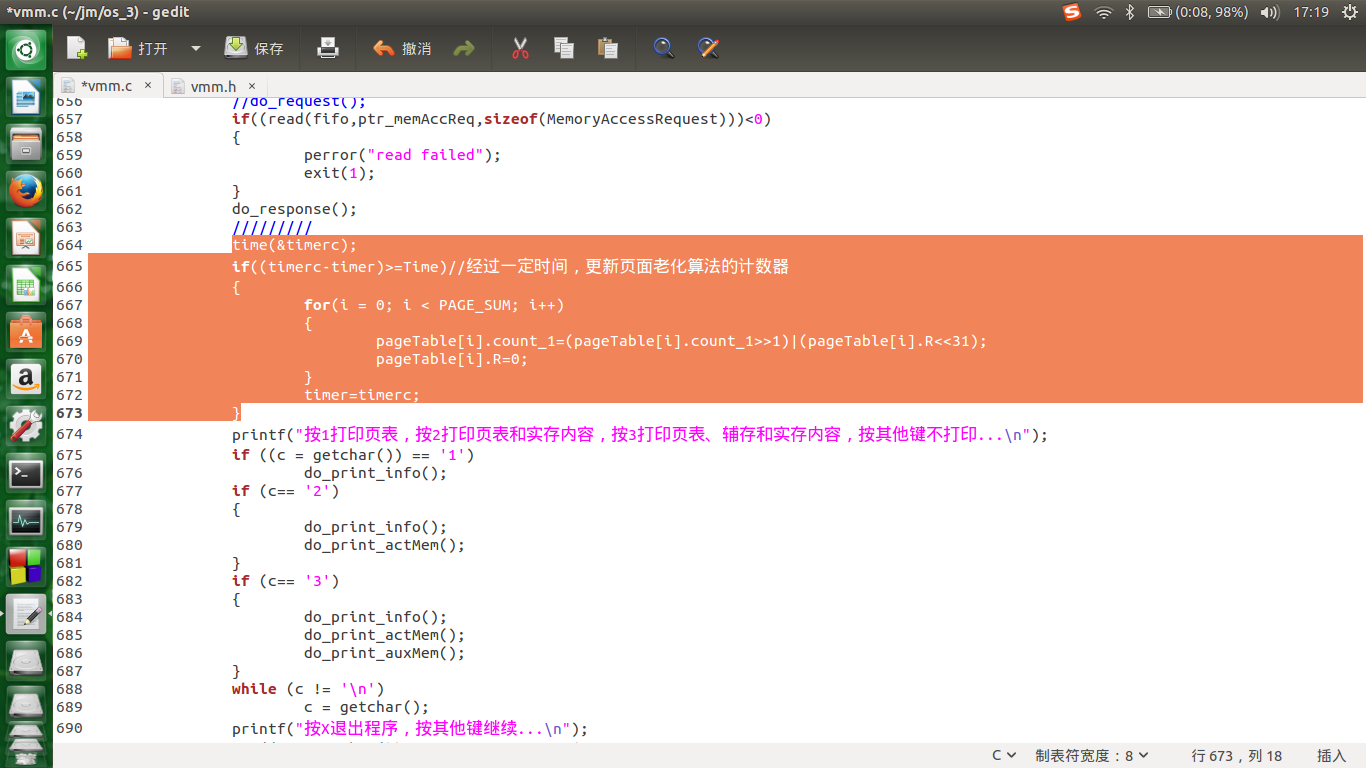
页面老化算法是LRU的一种近似算法，计数过程与时钟相关。本次实验采用的实现方法是每个页表项中添加一个计数器和一个访问位R。每次访问页面时，将被访问的页表项的R置为1。每过一个时间段更新页表项的计数器：先将计数器右移一位，再将R位加到计数器的最左端的位。当需要替换页面时，选择内存中计数值最小的页面替换。

1. 实现代码

在页表项结构中添加计数器和访问位R：



每过一定时间段更新计数器：



缺页中断时调用do\_OldPage算法选择计数值最小的页面替换：



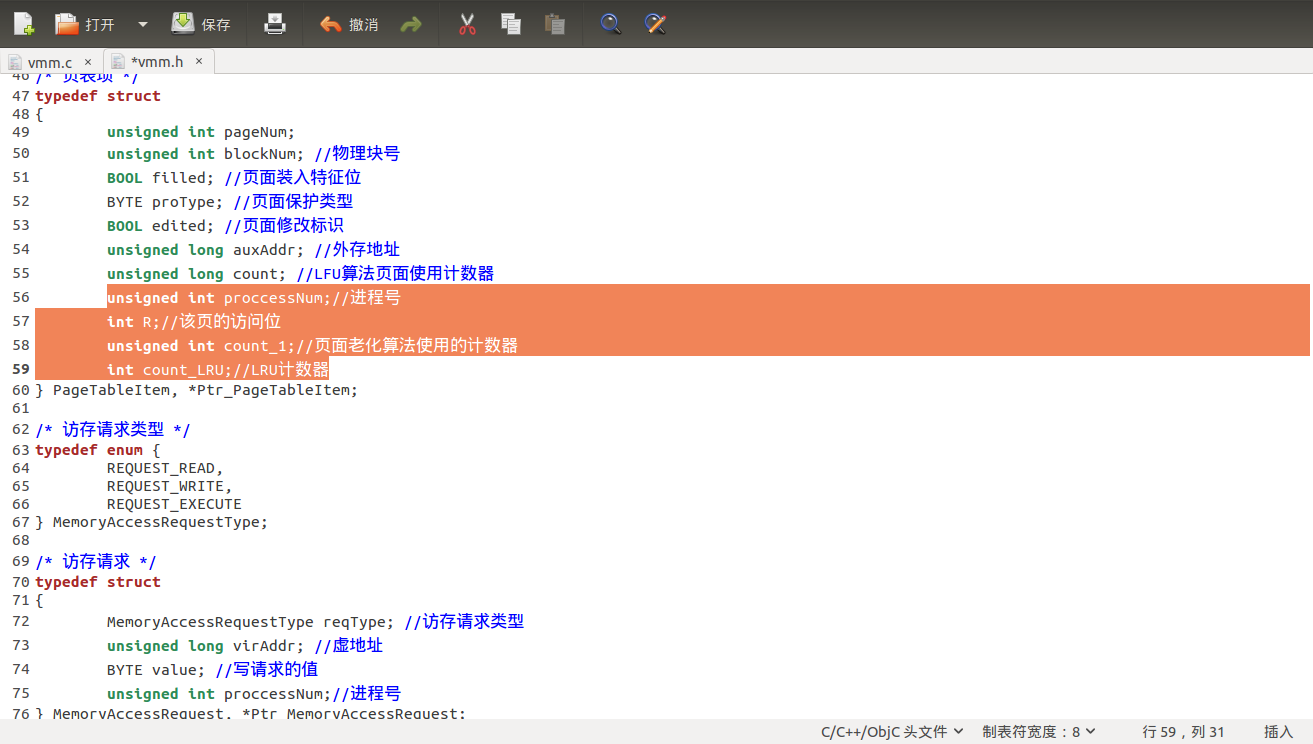
* + 1. LRU算法

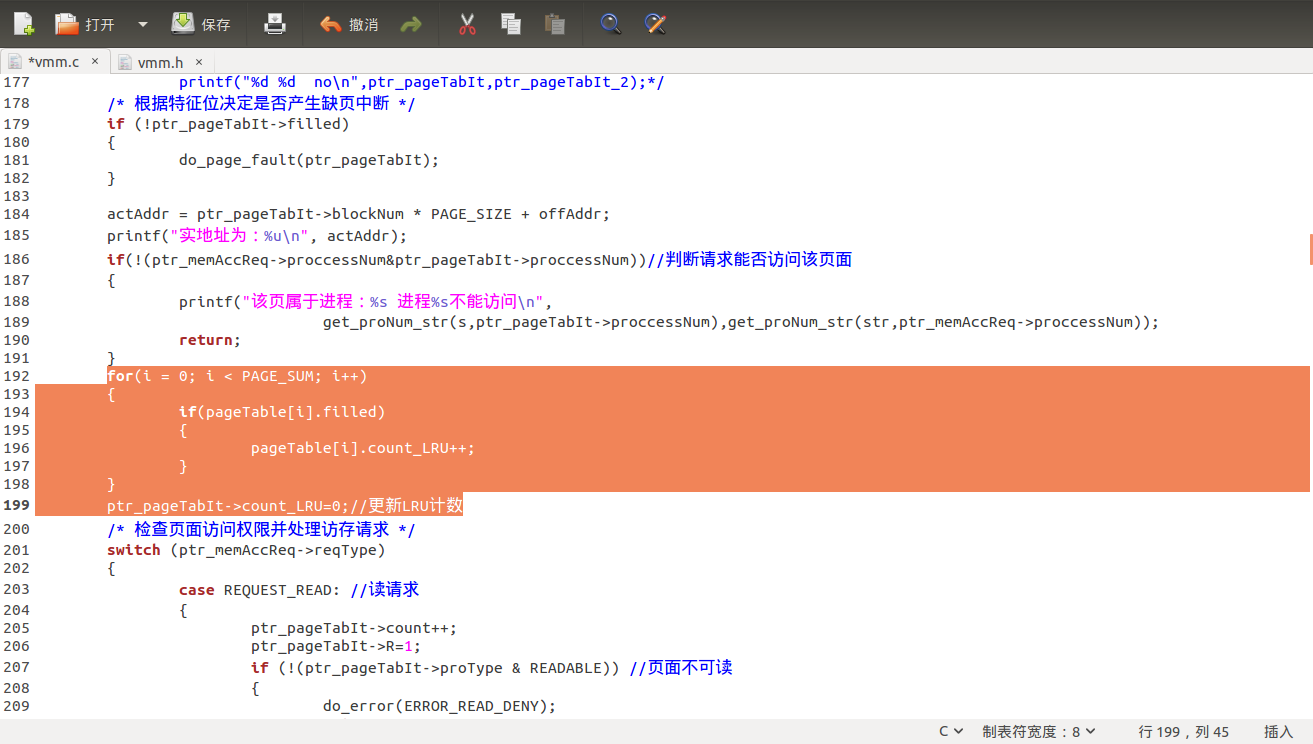
1. 算法基本思想与实现方法介绍

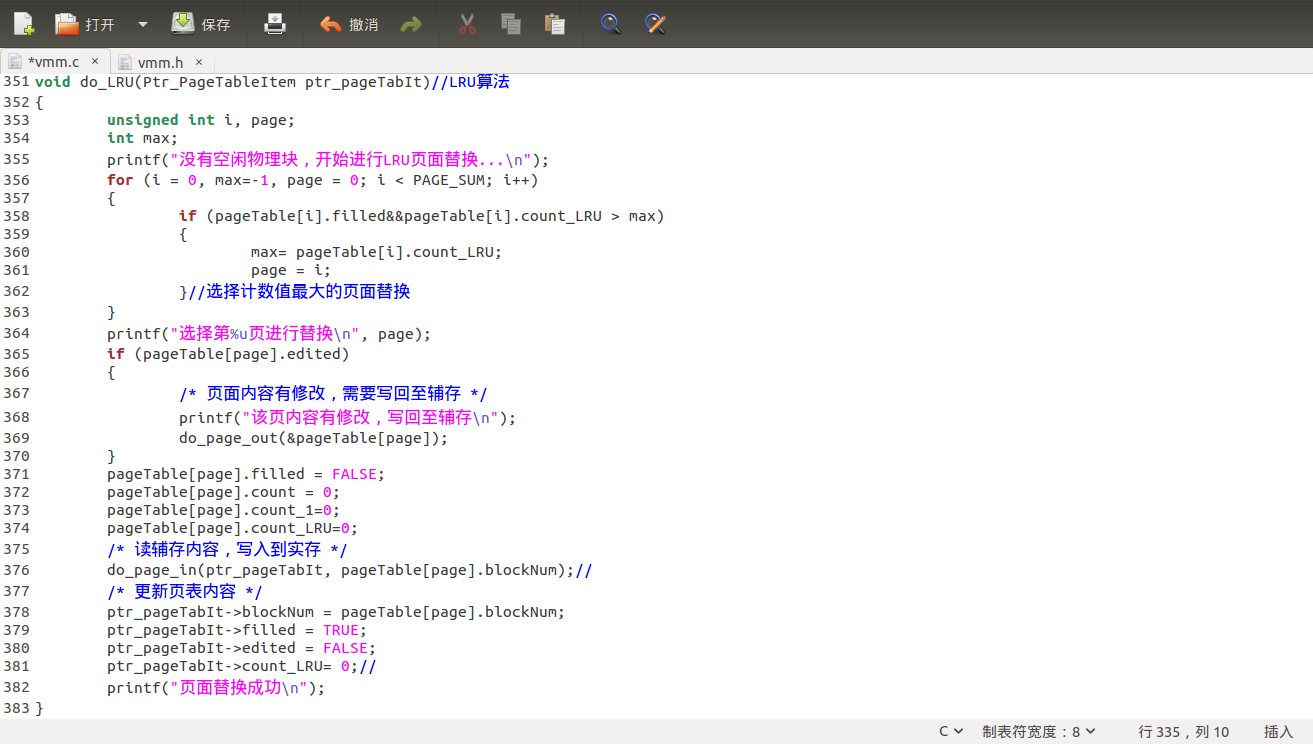
LRU （最近最少未使用） 页面置换算法的核心是置换未使用时间最长的页面。本次实验采用的实现方法是在每个页表项中添加一个计数器，每次访问一个页面时，将该页面的计数器清零，而其他在内存中的计数器均加一。当需要替换页面时，选择内存中计数值最大的页面替换。

1. 实现代码

在页表项结构中添加LRU计数器：

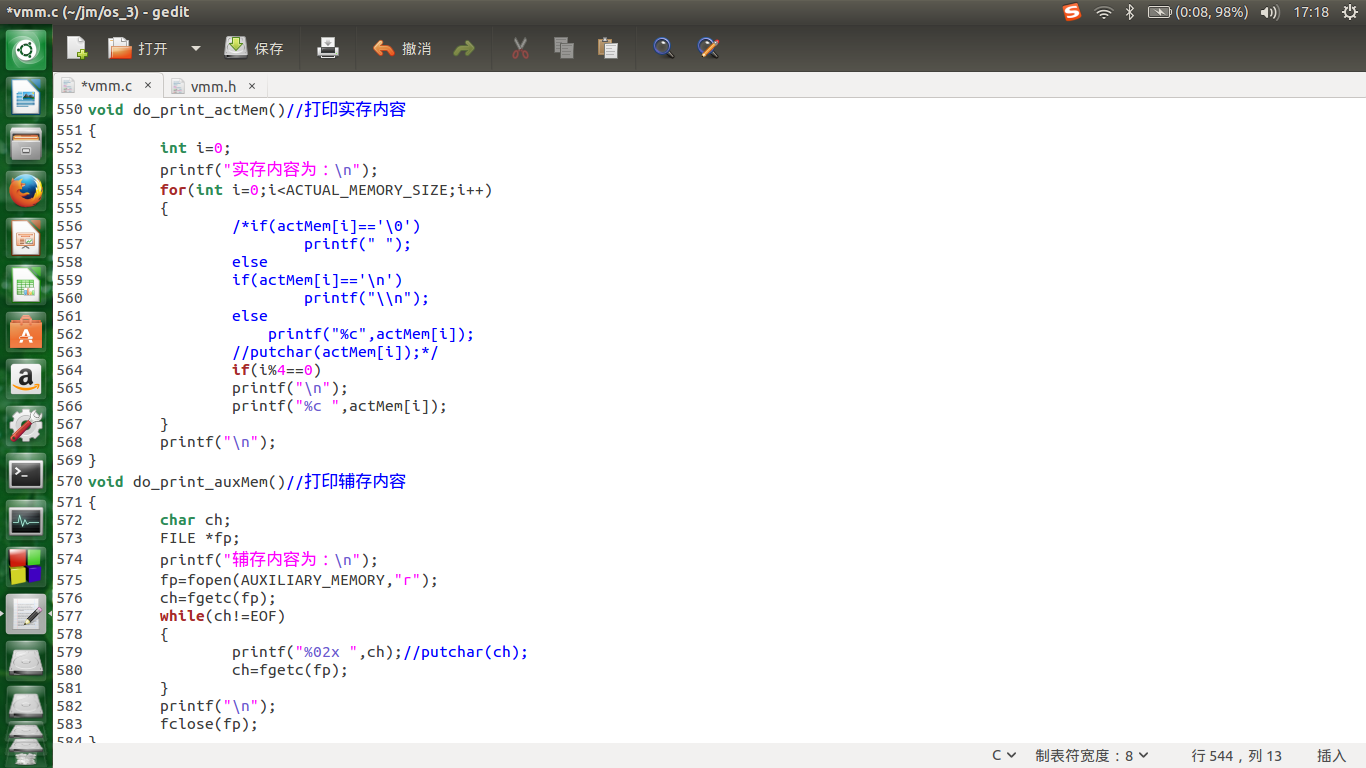
每次访问页面后更新计数器的值：

 缺页中断时调用do\_LRU算法选择计数值最大的页面替换：



* + 1. 实现辅存和实存内容的打印

将辅存和实存内容打印在终端，便于查看。实现代码如下：



1. 问题及解决方法


5. 1. 使用指针遇到的问题

实验中有较多的地方用到指针，但在指针的操作过程中，需要编程者负责指针操作的每一个细节，容易因疏忽而出现各种各样的错误。比如使用空指针、未将不使用的指针释放等。我在编程中也由于疏忽遇到了问题，使用了空指针，导致内存段错误，程序在运行时有一定的概率崩溃。经过使用gdb调试、定位错误，这一问题很容易解决了。

* 1. 源码中存在的问题

源码基本实现了基本要求，但我们在实验过程中也发现了其中一些问题。在实现页面替换算法后，运行程序时，我发现页面的每次替换都不对。经过反复阅读源码，最后发现是源码中do\_LFU函数中寻找计数器的最小值有误。源程序是找所有页面的计数器的最小值，没有考虑该页面是否装入内存，结果每次找的都是为装入的页面进行替换。在程序中应该加一个判断该页面是否装入内存的条件。另外在其他一些地方也发现了一些细小的错误，如do\_page\_out函数最后输出语句的参数写反了等等。

* 1. 一些细节带来的问题

编程时应当十分注意，否则很可能在一些细节上犯错。程序中实现了几个页面替换算法，每个算法都有其计数器。由于不够谨慎，我在最初为计数器命名时没太注意，结果导致在程序中混用了，导致程序中出现了莫名其妙的问题，而这些问题难以发现，浪费了不少时间。上文提到指针使用的问题也是忽略细节造成的。可见编程时有一个良好的编程习惯、注重细节是很重要的。

1. 收获和感想

经过这次实验，我们熟悉了Linux系统下的页式存储管理机制，在实验中我们对课上所学的虚存管理的相关理论知识的有了更深层次的理解，还间接巩固了进程通信的知识。在查找资料的过程中对页面替换算法有了更多的理解。同时，在编写程序的过程中，提高了我们分析问题、发现并解决问题的能力以及团队成员间的沟通协作能力。

本次实验并不太难，但我们在实验过程中也遇到了一些困难。首先是对一些实验要求有不同理解，各组员提出了不同实现方案。其次是编程时遇到的问题，如使用未初始化的指针等。不过我们通过上网查询相关资料、与同学们讨论、询问助教、使用gdb调试等方式逐一解决了这些问题。正是在不断发现和解决问题的过程中，我们不断得到了提高。