项目说明文档

数据结构课程设计

——修理牧场

作 者 姓 名： 李佳诺

学 号： 1751188

指 导 教 师： 张 颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

# 目录

1 分析 2

分析 2

1.1 背景分析 2

1.2 功能分析 3

2 设计 3

2.1 算法设计 3

2.2 数据结构设计 4

3 实现 5

3.1 模拟构造Huffman树的函数huffmanForm（） 5

3.1.1 模拟构造Huffman树的流程图 5

3.1.2具体代码： 6

4 测试 7

4.1 功能性测试 7

4.2 出错性测试 7

# 1 分析

## 背景分析

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）.

## 功能分析

（1） 输入格式：输入第一行给出正整数N（N《104），表示要将木头锯成N块。第二行给出N个正整数，表示每块木头的长度。

（2） 输出格式：输出一个整数，即将木头锯成N块的最小花费。

# 设计

## 算法设计

本题显然是Huffman树的典型应用，是求最优的决策。

Huffman算法：

1.初始化： 根据给定的n个权值{w1,w2,…wn}构成n棵二叉树的集合F={T1,T2,..,Tn}，其中每棵二叉树Ti中只有一个带权wi的根结点，左右子树均空。

2. 找最小树：在F中选择两棵根结点权值最小的树作为左右子树构造一棵新的二叉树，且至新的二叉树的根结点的权值为其左右子树上根结点的权值之和。

3. 删除与加入：在F中删除这两棵树，并将新的二叉树加入F中。

4. 判断：重复前两步（2和3），直到F中只含有一棵树为止。该树即为哈夫曼树

## 数据结构设计

由于本题只涉及构造Huffman树，并且权值也较为单一，为此我们没有设计专门的类，但设计了通过vector模拟构造Huffman树的函数。里面存储的是该结点的权值（每块木头的花费）

//arr用来存储当前Huffman树各结点的权值

vector<int> arr;

# 实现

## 模拟构造Huffman树的函数huffmanForm（）

### 模拟构造Huffman树的流程图

将MIN1和MIN2的权值相加，入队

开 始

huffmanArr中寻找最小权值MIN1，出队

huffmanArr.size>1?

huffmanArr中寻找最小权值MIN2，出队

结 束

是

否

### 3.1.2具体代码：

int huffmanForm(vector<int> &huffmanArr)

{

int total = 0; //总花费

while (huffmanArr.size() > 1)

{

int min1 = huffmanArr[0];//权值最小的两个值

int pos1 = 0, pos2 = 0; //当前最小结点的下标

int tmp = 0; //保存当前两个最小结点的权值之和

for (int i = 1; i < huffmanArr.size(); i++)

{

if (huffmanArr[i] < min1)

{

min1 = huffmanArr[i];

pos1 = i;

}

}

tmp += min1;

//从huffmanArr中删除找出的最小权值的

huffmanArr.erase(huffmanArr.begin() + pos1);

int min2 = huffmanArr[0];//删除一个数之后的第一个元素

for (int i = 1; i < huffmanArr.size(); i++)

{

if (huffmanArr[i] < min2)

{

min2 = huffmanArr[i];

pos2 = i;

}

}

tmp += min2;

//从huffmanArr中删除找出的最小权值的

huffmanArr.erase(huffmanArr.begin() + pos2);

total += tmp;

//将两最小权值之和重新加入

huffmanArr.push\_back(tmp);

}

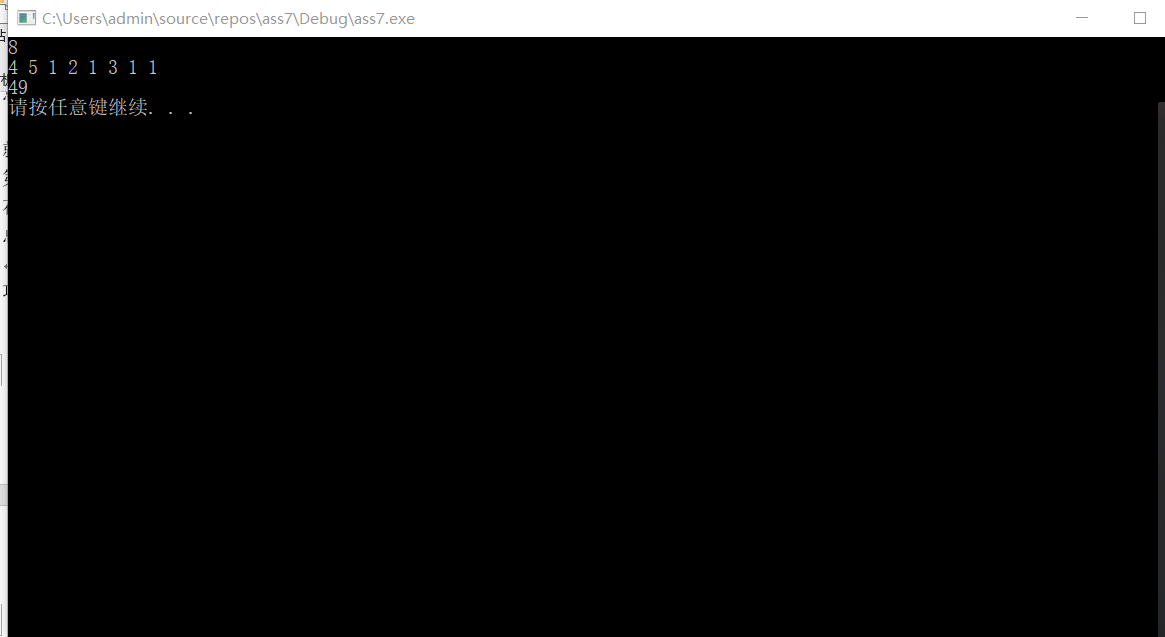
return total;

}

# 测试

## 功能性测试

本测试意在验证基本功能是否可以正常实现，在给定条件下输出最优解，即最小花费，测试结果如下：



## 出错性测试

本测试主要是测试程序是否有一定的健壮性，即可以防止因用户输入错误而导致程序崩溃，测试结果如下：

