Лабораторная работа №4
Оценка сложности эвристических алгоритмов

1Цель работы

1.1 Научиться реализовывать и оценивать сложность эвристических алгоритмов на C#.

2Литература

2.1 Фленов, М.Е. Библия C#. – 3 изд. – Санкт-Петербург: БХВ-Петербург, 2016. – URL: https://ibooks.ru/bookshelf/353561/reading. – Режим доступа: для зарегистрир. пользователей. – Текст : электронный. – гл.8.

3Подготовка к работе

3.1 Повторить теоретический материал (см. п.2).

3.2 Изучить описание лабораторной работы.

4Основное оборудование

4.1 Персональный компьютер.

5Задание

5.1 Заполнение массива

Объявить двумерный массив из N строк и M столбцов (N и M вводятся пользователем). Заполнить его значениями -1.

В ячейку с координатами x,y записать значение 0. Координаты вводятся пользователем.

Добавить в несколько случайных несколько ячеек препятствия (записать в них -2).

В случайную ячейку массива записать значение 99 (это финишная ячейка).

Вывести на экран полученный массив в виде таблицы.

5.2 Поиск длины пути

Определить, используя волновой алгоритм, за сколько шагов можно достичь ячейку со значением 99 из ячейки x,y в массиве из п.5.1. Считать, что перемещаться можно только вверх-вниз и влево-вправо, перемещение по диагонали запрещено.

Вывести на экран массив, в котором отображается, и за сколько шагов можно дойти от исходной ячейки до остальных (пока не найдена ячейка со значением 99).

Пример:

9	10	■	10	9	8	9	10	-1	-1	-1	-1
8	9	■	9	8	7	8	9	10	-1	-1	-1
7	8	9	8	7	6	7	8	9	10	-1	-1
6	7	8	7	6	5	6	7	■	■	10	-1
5	■	■	■	■	4	5	6	7	8	9	10
4	3	2	1	2	3	4	5	6	■	■	-1
3	2	1	0	1	2	3	4	5	■	■	10
4	3	2	1	2	3	4	5	6	7	8	9

Порядок обхода вершин определяется эвристической функцией «расстояние + стоимость» (обычно обозначаемой как f(x)). Эта функция — сумма двух других: функции стоимости достижения рассматриваемой вершины (x) из начальной (обычно обозначается как g(x) и может быть как эвристической, так и нет), и функции эвристической оценки расстояния от рассматриваемой вершины к конечной (обозначается как h(x)).

Функция h(x) должна быть допустимой эвристической оценкой, то есть не должна переоценивать расстояния к целевой вершине. Например, для задачи маршрутизации h(x) может представлять собой расстояние до цели по прямой линии, так как это физически наименьшее возможное расстояние между двумя точками.

Алгоритм поиска длины пути:
d = 0 (d – длина пути от стартовой ячейки)

do
из каждой ячейки со значением d зайти в соседние ячейки (над ней, под ней, слева и справа) и изменить их значение на d+1.
после обхода всех соседних значений изменить значение d: d = d+1
while (финишная ячейка не помечена) И (есть возможность распространения волны)

Вывести на экран значение d, если достигнута финишная ячейка, иначе — сообщить, что путь не найден.

5.3 Восстановление пути.

Определить, используя волновой алгоритм, за сколько шагов можно достичь ячейку со значением 99 из ячейки x,y в массиве из п.5.2. Считать, что перемещаться можно только вверх-вниз и влево-вправо, перемещение по диагонали запрещено.

Алгоритм восстановления пути:

```
ЕСЛИ финишная ячейка помечена
ТО
  перейти в финишную ячейку
  ЦИКЛ
    выбрать среди соседних ячейку, помеченную числом
    на 1 меньше числа в текущей ячейке;
    перейти в выбранную ячейку и добавить её к пути
  ПОКА текущая ячейка — не стартовая
  ВОЗВРАТ путь найден
ИНАЧЕ
  ВОЗВРАТ путь не найден
```

6Порядок выполнения работы

6.1 Запустить MS Visual Studio и создать консольное приложение C# (Console Application (.Net Framework)) с названием LabWork4.

6.2 Выполнить все задания из п.5 в проекте LabWork4.

При выполнении заданий использовать минимально возможное количество команд и переменных и выполнять форматирование и рефакторинг кода.

6.3 Ответить на контрольные вопросы.

7Содержание отчета

7.1 Титульный лист
7.2 Цель работы
7.3 Ответы на контрольные вопросы
7.4 Вывод

8Контрольные вопросы

8.1 Что такое «многомерный массив»?
8.2 Как описывается двумерный массив?
8.3 Как обратиться к некоторому элементу двумерного массива?
8.4 Как узнать количество строк двумерного массива?
8.5 Как узнать количество столбцов двумерного массива?
8.6 Как вывести двумерный массив на консоль в виде таблицы?

9Приложение

9.1 Многомерные массивы в C#

Многомерный массив — это массив, который отличается двумя или более измерениями.

Доступ к каждому элементу массива осуществляется с помощью определенной комбинации двух или более индексов. Многомерный массив индексируется двумя и более целыми числами.

Двумерные массивы

Простейшей формой многомерного массива является двумерный массив. Местоположение любого элемента в двумерном массиве обозначается двумя индексами. Такой массив можно представить в виде таблицы, на строки которой указывает один индекс, а на столбцы — другой.

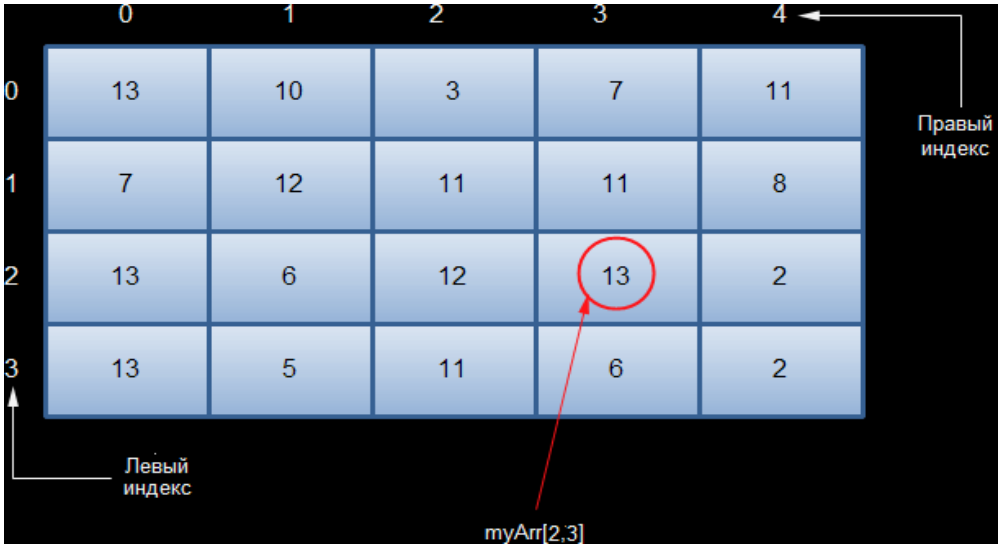
Пример объявления и инициализации двумерного массива показан ниже:

```
// Объявляем двумерный массив из 4 строк и 5 столбцов
int[,] arr = new int[4, 5];

Random random = new Random();

// Инициализируем данный массив
for (int i = 0; i < arr.GetLength(0); i++) // GetLength(0) – количество строк массива
{
   for (int j = 0; j < arr.GetLength(1); j++) // GetLength(1) – количество столбцов массива
   {
     myArr[i, j] = random.Next(1, 15);
     Console.Write("{0}\t", arr[i, j]);
   }
   Console.WriteLine();
}
```

Схематическое представление массива arr показано ниже:



Для инициализации двумерного массива достаточно заключить в фигурные скобки список инициализаторов каждого его размера:

```
тип[,] имя_массива =
{
   {val00, val01, val02, ..., val0N},
   {val10, val11, val12, ..., val1N},
   {val20, val21, val22, ..., val2N},
   ...
};
```

Ниже приведен пример инициализации двумерного массива:

```
int[,] arr =
{
    {1, 10},
    {2, 20},
    {3, 30},
    {4, 40}
};
```

9.2 Волновой алгоритм

Алгоритм волновой трассировки (волновой алгоритм, алгоритм Ли) — алгоритм поиска пути, алгоритм поиска кратчайшего пути на планарном графе. Принадлежит к алгоритмам, основанным на методах поиска в ширину.