

第三章 实验三

模拟时间片轮转调度算法

按照时间片轮转调度进程

- 动态地输入进程(key,run_time,message)，按照输入次序建立就绪队列
- 输入 CPU 运行的单位时间片(cpu_base_time)
- 按照时间片轮转方式模拟进程逐个被调度并执行单位时间片(运行结束进程结束，否则修改进程运行时间 run_time,将该进程放置在就绪队列尾巴)。

例如：

```
新建的进程控制表为：
key run_time message
1 2 process1
2 4 process2
3 6 process3
4 3 process4
0 0 0
```

```
CPU运行的单位时间片cpu_base_time为：
2
```

输入多个进程

```
第1次被调度的就绪进程为：
key=1,run_time=2,message= process1

recent table is:
2,4, process2
3,6, process3
4,3, process4

第2次被调度的就绪进程为：
key=2,run_time=2,message= process2

recent table is:
3,6, process3
4,3, process4
2,2, process2

第3次被调度的就绪进程为：
key=3,run_time=2,message= process3

recent table is:
4,3, process4
2,2, process2
3,4, process3

第4次被调度的就绪进程为：
key=4,run_time=2,message= process4

recent table is:
2,2, process2
3,4, process3
4,1, process4
```

输入时间片

```
第5次被调度的就绪进程为：
key=2,run_time=2,message= process2

recent table is:
3,4, process3
4,1, process4

第6次被调度的就绪进程为：
key=3,run_time=2,message= process3

recent table is:
4,1, process4
3,2, process3

第7次被调度的就绪进程为：
key=4,run_time=1,message= process4

recent table is:
3,2, process3

第8次被调度的就绪进程为：
key=3,run_time=2,message= process3
distribute is over!
Press any key to continue
```

输出结果

- 使用 Microsoft Visual Studio C++ 6.0 或 CodeBlocks 编程：程序 3_4_timeslice.cpp。完善如下程序代码：

```
#include <malloc.h>
#include <stdio.h>
#include <string.h>
#define NULL 0
typedef struct table
{
    int key;                /*进程 ID 号*/
    int run_time;           /*进程运行时间*/
    char message[10];       /*进程说明信息*/
    struct table *next;
}node;
node *creat(void)          /*定义函数，输入 ID 号和顺序号，按照输入次序建立进程链表*/
{
    填补程序
}
void print (node *head)    /*输出链表*/
{
    填补程序
}
node *insert(node *head,node *news) /*将进程 news 插入到队列尾部*/
{
    填补程序
}
node *timeslice(node *head,int cpu_base_time)
/*模拟时间片轮转调度过程:队列首进程使用一个时间片的 CPU*/
{
    填补程序
}
void main()
{
    int count=0,cpu_base_time;
    node *p;
    printf("新建的进程控制表为:\nkey run_time message\n");
    p=creat();             /*输入进程控制表*/
    print(p);              /*输出原始进程控制表*/
    printf("CPU 运行的单位时间片 cpu_base_time 为:\n");
    scanf("%d",&cpu_base_time);
    while(p)               /*模拟按单位时间片进程逐个被调度并进入 CPU 运行的过程*/
    {
        填补程序
    }
}
```