**农夫过河问题**

1. 到现在这个阶段了，我们就讲解基础的部分了，我们直接上代码了呀

/\*\* 农夫过河问题

\*

\* 大家有没有看过我的C学习中的位算，我们用8位二进制的数字来表示任务的完成与否，1完成、0未完成

\* 现在我们用4个二进制数来代表

\* 第一位表示农夫（1：北岸，0：南岸）

\* 第二位表示狼 （1：北岸，0：南岸）

\* 第三位表示🐀𓰀 （1：北岸，0：南岸）

\* 第四位表示白菜（1：北岸，0：南岸）

\* 我们怎么判断农夫在哪（是不是把当前状态与1000进行 & 操作，为1就在北岸，为0 就在南岸）

\* 那么判断安全（羊的位置不等于狼，羊位置不等于白菜，且两个状态可以存在的是农夫在）

\* 那么接下去我们来进行广度优先的遍历，当满足条件时，我们就把新的状态push进队列中去

\* 且我们要记录路径，所以需要一个并查集数组，0-15即可

\*

\*/

#include <iostream>

#include <stdio.h>

#include "SqQueue.h"

#define BIT(x) (1<<x)

#include <vector>

#include <algorithm>

using namespace std;

//第一位农夫。第二位狼。第三位白菜，第四位白菜

enum Status{

FARMER = BIT(3), //农夫

WOLF = BIT(2), //狼

GOAT = BIT(1), //羊

CABBAGE = BIT(0) //白菜

};

//farmer的位置

int farmer(int location){

if((location&FARMER) != 0)

return 1;

return 0;

}

//wolf的位置

int wolf(int location){

if((location&WOLF) != 0)

return 1;

return 0;

}

//goat的位置

int goat(int location){

if((location&GOAT) != 0)

return 1;

return 0;

}

//cabbage的位置

int cabbage(int location){

if((location&CABBAGE) != 0)

return 1;

return 0;

}

int isSafe(int location){

if((wolf(location) == goat(location)) && (farmer(location) != goat(location)))

return 0;

if((cabbage(location) == goat(location)) && (farmer(location) != goat(location)))

return 0;

return 1;

}

void farmer\_location(int location){

if(farmer(location) == 1){

printf("从南岸到北岸\n");

}else{

printf("从北岸到南岸\n");

}

}

void farmer\_acrry(int currStatus, int afterAtatus){

int status = currStatus^afterAtatus;

switch(status){

case 12:

printf("把狼带过河\n");

break;

case 10:

printf("把羊带过河\n");

break;

case 9:

printf("把菜带过河\n");

break;

default:

printf("什么都不带\n");

break;

}

}

void farmer\_cross\_river(){

// std::vector<bool> visited(16, false);

SqQueue<int> q; //定义队列来存放新的状态

int arr[16]; //定义并查集，并初始值为-1

for(int i = 0; i < 16; i++){

arr[i] = -1;

}

int location; //int正好是4个字节

q.push(0);

arr[0] = 0; //第一个状态是默认的

int newLocation; //新的位置情况

// visited[0] = true;

while( (!q.isEmpty()) && (arr[15] == -1)){ //队列不为空，且位置还没终止的位置

location = q.front();

q.pop();

//每次农夫可以带的东西有自己、狼、羊、白菜

for(int allStatus = 1; allStatus <= 8; allStatus <<= 1){

//但是，带的时候需要的是在同一边

if((farmer(location)!=0) == ((allStatus&location)!=0)){

//那么每次要带什么呢,可以一个人走，有也可以带东西

newLocation = location^( FARMER|allStatus ); // 渡河后的情况

if(isSafe(newLocation) && (arr[newLocation]==-1)){

cout <<"newLocation : ";

cout << newLocation << endl;

q.push(newLocation); //符合条件，推进去队列中去

arr[newLocation] = location; //更新并查集

// visited(newLocation) = true;

}

}

} //for end

} //while end

//

// for(int i = 0; i < 16; i++){

// cout << arr[i] << " " << endl;

// }

//现在从1111的位置通过并查集来推到0000的位置来推出路径

int res[16] = {0};

int counter = 0;

if(arr[15] == -1){

printf("no solution");

}else{

for(int status = 15; status !=0; status = arr[status]){

res[counter++] = status;

}

//打印一下状态

for(int i = 0; i < counter; i++){

cout << res[i] << " " << endl;

}

for(int i = 0; i < counter; i++){

int currStatus= res[i]; //第一次状态

// res.pop\_back();

int afterStatus = res[i+1]; //第二次状态

// res.pop\_back();

//然后我们进行对比一下

farmer\_location(currStatus);

farmer\_acrry(currStatus, afterStatus);

}

}

}

int main(){

farmer\_cross\_river();

return 0;

}