项目说明文档

数据结构课程设计

——勇闯迷宫游戏

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](#_Toc25907)

[1.1 功能分析 1](#_Toc24270)

[2 设计 1](#_Toc13065)

[2.1 文件结构设计 1](#_Toc28389)

[2.2 数据结构设计 1](#_Toc24214)

[2.3 类结构设计 1](#_Toc32487)

[2.4 成员与函数设计 1](#_Toc1084)

[3 实现 7](#_Toc25618)

[3.1 寻路功能的实现 7](#_Toc19712)

[3.1.1 寻路功能流程图 7](#_Toc12564)

[3.1.2 寻路功能核心代码 8](#_Toc16300)

[3.1.3 寻路功能截屏示例 8](#_Toc23835)

[3.2 总体系统的实现 9](#_Toc31768)

[3.2.1 总体系统流程图 9](#_Toc120)

[3.2.2 总体系统核心代码 10](#_Toc14785)

[3.2.3 总体系统截屏示例 10](#_Toc15196)

[4 测试 10](#_Toc27670)

[4.1 鲁棒性测试 10](#_Toc16246)

[4.1.1 正常案例测试 10](#_Toc16927)

[4.1.2 路径不存在测试 11](#_Toc31606)

# 分析

## 功能分析

拟在Linux系统内设计一个程序，根据题意至少应该具有以下基本功能：

1. 根据地图以及所给的起点终点进行寻路
2. 具有回溯功能，一条路不通时可以回到上一个地方
3. 寻路完成后能对路径进行输出
4. 未找到路径时进行提示

# 设计

## 文件结构设计

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

## 数据结构设计

考虑到只要Map够大，理论上Solution可以无限大，因此采用Vector进行存放。计划自行设计一个较为完备的Vector，以便该项目以及后续项目的使用。

## 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 Position

class Solution

class Map

其中Position是位置类，Solution类是解决路径类，Map类是迷宫地图类

## 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()；

**Position类：**

**私有成员：**

int \_x, \_y;

**公有成员：**

Position()；

Position(int x, int y) : \_x(x), \_y(y)；

bool isInside()；

Position operator+(const Position &p2) const；

bool operator==(const Position &p2) const；

friend ostream &operator<<(ostream &os, const Position &pos)；

**Solution类：**

**私有成员：**

MyVector<Position> \_solution;

    bool \_arrive;

**公有成员：**

Solution()；

void setArrive(bool arrive)；

bool arrive() const；

bool empty()；

void clear()；

Solution operator+(const Position &pos) const；

Solution operator+=(const Position &pos)；

Solution operator--()；

friend ostream &operator<<(ostream &os, Solution &solution)；

**Map类：**

**私有成员：**

char \_map[SIZE][SIZE];

Position \_entrance, \_exit;

Solution \_solution;

**公有成员：**

Map()；

Map(const char map[SIZE][SIZE])；

void setMap(const char map[SIZE][SIZE])；

void showMap()；

bool findSolution()；

char operator[](Position pos) const；

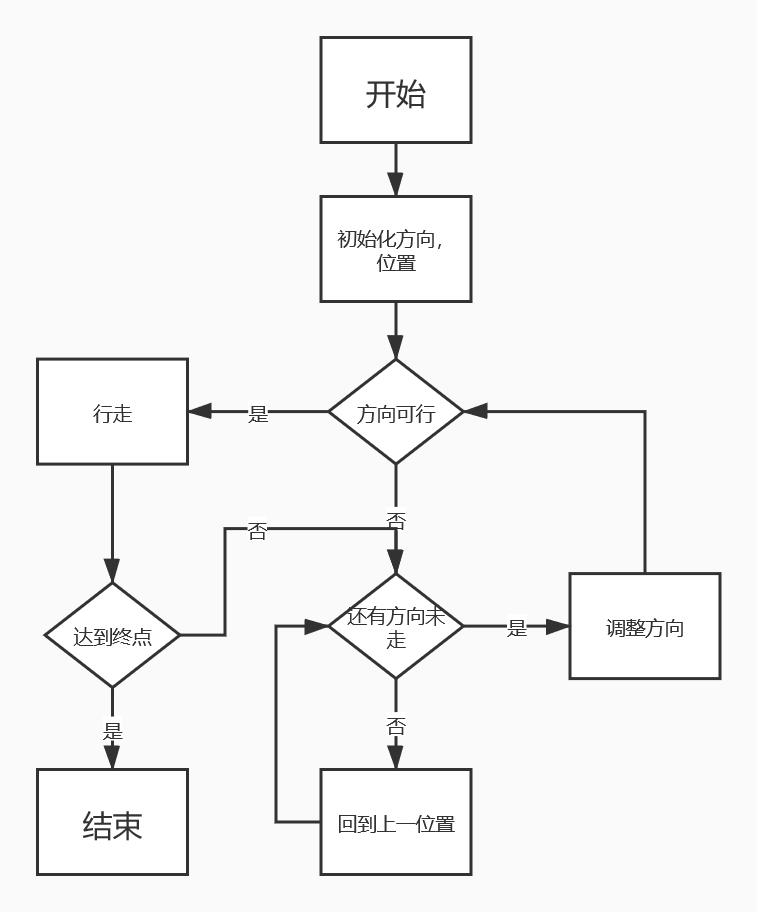
Solution findNextStep(Position pos, Solution solution, int lastdir) const；

void showSolution()；

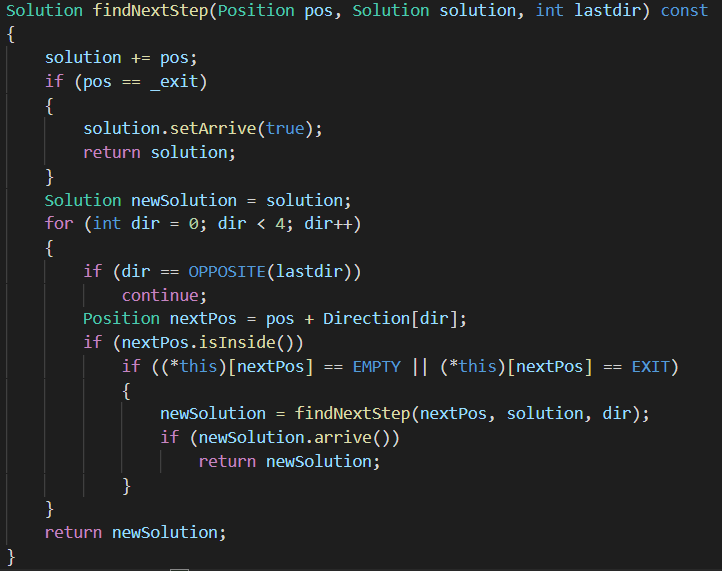
# 实现

## 寻路功能的实现

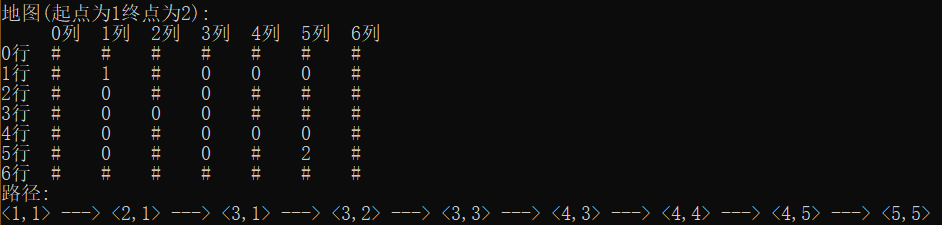
### 寻路功能流程图



### 寻路功能核心代码

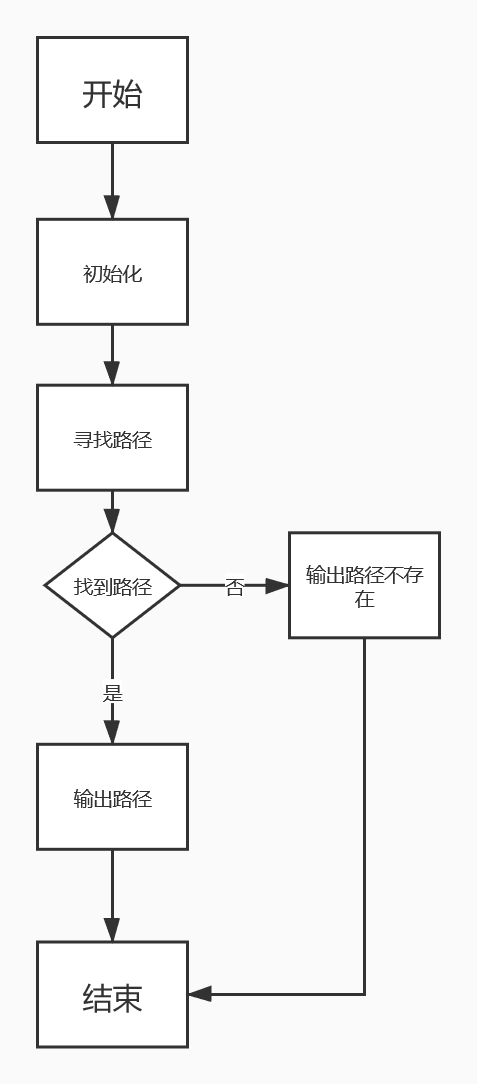


### 寻路功能截屏示例

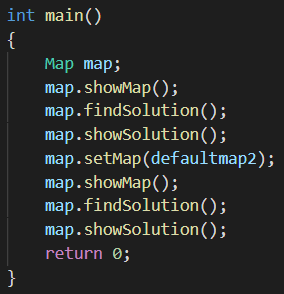


## 总体系统的实现

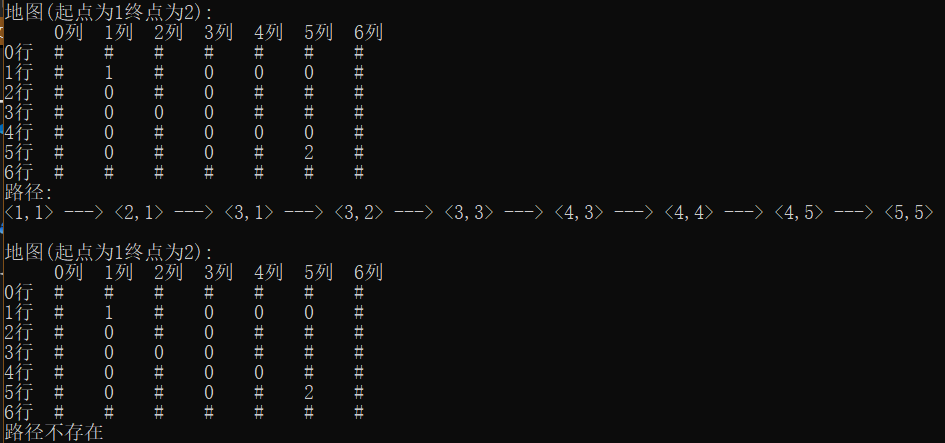
### 总体系统流程图



### 总体系统核心代码



### 总体系统截屏示例



# 测试

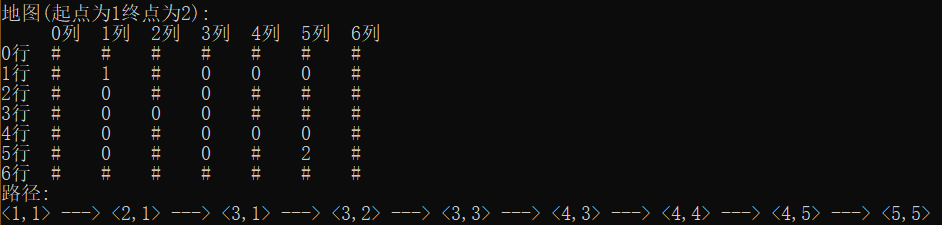
## 鲁棒性测试

### 正常案例测试

**测试用例：**正常案例。

**预期结果：**对应的正确结果。

**实验结果：**



### 路径不存在测试

**测试用例：**该地图不存在路径

**预期结果：**输出路径不存在

**实验结果：**

