## 实验六：存储管理

7．编写一个程序，利用内存映象文件，实现less工具的功能（多屏显示）。

答：

程序代码如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  3435  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92 | #include <stdio.h>  #include <sys/stat.h>  #include <fcntl.h>  #include <unistd.h>  #include <sys/mman.h>  #include <string.h>  #include <memory.h>  #include <stdlib.h>  #include <stdio.h>  int lastrow(char \*s, int d);  int nextrow(char \*s, int d);  int onepage(char \*s, int d);  int main()  {  int fd, play = 0;  char lab;  char \*start;  struct stat sb;  fd = open("7.c", O\_RDONLY); // 以只读方式打开文件  fstat(fd, &sb); // 获取文件的大小  start = mmap(NULL, sb.st\_size, PROT\_READ, MAP\_PRIVATE, fd, 0);  if (start == MAP\_FAILED) // MAP\_FAILED表示映射失败  return (1);  play = onepage(start, play) + 1;  lab = getchar();  while (lab != 'q') // 输入的字符为q，退出  {  if (play > sb.st\_size) // 如果onepage返回的字节数大于文件的大小，输入任意字符退出  {  lab = getchar();  break;  }  else if (lab == 'p') // 输入p，继续读10行  play += onepage(start, play) + 1;  else if (lab == 'n') // 输入n，显示下一行  play += nextrow(start, play) + 1;  else if (lab == 'l') // 输入l，显示上一行  play = lastrow(start, play) + 1;  lab = getchar();  }  munmap(start, sb.st\_size); // 解除映射  close(fd); // 关闭文件fd  return 0;  }  int onepage(char \*s, int d)  {  int i, count = 0; // count在这里表示文件中行的数量  char \*buffer = malloc(2048); // 配置内存空间，由buffer指向该空间  s += d; // 每10行作为一页输出  for (i = 0; i < 2048; i++)  {  if (s[i] == '\n')  count++;  if (count == 10)  break;  }  memcpy(buffer, s, i); // 从s处开始的地方拷贝i个字节到buffer  buffer[i] = '\0'; // 添加结束标识  printf("%s\n", buffer);  return i;  }  int nextrow(char \*s, int d) // 下一行  {  int i;  char \*buffer = malloc(100);  s += d;  for (i = 0; i < 100; i++)  if (s[i] == '\n')  break;  memcpy(buffer, s, i);  buffer[i] = '\0';  printf("%s\n", buffer);  return i;  }  int lastrow(char \*s, int d) // 上一行  {  int i, count = 0;  char \*buffer = malloc(100);  int py = d;  for (; d > 0; d--)  {  if (s[d] == '\n')  count++;  if (count == 2)  break;  }  memcpy(buffer, s + d + 1, py - d - 2);  buffer[py - d - 2] = '\0';  printf("%s\n", buffer);  return d;  } |

这段代码先使用fstat函数获得文件的大小，保证后续对文件内容的读取操作不发生越界，然后使用mmap函数将文件的内容映射到内存中。其中第一个参数start为NULL时表示由系统决定映射区的起始地址；第二个参数length表示映射区的长度，不足一页按一页处理，这里即为前面取得的文件大小；第三个参数期望的内存保护标志prot的PROT\_READ表示页内容可以被读取；第四个参数映射的对象的类型flags的MAP\_PRIVATE表示建立一个写入时拷贝的私有映射。内存区域的写入不会影响到原文件；第五个参数fd表示有效的文件描述词，一般是由open函数返回；第六个参数off\_toffset表示被映射对象内容的起点。mmap函数的返回值为映射区内存的起始地址，此后调用的三个函数onepage、nextrow、lastrow的功能分别是显示一页内容、显示下一行内容、显示上一行内容。

运行该程序，映射7.c的文件内容到内存，使用p显示一页（10行）内容，使用n显示下一行内容，使用l显示上一行内容，效果如下图：

