项目说明文档

数据结构课程设计

——修理牧场

作 者 姓 名： 李圣

学 号： 1952723

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji Universit

目录

[1 分析 1](#_Toc27502)

[1.1 背景分析 1](#_Toc21791)

[1.2 功能分析 1](#_Toc25648)

[2 设计 1](#_Toc4540)

[2.1 文件结构设计 1](#_Toc31566)

[2.2 类结构设计 1](#_Toc8359)

[2.3 成员与函数设计 2](#_Toc8294)

[3 实现 6](#_Toc20475)

[3.1 输入（input）功能的实现 6](#_Toc31282)

[3.1.1 输入（input）功能流程图 6](#_Toc16468)

[3.1.2 输入（input）功能核心代码 6](#_Toc4232)

[3.2 计算结果（countPrice）功能的实现 7](#_Toc9552)

[3.2.1 计算结果（countPrice）功能流程图 7](#_Toc20287)

[3.2.2 计算结果（countPrice）功能核心代码 8](#_Toc28496)

[3.2.3 计算结果（countPrice）功能截屏示例 8](#_Toc29042)

[4 测试 8](#_Toc20540)

[4.1 鲁棒性测试 8](#_Toc8125)

[4.1.1 正常案例测试 8](#_Toc23371)

[4.1.2 输入错误测试 9](#_Toc8542)

# 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

该项目至少应该具有的基本功能：

1. 读取输入，格式如下：

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。

1. 输出结果，格式如下：

输出一个整数，即将木头锯成N块的最小花费。

# 设计

## 文件结构设计

表达式求解的过程中需要使用Vector，因此该项目包含两个文件，如下：

main.cpp

MyVector.h

其中main.cpp是主程序，且main.cpp调用了MyVector.h

## 类结构设计

考虑到功能需要，除了Vector相关的类以外，拟定设计以下类：

class Wood

其中Wood是树类

## 成员与函数设计

**MyVectorIterator类：**

template<typename val\_type>

using iterator = MyVectorIterator;

using pointer = val\_type\*;

using reference = val\_type&;

**私有成员：**

pointer \_ptr;

int \_pos;

**公有成员：**

MyVectorIterator(pointer ptr = nullptr, int pos = -1)；

~MyVectorIterator()；

int getPos()；

bool operator==(const iterator& it)const；

bool operator!=(const iterator& it)const；

iterator& operator++()；

iterator& operator++(int)；

iterator& operator--()；

iterator& operator--(int)；

reference operator\*()；

pointer operator->()；

**MyVector类：**

template<typename val\_type>

using vector = MyVector<val\_type>;

using reference = val\_type&;

using iterator = MyVectorIterator<val\_type>;

**私有成员：**

int \_size;

int \_capacity;

val\_type\* \_data;

**公有成员：**

MyVector(int capacity = DEFAULT\_CAPACITY)；

MyVector(const vector& vec)；

~MyVector()；

bool empty()；

int size()；

int length()；

int capacity()；

void resize(int newcapacity)；

void insert(iterator pos, const val\_type& val)；

val\_type erase(iterator pos)；

void push\_back(val\_type val)；

val\_type pop\_back()；

vector& operator=(vector& vec)；

vector& operator=(vector&& vec)；

iterator find(val\_type val)；

reference at(int pos)；

reference operator[](int pos)；

reference front()；

reference back()；

iterator begin()；

iterator end()；

void clear()；

**Wood类:**

**私有成员：**

MyVector<int> \_wood;

    int \_price;

void sortBack();

**公有成员：**

Wood();

void input();

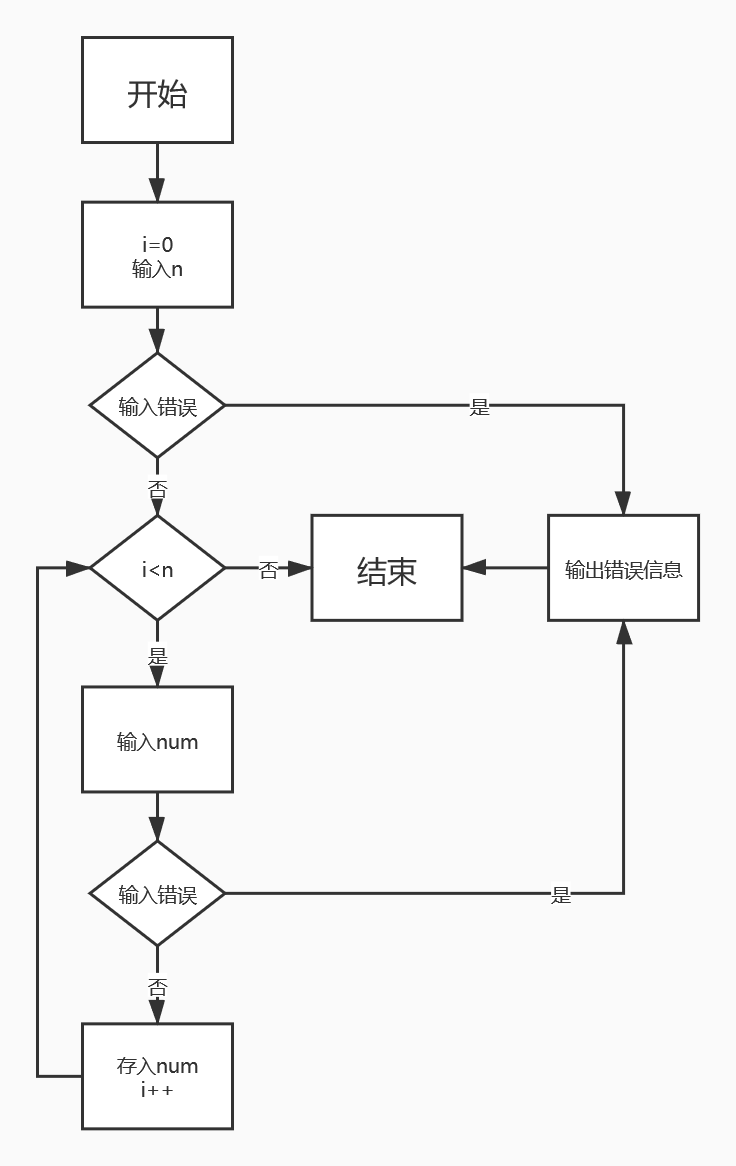
void countPrice();

int getPrice();

# 实现

## 输入（input）功能的实现

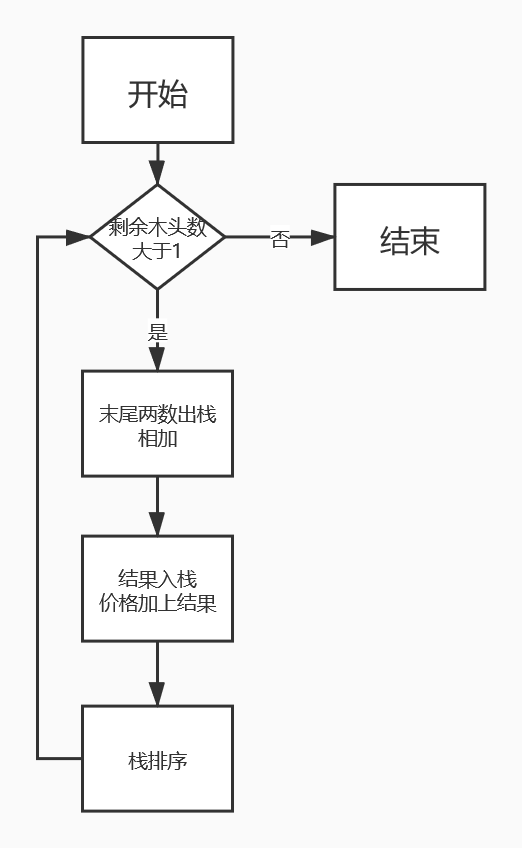
### 输入（input）功能流程图



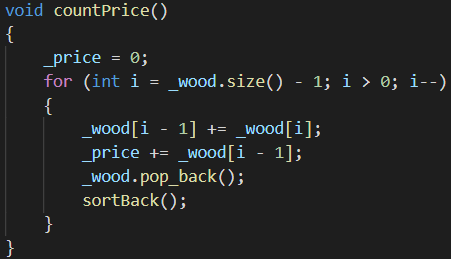
### 输入（input）功能核心代码

## 计算结果（countPrice）功能的实现

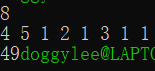
### 计算结果（countPrice）功能流程图



### 计算结果（countPrice）功能核心代码



### 计算结果（countPrice）功能截屏示例



# 测试

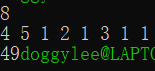
## 鲁棒性测试

### 正常案例测试

**测试用例：**一般情况

**预期结果：**对应的正确结果。

**实验结果：**



**测试用例：**1根木头的情况

**预期结果：**对应的正确结果。

**实验结果：**



### 输入错误测试

**测试用例：**输入字符的情况

**预期结果：**给出提示，程序终止

**实验结果：**





**测试用例：**输入不为正整数的情况

**预期结果：**给出提示，程序终止

**实验结果：**



