Министерство Образования и Науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего профессионального образования

**“Санкт-Петербургский государственный электротехнический университет “ЛЭТИ” им. В.И. Ульянова (Ленина)”**

**(СПбГЭТУ)**

ФКТИ, кафедра МОЭВМ

**Отчет**

**по лабораторной работе №3**

**по дисциплине «Проектирование и анализ алгоритмов»**

Выполнила: Студентка группы 1381

Шнейдер Екатерина

Проверил: Казаков Б.Б.

г. Санкт-Петербург

2013г.

**Цель работы**

Изучить метод ветвей и границ. Решить поставленную задачу.

**Задание**

Вдоль прямой дороги расположены деревни. Дорога представляется целочисленной осью, а расположение каждой деревни задается целым числом – координатой на этой оси. Никакие две деревни не имеют одинаковых координат. Расстояние между двумя деревнями вычисляется как модуль разности их координат.

В некоторых, не обязательно во всех, деревнях будут построены почтовые отделения. Деревня и расположенное в ней почтовое отделение имеют одинаковые координаты. Почтовые отделения необходимо расположить в деревнях таким образом, чтобы общая сумма расстояний от каждой деревни до ближайшего к ней почтового отделения была минимальной.

Требуется по заданным координатам деревень и количеству почтовых отделений найти такое их расположение по деревням, при котором общая сумма расстояний от каждой деревни до её ближайшего почтового отделения будет минимальной.

**Входные данные:** В первой строке содержатся два целых числа: первое число — количество деревень, второе число — количество почтовых отделений.

Вторая строка содержит целые числа, являющихся координатами деревень.

**Выходные данные:** Первая строка содержит одно целое число — общую сумму расстояний от каждой деревни до её ближайшего почтового отделения для расположения почтовых отделений, описанного во второй строке.

Вторая строка содержит целые числа в возрастающем порядке.

Эти числа являются искомыми координатами почтовых отделений.

Если для заданного расположения деревень есть несколько решений, то программа должна найти одно из них.

Пример входных данных

10 5

1 2 3 6 7 9 11 22 44 50

Соответствующие выходные данные

9

2 7 22 44 50

**Описание алгоритма и его оценка**

При размещении одного почтового отделения эта задача известна как задача Лео Мозера (1952 год). Ответ V Div 2+1 (V – нечетное число) и любая из двух средних деревень (V - четное).

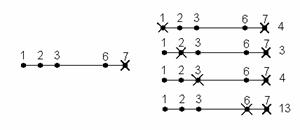
Решение задачи с одним почтовым отделением на взвешенном графе основано на использовании алгоритма Флойда и поиска вершины с минимальной суммой расстояний до других вершин.

При увеличении количества почтовых отделений эти идеи «не проходят». Переборные схемы также не работают при данных ограничениях.

Остается поиск динамической схемы. Рассмотрим пример, приведенный в формулировке задачи. Размещаем одно почтовое отделение в деревне с номером i.

Оценим расстояние до деревень, находящихся левее её.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 0 | 1 | 3 | 12 | 16 | 26 | 38 | 115 | 291 | 345 |



Что это дает? Пока ничего. Попытаемся решить задачу для двух почтовых отделений. Рассмотрим деревни с номерами от 1 до i. Одно почтовое отделение размещается в деревне с номером i. Найдем место второго почтового отделения, дающего минимальную сумму расстояний. Пример (i равно 5). Крестиками обозначены деревни с почтовыми отделениями. Справа на рисунке приведены суммы расстояний до ближайших почтовых отделений. Из четырех вариантов выбираем второй, дающий минимальную оценку - 3. Просчитаем для всех значений i. Массив оценок имеет вид.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| ¥ | 0 | 1 | 2 | 3 | 7 | 12 | 21 | 37 | 43 |

Символом ¥ обозначен бессмысленный случай – два почтовых отделения в одной деревне. Продолжим рассмотрение – три почтовых отделения. На рисунке приведен случай при i=7 (в 7 деревнях размещаются 3 почтовых отделения). Первый справа отрезок характеризует вариант, при котором два почтовых отделения расположены в деревнях с 1 по 6. Он подсчитан (равен 7, третье отделение в седьмой деревне, итоговая оценка не изменяется и равна 7), мы почти «нащупали» динамическую схему. Второй отрезок – два почтовых отделения в деревнях с 1 по 5. Из предыдущего рассмотрения стоимость этого случая равна 3, расстояние от 6 деревни до ближайшего почтового отделения 2, итого 5.

Окончательные оценки при 3, 4 и 5 отделениях приведены в таблице.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| ¥ | ¥ | 0 | 1 | 2 | 3 | 5 | 9 | 21 | 27 |
| ¥ | ¥ | ¥ | 0 | 1 | 2 | 3 | 5 | 9 | 15 |
| ¥ | ¥ | ¥ | ¥ | 0 | 1 | 2 | 3 | 5 | 9 |

Осталось сделать последний шаг. Оценка расстояний в нашей схеме проводилась до деревень, расположенных слева. К этим оценкам следует прибавить сумму расстояний до деревень, находящихся справа. Окончательные оценки.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| ¥ | ¥ | ¥ | ¥ | 101 | 92 | 85 | 53 | 11 | 9 |

Итак, общая сумма расстояний от каждой деревни до её ближайшего почтового отделения равна 9. Это минимум в окончательном массиве оценок. Как зафиксировать в этой схеме деревни с почтовыми отделениями? Как обычно в алгоритмах типа Флойда, при улучшении оценки, запоминать соответствующую ситуацию.

**Описание и ограничения программы**

1. Исходные данные вводятся из файла или с клавиатуры.
2. Файл должен иметь следующую структуру:

10 5

1 2 3 6 7 9 11 22 44 50

где, в первой строке содержатся два целых числа: первое число — количество деревень, второе число — количество почтовых отделений. Вторая строка содержит целые числа, являющихся координатами деревень

1. Выходные данные выводятся на экран и имеют следующую структуру:

9

2 7 22 44 50

1. Количество чисел во второй строке и первое число в первой должны совпадать.
2. Не должно быть одинаковых чисел во второй строке.

**Исходный код программы**

Файл Menu.h

#ifndef \_menu\_h\_

#define \_menu\_h\_

class Menu

{

private:

int var\_menu;

public:

void print\_in\_menu();

void print\_out\_menu();

void error\_message();

int getVar\_menu();

};

#endif

Файл Menu.cpp

#include <iostream>

#include "Menu.h"

using namespace std;

void Menu::print\_in\_menu()

{

cout << "1. Ввод из файла" << endl;

cout << "2. Ввод с клавиатуры" << endl << endl << endl << endl;

cout << "Введите номер операции ";

cin >> this->var\_menu;

}

void Menu::print\_out\_menu()

{

cout << "1. Сохранение в файл" << endl;

cout << "2. Вывод на экран" << endl << endl << endl << endl;

cout << "Введите номер операции ";

cin >> this->var\_menu;

}

int Menu::getVar\_menu()

{

return this->var\_menu;

}

void Menu::error\_message()

{

cout << "Ошибка ввода!" << endl;

}

Файл Village.h

#ifndef \_village\_h\_

#define \_village\_h\_

class Village

{

private:

int count\_vil;

int count\_post;

int \*coordinates;

int result\_size;

int \*result\_coordinats;

void calculation();

public:

bool from\_file();

bool from\_keyboard();

void main\_function();

void save\_to\_file();

void print\_on\_screen();

};

#endif

Файл Village.cpp

#include "Village.h"

#include <fstream>

#include <iostream>

#include <limits.h>

#include <math.h>

#include <cmath>

using namespace std;

int compare (const void \* a, const void \* b){

return ( \*(int\*)a - \*(int\*)b );

}

bool consist\_mass(int \*massiv, int size, int element)

{

for(int i = 0; i < size; i ++)

{

if(massiv[i] == element)

return true;

}

return false;

}

bool Village::from\_file()

{

char filename[100];

int i;

cout << "Введите название файла и путь к нему ";

cin >> filename;

ifstream in(filename);

in >> this->count\_vil;

in >> this->count\_post;

this->coordinates = new int [this->count\_vil];

for(i = 0; !in.eof() ; i ++)

{

in >> this->coordinates[i];

if(i == this->count\_vil - 1)

{

i ++;

break;

}

}

if(i < this->count\_vil)

return false;

if(this->count\_vil > 0&&this->count\_post > 0&&this->count\_vil >= this>count\_post)

{

qsort(this->coordinates, this->count\_vil, sizeof(int), compare);

for(i = 1; i < this->count\_vil; i ++)

if(this->coordinates[i] == this->coordinates[i - 1])

return false;

return true;

}

return false;

}

bool Village::from\_keyboard()

{

int i;

cout << "Введите количество деревень ";

cin >> this->count\_vil;

cout << "Введите количество почтовых отделений ";

cin >> this->count\_post;

cout << "Введите координаты деревень " << endl;

this->coordinates = new int [this->count\_vil];

for(i = 0; i < this->count\_vil; i ++)

cin >> this->coordinates[i];

if(this->count\_vil > 0&&this->count\_post > 0&&this->count\_vil >= this>count\_post)

{

qsort(this->coordinates, this->count\_vil, sizeof(int), compare);

for(i = 1; i < this->count\_vil; i ++)

if(this->coordinates[i] == this->coordinates[i - 1])

return false;

return true;

}

return false;

}

void Village::main\_function()

{

int i, j, k, l, min, line;

int \*resulting\_index = new int[this->count\_post];

int \*mass1 = new int[this->count\_vil];

int \*mass2 = new int[this->count\_post];

this->result\_coordinats = new int[this->count\_post];

for(k = 0; k < this->count\_post; k ++)

{

min = INT\_MAX;

for(i = 0; i < this->count\_vil; i ++)

{

if(!consist\_mass(resulting\_index, k+1, i))

{

mass1[i] = 0;

for(j = 0; j < this->count\_vil; j ++)

{

for(l = 0; l < k; l ++)

mass2[l] = abs(this->coordinates[j] –

this-> coordinates[resulting\_index[l]]);

if(k > 0)

qsort(mass2, k, sizeof(int), compare);

else

mass2[0] = INT\_MAX;

if(mass2[0] > abs(this->coordinates[i] –

this->coordinates[j]))

mass1[i] += abs(this->coordinates[i] –

this->coordinates[j]);

else

mass1[i] += mass2[0];

}

if(mass1[i] < min)

{

min = mass1[i];

resulting\_index[k] = i;

}

}

}

cout << "\n\n\n\n============Шаг" << k + 1 << "===============" <<endl;

cout << "\====Строим " << k + 1 << "-ое почтовое отделение=======" <<endl;

for(j = 0; j < this->count\_vil; j ++)

{

if(resulting\_index[k] == j)

cout << ">>" <<mass1[j] << "<<" << " ";

else

cout << mass1[j] << " ";

}

cout << endl << "Почтовое отделение <строим> в деревне с координатой"<<

this->coordinates[resulting\_index[k]];

}

cout << "\n\nНаимеьшая сумма расстояний от каждой деревни до её ближайшего

почтового отделения выделена >> <<" << endl << endl << endl;

for(i = 0; i < this->count\_post; i ++)

this->result\_coordinats[i] = this->coordinates[resulting\_index[i]];

cout << endl;

this->calculation();

system("Pause");

}

void Village::calculation()

{

int i, j, min;

this->result\_size = 0;

for(i = 0; i < this->count\_vil; i ++)

{

min = INT\_MAX;

for(j = 0; j < this->count\_post; j ++)

{

if(abs(this->coordinates[i] - this->result\_coordinats[j]) <min)

min = abs(this->coordinates[i] - this->result\_coordinats[j]);

}

this->result\_size += min;

}

}

void Village::save\_to\_file()

{

char filename[100];

cout << "Введите название файла и путь к нему для сохранения результатов ";

cin >> filename;

ofstream out(filename);

out << this->result\_size;

for(int i = 0; i < this->count\_post; i ++)

out << this->result\_coordinats[i] << " ";

}

void Village::print\_on\_screen()

{

cout << "Сумма расстояний от каждой деревни до её ближайшего почтового отделения ";

cout << this->result\_size << endl;

cout << "Координаты почтовых отделений " << endl;

for(int i = 0; i < this->count\_post; i ++)

cout << this->result\_coordinats[i] << " ";

cout << endl;

}

Файл Main.cpp

#include <iostream>

#include "Menu.h"

#include "Village.h"

int main()

{

setlocale(0, "Rus");

Menu menu;

Village village;

bool check = false;

menu.print\_in\_menu();

if(menu.getVar\_menu() == 1)

check = village.from\_file();

if(menu.getVar\_menu() == 2)

check = village.from\_keyboard();

if(check)

{

village.main\_function();

menu.print\_out\_menu();

if(menu.getVar\_menu() == 1)

village.save\_to\_file();

if(menu.getVar\_menu() == 2)

village.print\_on\_screen();

}

else

menu.error\_message();

system("Pause");

return 0;

}

**Вывод**

При выполнении лабораторной работы были получены практические навыки работы с использованием метода ветвей и границ. Была составлена программа, решающая задачу про почтовые отделения. Ошибок в её работе не обнаружено.