**暨南大学本科实验报告专用纸**

课程名称 操作系统原理实验 成绩评定

实验项目名称 页面置换算法与动态内存分配指导教师 郝振明

实验项目编号 0806015706 实验项目类型 验证 实验地点

学生姓名 张印琪 学号 2018051948

学院 信息科学技术学院 系 计算机科学系 专业 网络工程

实验时间 20XX 年 X 月 X日 上 午～ X 月 X 日 上 午

温度 ℃湿度

1. **实验目的** 
2. 掌握OPT、FIFO、LRU、LFU、Clock等页面置换算法；
3.  掌握可用空间表及分配方法；
4.  掌握边界标识法以及伙伴系统的内存分配方法和回收方法。

**三、 实验内容**

由于在 Linux 0.11 内核中还没有实现虚拟内存的管理，当然也不存在页面置换算法，而 且修改 Linux 0.11 内核的动态内存分配会比较复杂，所以本实验在 Windows 控制台中通过模 拟来实现页面置换算法和动态内存分配。 3.1 页面置换算法 3.1.1 准备实验 1. 启动 Engintime Linux Lab。 2. 使用“控制台应用程序(c)”模板，新建一个 Windows 控制台应用程序项目。

Linux 内核实验教程

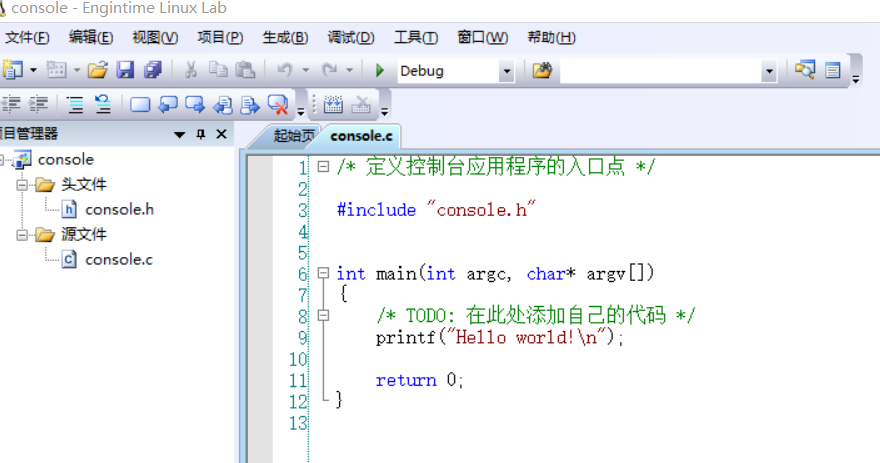
101

北京英真时代科技有限公司 http://www.engintime.com

3.1.2 查看最佳页面置换算法(OPT)和先进先出页面置换算法(FIFO)的执行过程 在“学生包”本实验对应文件夹中，提供了实现最佳页面置换算法（OPT）和先进先出 页面置换算法（FIFO）的源代码文件 page.c。使用 Linux Lab 打开此文件（将文件拖到 Linux Lab 窗口中释放即可打开），仔细阅读此文件中的源代码和注释，查看 OPT 和 FIFO 是如何实 现的。 其中，在 main 函数中定义了一个数组 PageNumofR[]来存放页面号引用串，定义了一个 指针\*BlockofMemory 指向一块存放页面号的内存块。依次将数组 PageNumofR[]中的页面装 入内存，根据不同的页面置换算法淘汰内存中的页面。 按照下面的步骤查看 OPT 和 FIFO 的执行过程： 1． 使用 page.c 文件中的源代码，替换之前创建的 console.c 文件内的源代码。 2． 按 F7 键生成修改后的 c 控制台应用程序项目，确保没有语法上的错误。 3． 按 Crtl+F5 执行此程序，查看 OPT 和 FIFO 的执行过程。执行结果如下图所示：

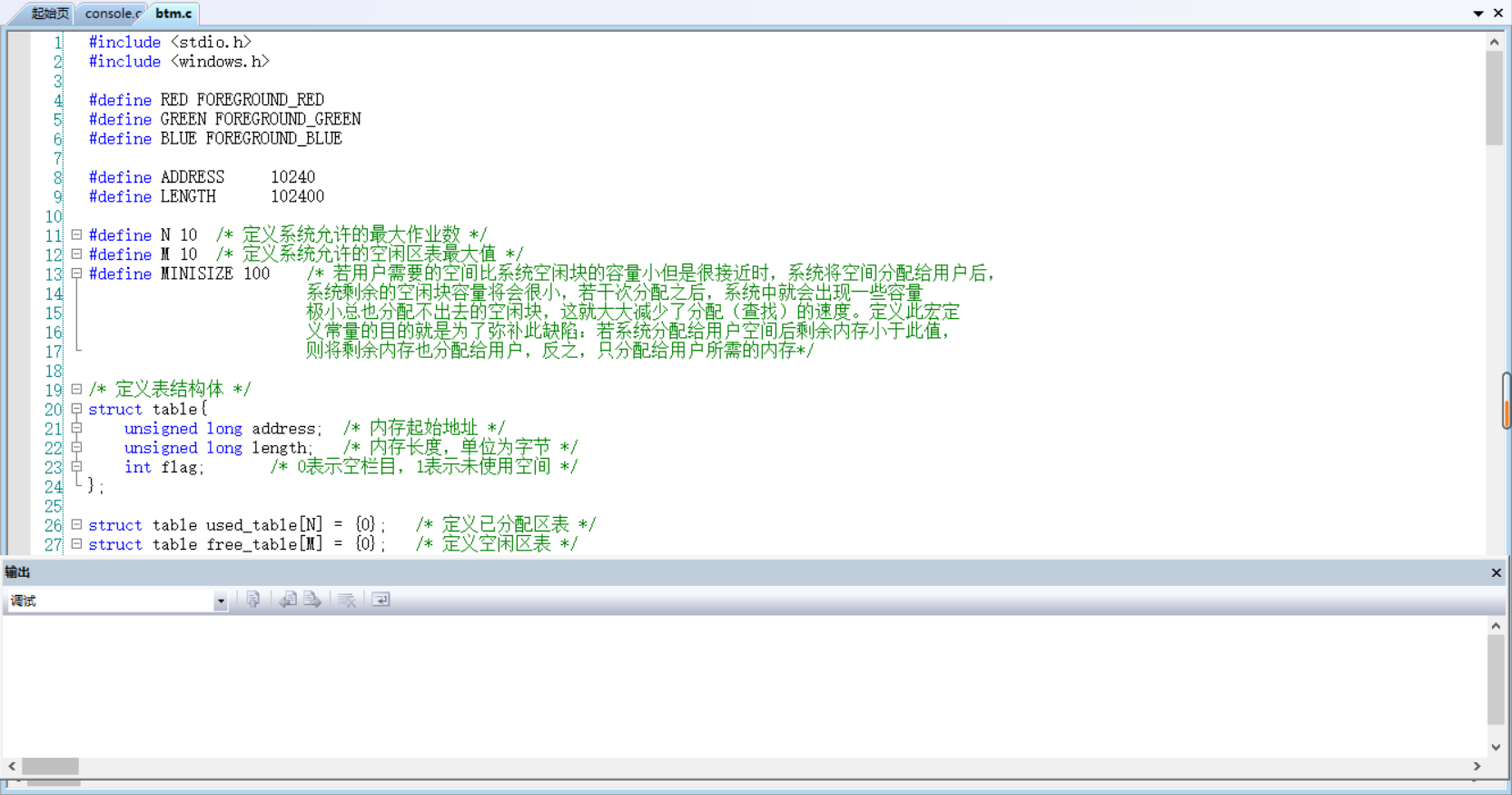
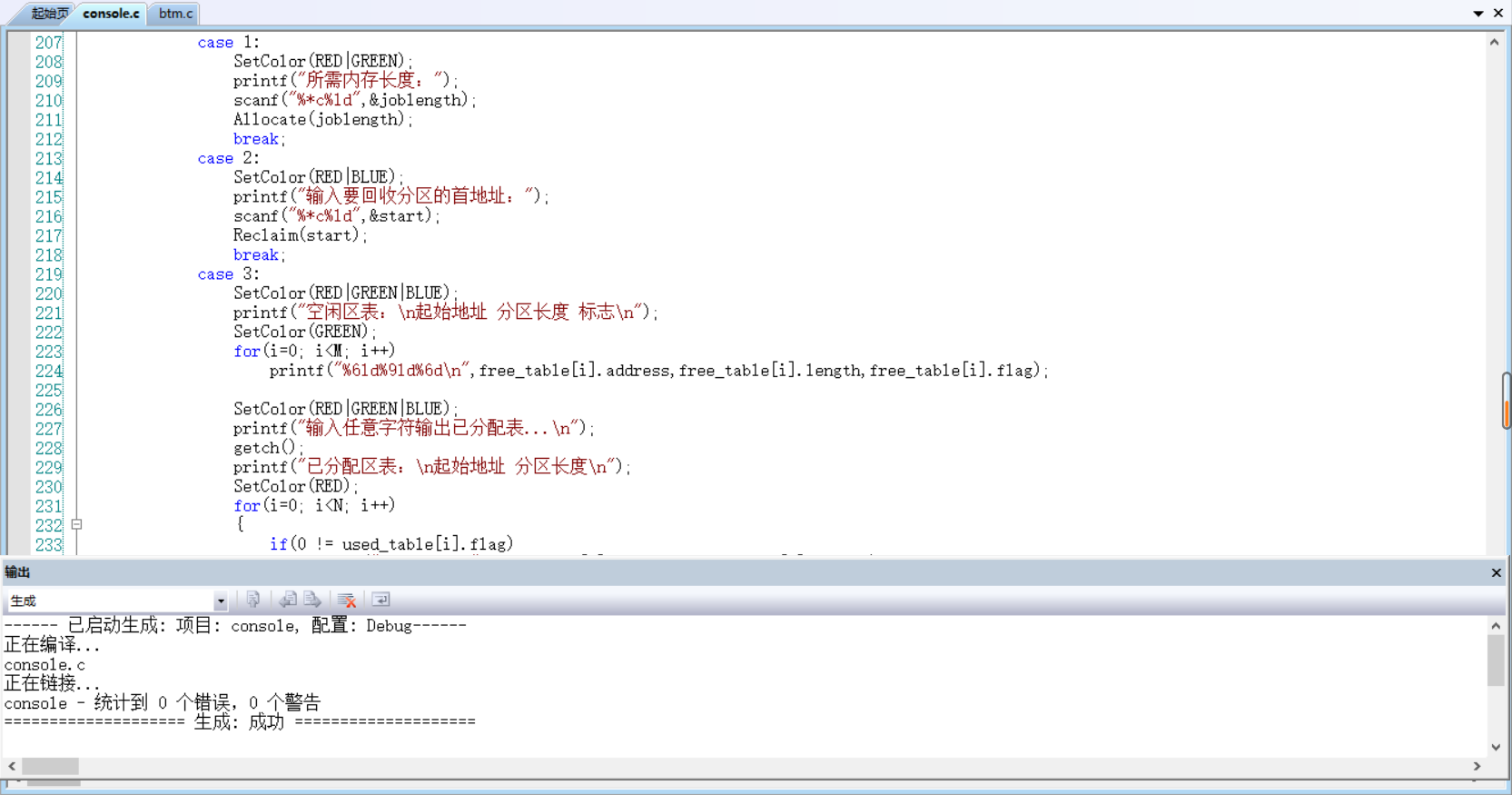
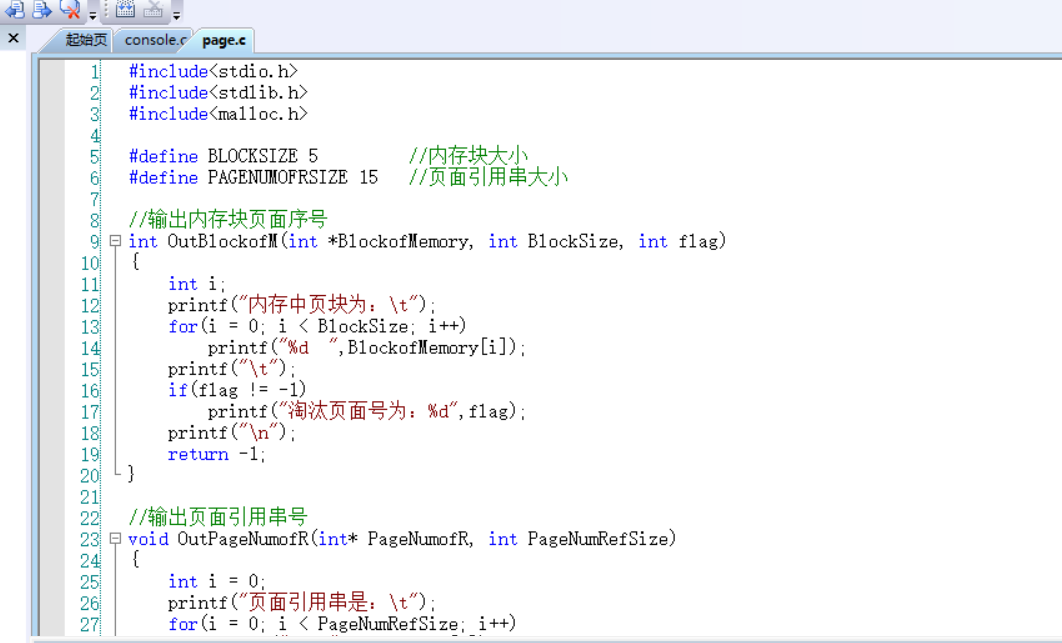
图 9-1 OPT 和 FIFO 的执行过程

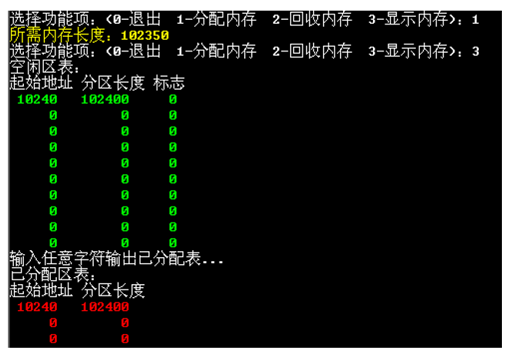
Linux 内核实验教程



102

北京英真时代科技有限公司 http://www.engintime.com

3.1.3 完成最近最久未使用页面置换算法和简单的 clock 页面置换算法 “学生包”本实验对应文件夹下的 page2.c 文件中提供了两个算法的参考代码。将 page2.c 文件内的源代码拷贝到 console.c 中，仔细阅读其中的源代码，并根据提示实现最 近最久未使用(LRU)页面置换算法和简单的 Clock 页面置换算法。 有兴趣的读者还可以使尝试写出最少使用（LFU）页面 置换算法和页面缓冲页面置换（PBA） 算法。 3.2 动态内存分配 3.2.1 准备实验 使用“控制台应用程序(c)”模板，新建一个 Windows 控制台应用程序项目。 3.2.2 边界标识法的设计实现 在“学生包”本实验对应文件夹中，提供了实现边界标识法的源代码文件 btm.c，将其 拖到 Linux Lab 中释放，即可打开此文件。仔细阅读其中的源代码及注释，阅读时着重注意 以下几点：  定义 MINISIZE 的目的是为了防止在多次分配以后链表中出现一些极小的、总也分 配不出去的空闲块；  表结构体 struct table{ unsigned long address; unsigned long length; int flag; }; 其中的 address 是内存的起始地址；length 是内存长度，单位为字节；flag 是标 志位，为 0 表示空栏目，为 1 表示内存未使用。  定义了 used\_table[]和 free\_table[]两个数组来分别存放已分配区表和空闲区表；

 allocate 函数是采用首次适应算法分配内存的。  回收内存 reclaim 函数，在回收内存时要跟空闲分区中的左右进行比较，若有空闲 块则进行内存合并。  SetColor 是设置字体的函数，是为了使输出结果更加醒目。 按照下面的步骤查看边界标识法的实现过程： 1. 用 btm.c 文件内的代码替换刚刚创建的控制台应用程序项目的 console.c 文件内 的代码。 2. 按 F7 生成项目，确保没有语法错误。 3. 按 Ctrl + F5 运行项目，查看运行结果。其结果如下图所示： 

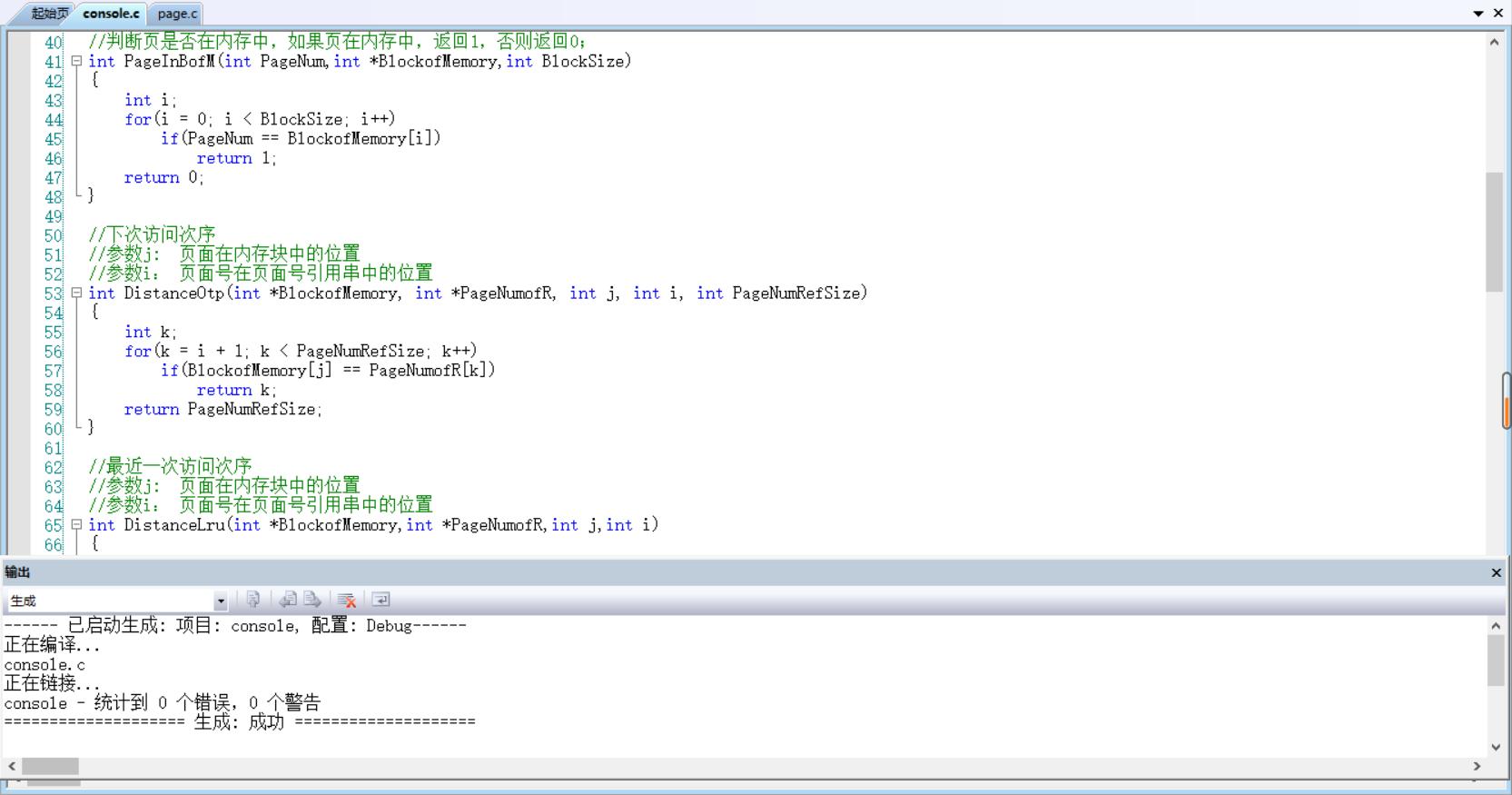
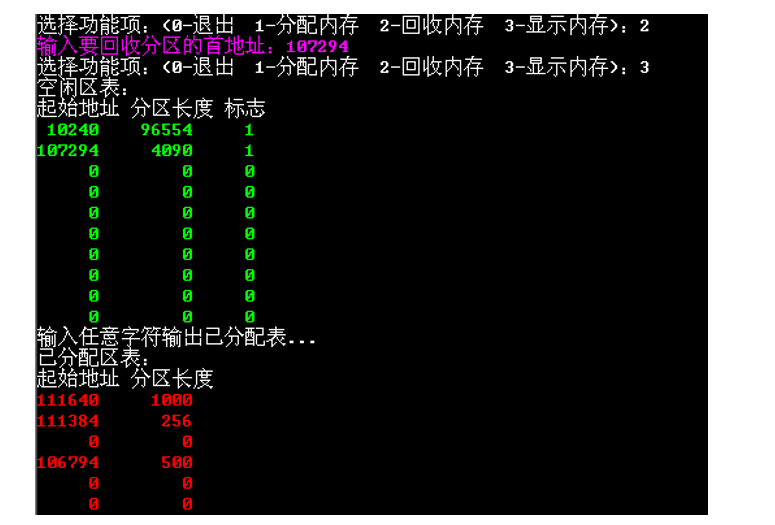
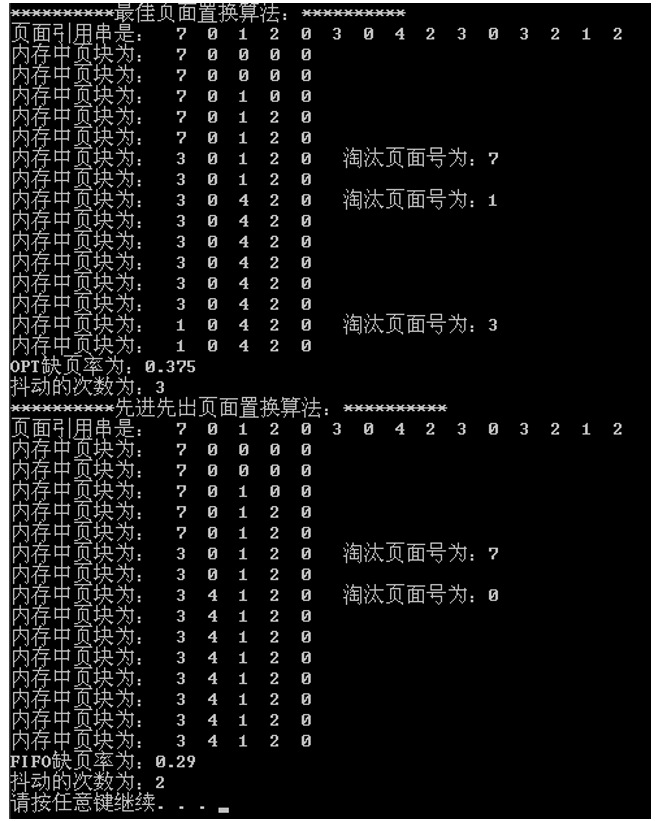


图 9-2 边界标识法的运行结果 系统会停留在此界面等待用户输入功能项序号 0-3 中的其中一个。系统会根据用户输入 的序号执行相应的功能。如果没有按数字 0，系统会执行完相应的功能后继续停留在此界面 等待用户的输入，直到用户按 0 退出应用程序。 例如，输入 1 来分配内存，所需内存长度 102350，由于初始化的内存大小为 102400 字， 申请内存以后系统剩余可用内存大小小于 100 字，所以此时再输入 3 查看内存，可以看到系 统将 102400 字的内存全都分配给来该作业。如图 9-3 所示。如果输入的作业长度小于 102300， 则申请多少内存，系统就分配多少内存。如果申请的作业数大于 10 个，则会提示“已分区



Linux 内核实验教程

103

北京英真时代科技有限公司 http://www.engintime.com

表已满，无法继续分配内存”。如图 9-4 所示。当然还可以输入 2 来回收内存。如果要回收 的内存左右有空闲区，就会与之合并留在可利用空间表，否则就作为一个结点插入到可利用 空间表中。如图 9-5 所示。 分配空间时，采用的是首次适应算法，请读者尝试采用循环首次适应算法和最佳适应算 法分别实现边界标识法。

图 9-3 申请 102350 字节后的内存分配

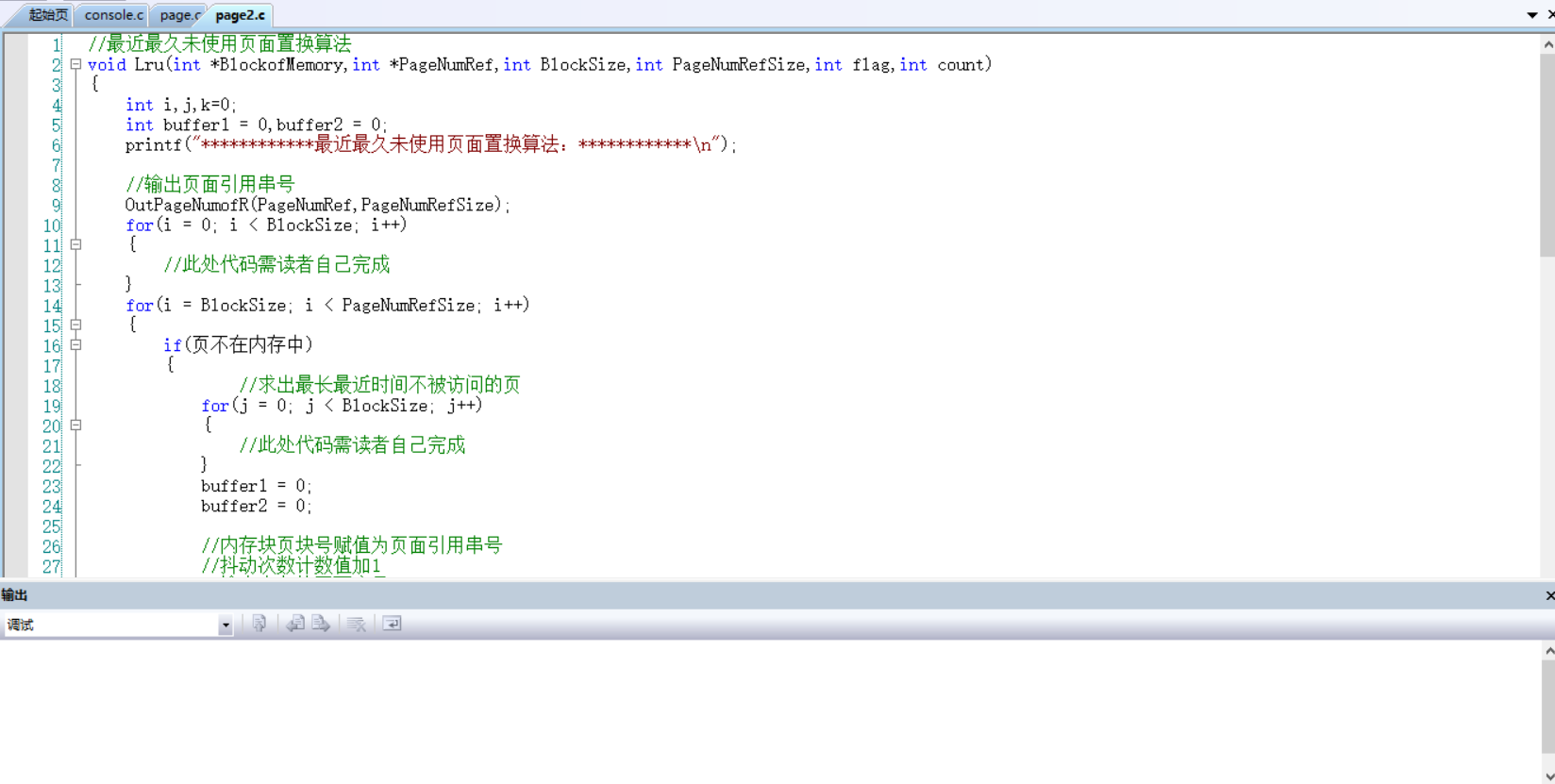
图 9-4 申请第 11 个内存块时的内存显示

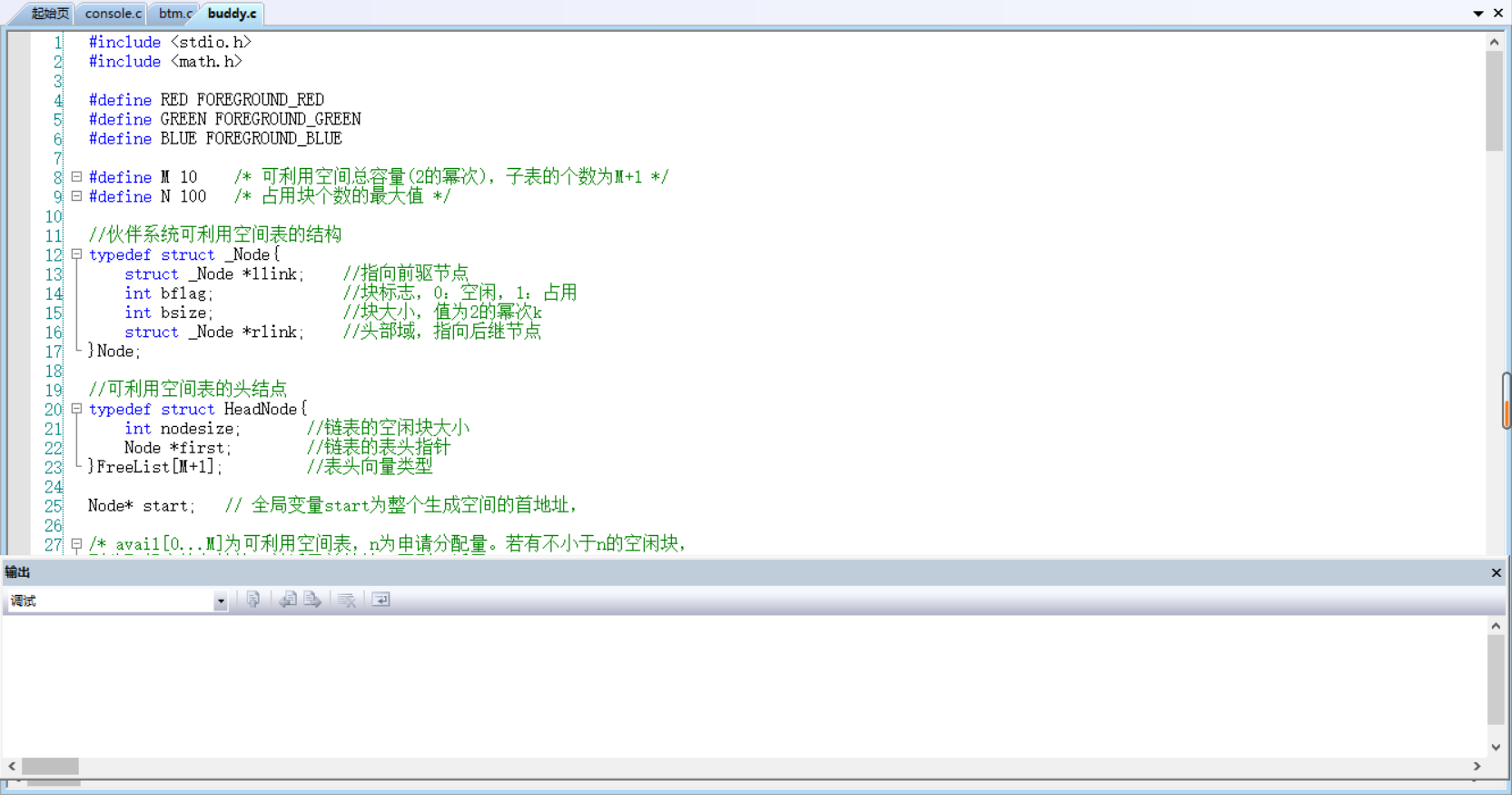
Linux 内核实验教程

104

北京英真时代科技有限公司 http://www.engintime.com

图 9-5 回收内存



3.2.3 伙伴系统的设计实现 在“学生包”本实验对应文件夹中，提供了实现伙伴系统的部分源代码文件 buddy.c。 将其拖到 Linux Lab 中释放，即可打开此文件。用此文件内的代码替换刚刚创建的 Windows 控制台应用程序项目的 console.c 文件内的代码。仔细查看其中的源代码及注释，并根据注 释设计完善伙伴系统。 阅读代码时注意以下几点： ● 伙伴系统可利用空间表的结构体 typedef struct \_Node{ struct \_Node \*llink; //指向前驱节点 int bflag; //块标志，0：空闲，1：占用 int bsize; //块大小，值为 2 的幂次 k struct \_Node \*rlink; //指向后继节点 }Node; 可利用空间表的头结点的结构体 typedef struct HeadNode{ int nodesize; //链表的空闲块大小 Node \*first; //链表的表头指针 }FreeList[M+1]; ● 在 AllocBuddy 函数中分配空闲块以后，还需要将剩余块插入相应的子表中。 ● Reclaim 函数回收内存时，需要先查找其伙伴是否为空闲块。若否，则只要将释放 的空闲块简单插入相应的子表中即可；若是，则需在相应子表中找到其伙伴并删除 之，然后在判别合并后的空闲块的伙伴是否是空闲块。依此重复，直到归并所得空 闲块的伙伴不是空闲块时，再插入到相应的子表中去。 

**总结：**

**本次实验我掌握了以下知识**

**页面置换算法 请求分页虚存管理的实现原理是：把作业的所有分页副本存放在磁盘中，当它被调度投 入运行时，首先把当前需要的页面装入内存，之后根据程序运行的需要，动态装入其他页面； 当内存空间已满，而又需要装入新页面时，根据某种算法淘汰某个页面，以便装入新页面。 因此，在页表中必须说明哪些页已在内存，存在什么位置；哪些页不再内存，它们的副本在 磁盘中的什么位置。还可以设置页面是否被修改过，是否被访问过，是否被锁住等标志供淘 汰页面使用。 在地址映射过程中，若页表中发现所要访问的页不在内存，则产生缺页异常，操作系统 接到此信号后，就调出缺页异常处理程序，根据页表中给出的磁盘地址，将该页面调入内存， 使作业继续运行下去。如果内存中有空闲页，则分配一个页，将新调入页面装入，并修改页 表中相应页表项的驻留位以及相应的内存块号；若此时内存中没有空闲页，则要淘汰某页面， 若该页在此期间被修改过，还要将其写回磁盘，这个过程称为页面替换。 动态内存分配 边界标识法是操作系统中用以进行动态分区分配的一种存储管理方法。系统将所有的空 闲块链接在一个双重循环链表结构的可利用空间中；分配可按首次拟合进行，也可按最佳拟 合进行。其特点在于：在每个内存区的头部和尾部两个边界上分别设有标识，以标识该区域 为占用块或空闲块，使得在回收用户释放的空闲块时易于判别在物理位置上与其相邻的内存 区域是否为空闲块，以便将所有地址连续的空闲快组合成一个尽可能大的空闲块。 伙伴系统是操作系统中用到的另一种动态存储管理办法。它和边界标识法类似，在用户 提出申请时，分配一块大小“恰当”的内存区给用户；反之，在用户释放内存时即回收。所 不同的是：在伙伴系统中，无论是占用块或空闲块，其大小均为 2 的 k 次幂（ k 为某个整数）。**