修理牧场

问题描述

农夫要修理牧场的一段栅栏，他测量了栅栏，发现需要N块木头，每块木头长度为整数*Li*个长度单位，于是他购买了一个很长的，能锯成N块的木头，即该木头的长度是*Li*的总和。

但是农夫自己没有锯子，请人锯木的酬金跟这段木头的长度成正比。为简单起见，不妨就设酬金等于所锯木头的长度。例如，要将长度为20的木头锯成长度为8，7和5的三段，第一次锯木头将木头锯成12和8，花费20；第二次锯木头将长度为12的木头锯成7和5花费12，总花费32元。如果第一次将木头锯成15和5，则第二次将木头锯成7和8，那么总的花费是35（大于32）

算法分析

分析可知，该问题即相当于带权路径长度，用优先队列模拟哈弗曼树可求得解。

功能实现及代码分析

1、输入判断

读取木头参数，对m，即木头长度进行合法性判断，若不符合要求，要求用户重新输入。

```
void read(priority_queue<int, vector<int>, greater<int> > &farmer, int n) {
    cout << "请依次输入木头长度：";
    int m = 0, count = 0;   //逐个读取木头长度
    while (count < n) {
        cin >> m;
        if (m >= 0) {
            farmer.push(m);
            count++;
        } else {            //不符合要求的输入
            cout << m << "不符合要求，  请重新输入" << endl;
        }
    }
}
```

2、求解

进入项目，获取参数并引导用户输入，用优先队列第1、2个项反复求和并入队列获得哈弗曼树。

```
int main() {      //带权路径长度，即哈夫曼树
    priority_queue<int, vector<int>, greater<int> > farmer;

    cout << "请输入n值：";
    int n = 0;    //n是要把木头锯成的数量
    cin >> n;

    read(farmer, n);

    int sum = 0;                        //优先队列模拟哈弗曼树
    while (farmer.size() > 1) {
        int first = farmer.top();
        farmer.pop();
        int second = farmer.top();
        farmer.pop();
        sum += first + second;          //每次取队列前2个，即最小的2个数相加
        farmer.push(first + second);    //和重新推入优先队列
    }

    cout << "最小花费是" << sum << endl;
    return 0;
}
```

用例演示

非法判断

```
请输入n值: 3
请依次输入木头长度: -1 -1 7
-1不符合要求,  请重新输入
-1不符合要求,  请重新输入
5 8
最小花费是32

Process finished with exit code 0
```

合法输入

```
请输入n值: 8
请依次输入木头长度: 4 5 1 2 1 3 1 1
最小花费是49

Process finished with exit code 0
```