作业

- 问题描述：有一个磁盘请求序列给出了程序的I/O对各个柱面上数据块请求的顺序，例如一个请求序列98，183，37，122，14，124，65，67，n=8，请求编号为1～n。如果磁头开始位于位置C（假设不在任何请求的位置，例如C为53）。试使用贪心法编程输出给定的磁盘请求序列的调度方案和磁头移动总数。

题目要求使用贪心法编程，所以本质上就是采用 SSTF 磁盘调度算法。

SSTF 调度算法内容如下：

SSTF 即最短寻道时间优先(Shortest Seek Time First)，该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，但这种调度算法却不能保证平均寻道时间最短。

源代码如下所示（具体思路和分析均已包含在注释中）：

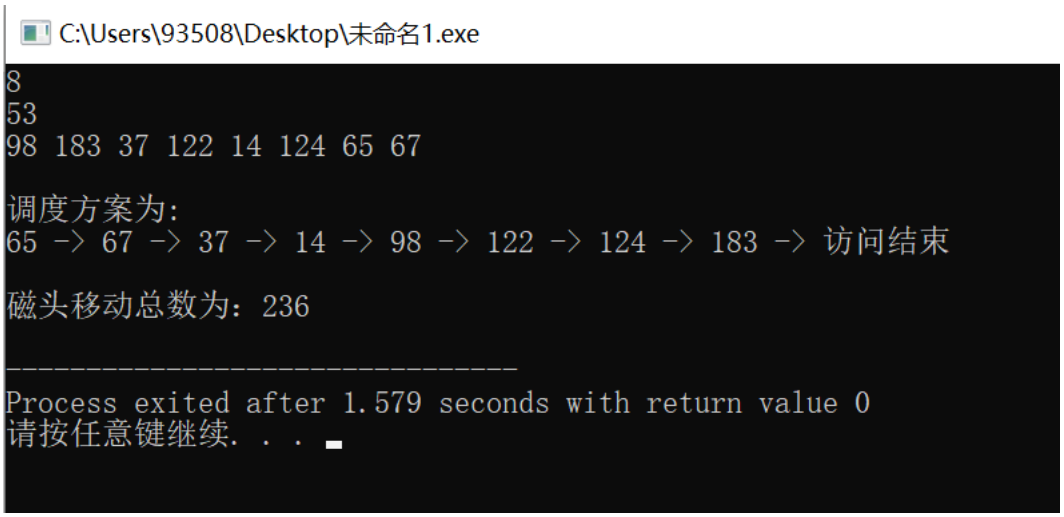
```
#include<iostream>
#include<math.h>
using namespace std;

int main()
{
    //设置磁道访问请求数量最多为1000
    const int N = 1000;
    //存储访问序列
    int queue[N];
    //存储当前位置到磁头处的距离
    int dis[N];
    //磁道访问请求数量
    int n = 0;
    //磁头起始位置
```

```
    int start = 0;
    //磁头移动总数
    int sum = 0;
    //输入磁道访问请求数量和磁头起始位置
    cin >> n >> start;
    //输入磁道访问序列并计算每个磁道与磁头初始位置的距离的绝对值
    for(int i = 0; i <= n-1 ; i++){
        cin >> queue[i];
        dis[i] = abs(queue[i] - start);
    }
    //循环 n 次，即访问 n 次磁道
    cout << endl << "调度方案为:" << endl;
    int times = n;
    while(times--){
        //min 为距离最小值，每次比较不断更新
        int min = 10000;
        //index 为离磁头距离最近的磁道的下标
        int index = 0;
        //从尚未访问过的磁道中找出距离当前磁头最近的下标和距离最小值并更新
        //若某磁道已访问过则会将其对应的 dis[i]置为-1 表示已访问过
        for(int i = 0; i <= n-1 ; i++){
            if(dis[i] < min && dis[i] != -1){
                index = i;
                min = dis[i];
            }
        }
        cout << queue[index] << " -> ";
        //更新磁头移动总数
        sum += dis[index];//更新 sum
        //若某磁道已访问过则会将其对应的 dis[i]置为-1 表示已访问过
        dis[index] = -1;
        //对磁头的位置进行移动（即更新）
        start = queue[index];
        //根据新的磁头位置对剩下的磁道访问请求的距离数组 dis 进行更新
        for(int i = 0; i <= n-1 ; i++){
            //若为-1 则代表已访问过，无需计算 dis
            if(dis[i] == -1){
                continue;
            }
            dis[i] = abs(queue[i] - start);
        }
    }
    cout << "访问结束" << endl;
    //输出磁头移动总数
```

```
    cout << endl << "磁头移动总数为: " << sum <<endl;
    return 0;
}
```

输出截图如下所示：

样例 1：



样例 2：