**实验三 内存分配（页式管理）**

**结果：**



代码：

#include <iostream>

#include <time.h>

#include <assert.h>

using namespace std;

#ifdef \_UNICODE

#undef \_UNICODE

#endif

#define MEM\_TOTAL 0x80

#define BLOCK\_UNIT 0x10

#define PAGE\_TOTAL 0x10

#define GetBlock(x) (x>>4)

#define GetMem(x,y) ((x<<4)|(y&0x0F))

#define MEM\_BLOCK GetBlock(MEM\_TOTAL)

typedef unsigned char Uint8;

typedef unsigned int Uint32;

char Memory[MEM\_TOTAL];

FILE \*fp;

typedef struct Page

{

Page():Exist(0){}

Uint8 MemPage;

Uint8 Exist;

}Page;

Page TotalPage[PAGE\_TOTAL];

Uint8 Fifo\_ptr=0;

char Fifo[MEM\_BLOCK];

void InitFifo()

{

for(int i=0;i<MEM\_BLOCK;++i)

Fifo[i]=-1;

}

void ExecFile(const char \*filename)

{

fp=fopen(filename,"r");

if(!fp)

exit(0);

InitFifo();

srand((unsigned int)time(NULL));

}

void EndFile()

{

fclose(fp);

}

void AllocMemory(Uint8 block)

{

Uint8 freeMem;

if(Fifo[Fifo\_ptr]!=-1)

{

//废弃一个页面

Uint8 freeblock=Fifo[Fifo\_ptr];

TotalPage[freeblock].Exist=0;

freeMem=TotalPage[freeblock].MemPage;

}

else

freeMem=Fifo\_ptr;

Fifo[Fifo\_ptr]=block;

TotalPage[block].MemPage=freeMem;

TotalPage[block].Exist=1;

//读取文件中的字符

assert(fp);

fseek(fp,block<<4,SEEK\_SET);

fread(&Memory[freeMem<<4],sizeof(char),BLOCK\_UNIT,fp);

Fifo\_ptr=(++Fifo\_ptr)%MEM\_BLOCK;

}

void VisitMemory(Uint8 addr)

{

Uint8 memAddr;

char content;

Uint8 block=GetBlock(addr);

if(!TotalPage[block].Exist)

{

//调入页面

AllocMemory(block);

TotalPage[block].Exist=1;

}

memAddr=GetMem(TotalPage[block].MemPage,addr);

content=Memory[memAddr];

printf("---------------------------------------------------------\n");

printf("该块内容为:\n");

for(int i=0;i<0x10;++i)

{

printf("%c",Memory[GetMem(TotalPage[block].MemPage,i)]);

}

printf("\n");

printf("访问逻辑地址0x%x,映射物理地址0x%x,得到内容:%c\n\n",addr,memAddr,content);

}

int main()

{

ExecFile("Access.txt");

for(int j=0;j<100;++j)

{

for(int i=0;i<10;++i)

{

VisitMemory(rand()%0x100);

}

printf("--------------------\n");

getchar();

}

EndFile();

return 0;

}