# 第五章　实验二

## 模拟最短寻道时间优先 SSTF 算法

- **最短寻道时间优先 SSTF(Shortest Seek Time First)算法：**选择这样的进程，其要求访问的磁道与当前磁头所在的磁道距离最近，以使每次的寻道时间最短。
- **例：**假定有一个具有 200 个磁道（编号为 0~199）的移动头磁盘，若磁头的当前位置为 100 磁道，磁头正向磁道号增加方向移动。现有一个磁盘读写请求队列：55，58，39，18，90，160，150，38，184。若采用最短寻址时间优先调度算法，试计算平均寻道长度各为多少？

| （从 100 号磁道开始） | |
|---|---|
| 被访问的下一个磁道号 | 移动距离（磁道数） |
| 90 | 10 |
| 58 | 32 |
| 55 | 3 |
| 39 | 16 |
| 38 | 1 |
| 18 | 20 |
| 150 | 132 |
| 160 | 10 |
| 184 | 24 |
| 平均寻道长度：27.5 | |

- 按照要求分别输入：

**输入：**

```
输入磁盘柱面总数:
200
输入磁盘读写请求总数:
9
输入磁盘读写请求柱面号序列:
55 58 39 18 90 160 150 38 184
输入磁盘当前位置为:
100
```

- 输出如下结果：

**输出：**

```
依次访问的柱面号为:
   90   58   55   39   38   18  150  160  184
总的移动柱面次数为:248

平均移动次数为：27.56
Press any key to continue
```

- 使用 Micrsoft Visual Studio C++ 6.0 或 CodeBlocks 编程：程序 5_2_SSTF.cpp。完善如下程序代码：

```
#include <malloc.h>
#include<stdio.h>
#include<math.h>
```

```
typedef struct track{
        int column;
        struct track *next;
}node;
int location;              /*当前磁头位置*/
int sum_move;          /*磁头移动总磁道数*/
float ave_move;        /*磁头移动平均磁道数*/

node *found_SSTF(node *head)   /*找到离当前磁头最近的磁道，并从 head 中删除该结点*/
  { 填补程序 }

 void main()
    {   int i,num,disk_length;
        node *head,*p,*pre;
        printf("输入磁盘柱面总数:\n");
        scanf("%d",&disk_length);
        printf("输入磁盘读写请求总数:\n");
        scanf("%d",&num);
        printf("输入磁盘读写请求柱面号序列:\n");
        for(i=1;i<=num;i++)
        { 填补程序 }
        printf("输入磁盘当前位置为:\n");
        scanf("%d",&location);
        printf("\n 依次访问的柱面号为:\n");
        sum_move=0;
        for(i=1;i<=num;i++)
        { 填补程序 }
        ave_move=(float)sum_move/num;
        printf("\n 总的移动柱面次数为:%d\n ",sum_move);
        printf("\n 平均移动次数为：%.2f \n",ave_move);
}
```