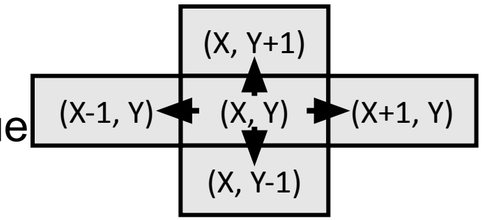
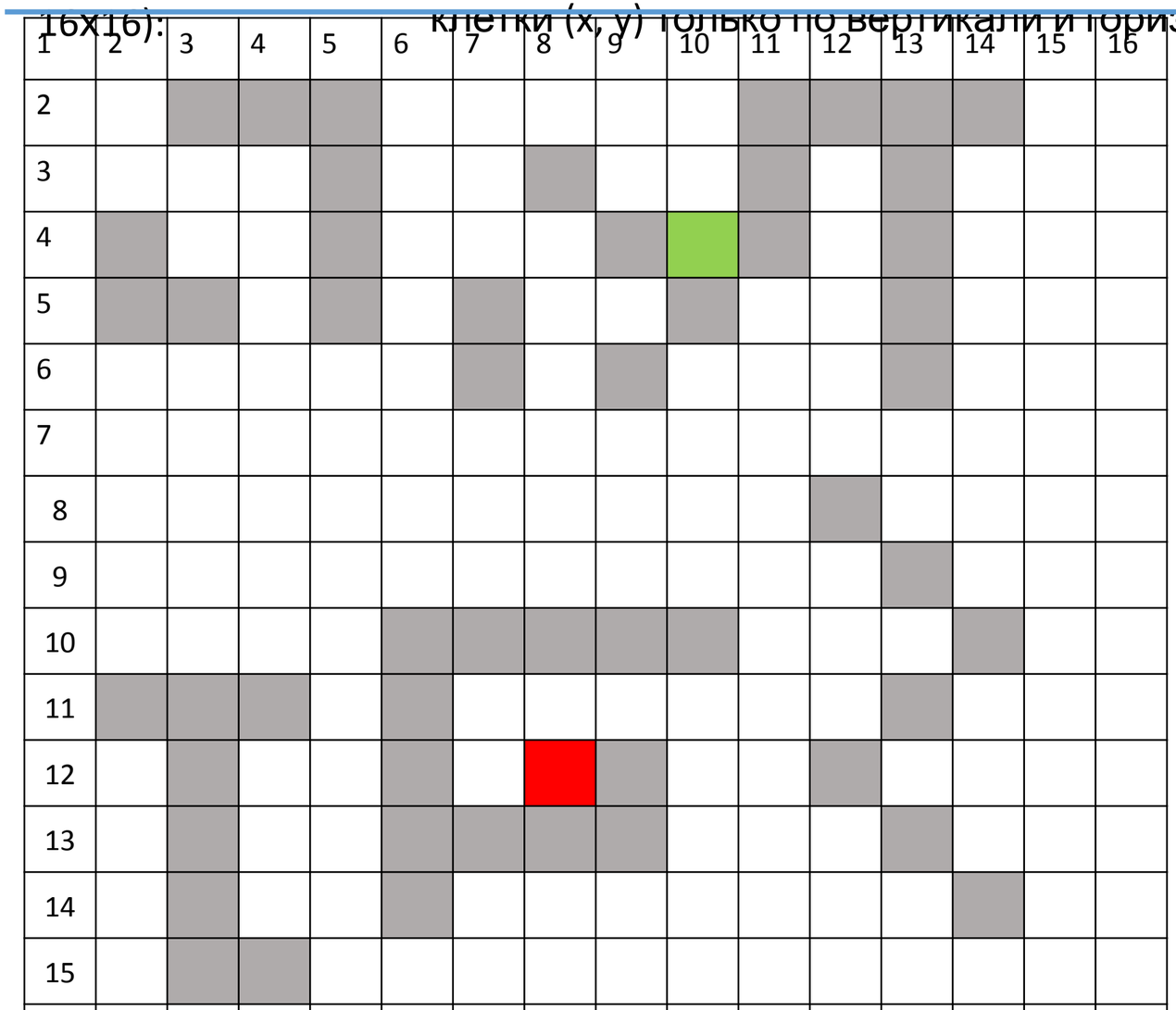
# Принципы динамического программирования на примере задачи поиска кратчайшего пути на клеточном поле

**Постановка задачи:** Дано ограниченное по размерам прямоугольной формы со стенками, где возможно перемещение из такого поля (двумерный массив 16x16):




клетки (x, y) только по вертикали и горизонтали:




Здесь серые клетки – элементы стен,
Красная клетка – стартовое поле,
Зеленая клетка – целевое поле

Каждую клетку удобно представить как запись со следующими компонентами:

1) «Вес» клетки – показывает прибавку к пути, которую дает эта клетка;

2,3) Координаты X и Y клетки из которой мы попали в данную клетку;

4) Маркер стены – если 1 – через клетку идти нельзя, а если 0, то можно

5) Общая длина пути при достижении данной клетки (в клетку идем тогда, когда новый путь короче предыдущего)

При решении задачи удобно увеличить размер массива, окружив поле стенками (все размерности массива увелим на 2)

Начальную длину пути к каждой клетке удобно задать любым большим числом

*Такая структура позволяет решить*

# Подсказки к ДЗ:

1) Структура данных (возможна оптимизация):

```
type pp=record
  x,y:integer;//координаты предшественника
  v:integer;//"вес" клетки
  f:integer;//флаг: стенка: f=-1, обычное поле: f=0
  s:integer;//путь до клетки (сумма весов по маршруту)
end;
var pole:array [,] of pp;//двумерный массив записей
    xs,ys,xe,ye,m,n:integer;
```

2) Описание модуля **Rread** - процедуры подготовки данных (возможна оптимизация):

```
//открываем файл для чтения;
//читаем из него m,n - количество строк и столбцов поля
setlength(pole,m+2,n+2);//здесь строки поля 1..m, а колонки 1..n. Остальное - стенка
//читаем из файла xs,ys - координаты стартовой клетки (начало координат: [1,1])
//читаем xe,ye - координаты конечной клетки (здесь x-строка, считая сверху, а y-колонка)
//читаем в цикле по строкам от 1 до m и столбцам от 1 до n из файла
//веса v всех клеток, и заполняем массив pole, учитывая, что:
//     начальное значение S задаем равным большому числу
//     если v=-1, то это стенка, задаем f=1, иначе f=0
//для нулевой и m+1 - й строки задаем f=1 (стенка)
//для нулевого и n+1 - й колонки задаем f=1 (стенка)
//для клетки (xs,ys) (стартовой) задаем s=0
```

3) Описание модуля **Pathfinder(I,j)** - процедуры (или функции?) поиска кратчайшего пути из (xs, ys) в (xe, ye):

Пусть (i, j) – текущая клетка, а соответствующий ей элемент массива: pole[I, j].

Если клетка [i+1, j] - не стенка и pole[i, j].s + pole[i+1, j].v<pole[i+1, j].s, тогда:

a) pole[i+1, j].s=pole[i, j].s + pole[i+1, j].v; pole[i+1, j].x:=i; pole[i+1, j].y:=j; b) Запускаем Pathfinder(i+1, j)

… и то же самое для остальных направлений

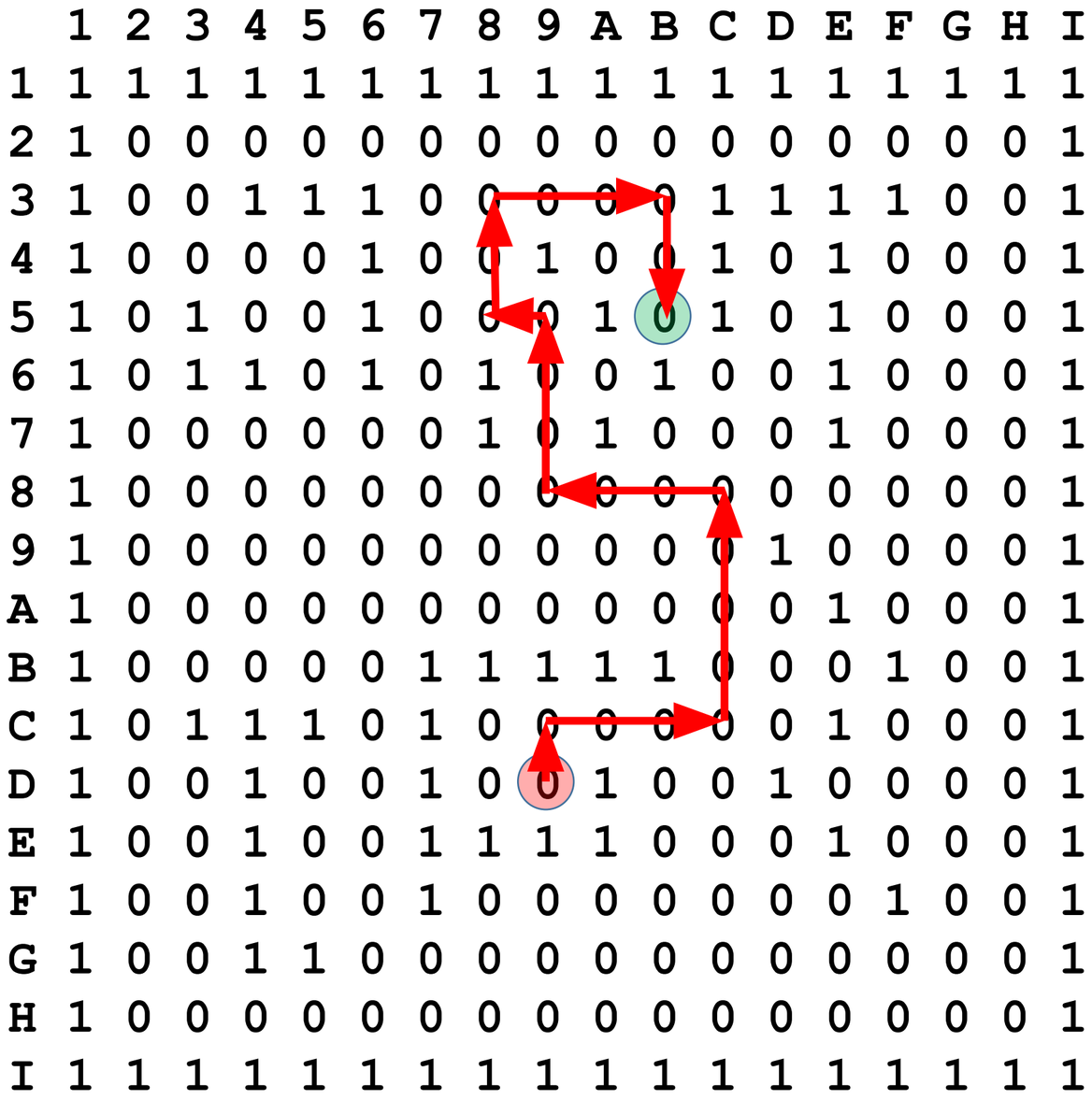
4) Текст (почти полный) головной программы:

```
begin
  Rread; Pathfinder(xs,ys);
  Println('Длина кратчайшего пути из (',xs,', ',ys,') в (', xe,', ',ye,'):',pole[xe,
ye].s);
  … Далее идет вывод кратчайшего пути в форме: (xs,ys)->, …, ->(xe, ye)
End.
```

**Раз файл с выполненным заданием выслать в мой E-mail адрес: LEOMTL@MAIL.RU, прикрепив к письму и указав в «Теме»**

16 16  - строк и столбцов
12 8   - строка и столбец стартовой клетки
4 10   - строка и столбец конечной клетки
Поле (первая строка и первый столбец – нумерация, для удобства)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 1 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 |
| 3 | 1 | 1 | 1 | 1 | -1 | 1 | 1 | -1 | 1 | 1 | -1 | 1 | -1 | 1 | 1 | 1 |
| 4 | 1 | -1 | 1 | 1 | -1 | 1 | 1 | 1 | -1 | 1 | -1 | 1 | -1 | 1 | 1 | 1 |
| 5 | 1 | -1 | -1 | 1 | -1 | 1 | -1 | 1 | 1 | -1 | 1 | 1 | -1 | 1 | 1 | 1 |
| 6 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | 1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 |
| 7 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 | 1 |
| 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 |
| 10 | 1 | 1 | 1 | 1 | 1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 |
| 11 | 1 | -1 | -1 | -1 | 1 | -1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 |
| 12 | 1 | 1 | -1 | 1 | 1 | -1 | 1 | 0 | -1 | 1 | 1 | -1 | 1 | 1 | 1 | 1 |
| 13 | 1 | 1 | -1 | 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 | 1 | 1 | 1 |
| 14 | 1 | 1 | -1 | 1 | 1 | -1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | 1 | 1 |
| 15 | 1 | 1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 16 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |


1 2 3 4 5 6 7 8 9 A B C D E F G H I
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
3 1 0 0 1 1 1 0 0 0 0 0 1 1 1 1 0 0 1
4 1 0 0 0 0 1 0 0 1 0 0 1 0 1 0 0 0 1
5 1 0 1 0 0 1 0 0 0 1 0 1 0 1 0 0 0 1
6 1 0 1 1 0 1 0 1 0 0 1 0 0 1 0 0 0 1
7 1 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 1
8 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
9 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1
A 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1
B 1 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 0 1
C 1 0 1 1 1 0 1 0 0 0 0 0 0 1 0 0 0 1
D 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 0 0 1
E 1 0 0 1 0 0 1 1 1 1 0 0 0 1 0 0 0 1
F 1 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1
G 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1
H 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
I 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

```
type pp=record
       x,y:integer;//координаты предшественника
       v:integer;//"вес" клетки
       f:integer;//флаг: стенка: f=-1, обычное поле: f=0
       s:integer;//протяженность пути до этой клетки (сумма "весов" всех клеток маршрута)
      end;
var pole:array [,] of pp;//динамический двумерный массив записей
    xs,ys,xe,ye,m,n:integer;

procedure Rread(s:string); begin
  var f:=openread(s);   //открываем файл для чтения;
 1) readln(f,m,n); //читаем из него m,n - количество строк и столбцов поля
 2) readln(f,xs,ys);//читаем xs,ys - координаты стартовой клетки (начало координат: [1,1])
 3) readln(f,xe,ye); //читаем из файла xe,ye - координаты конечной клетки
 4) readln(f);readln(f);
 5) setlength(pole,m+2,n+2);//здесь строки поля 1..m, а колонки 1..n. Остальное - стенка
 6) for var i:=0 to n+1 do begin pole[0,i].f:=1; pole[m+1,i].f:=1; end;//окружаем поле стеной
 7) for var i:=0 to m+1 do begin pole[i,0].f:=1; pole[i,n+1].f:=1; end;//...
 8) for var i:=1 to m do begin
 9)  read(f,pole[i,1].v);//читаем вспомогательный столбец i-й строки
 10) for var j:=1 to n do //читаем i-ю строку, присваиваем s,v,f (x,y присваивать не надо)
 11)  with pole[i,j] do begin read(f,v); s:=1000;if v=-1 then f:=1 else f:=0; end;
 12)  readln(f);//читаем конец строки (для перехода на следующую)
 13) end; f.close; pole[xs,ys].s:=0;
 14) for var i:=0 to m+1 do begin for var j:=0 to n+1 do write(pole[i,j].f:2);writeln end;//печать
end;

procedure Pathfinder(i,j:integer); begin
 15) with pole[i+1,j] do if (f<>1)and(s>pole[i,j].s+v) then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(i+1,j);end;
 16) with pole[i-1,j] do if (f<>1)and(s>pole[i,j].s+v) then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(i-1,j);end;
 17) with pole[i,j+1] do if (f<>1)and(s>pole[i,j].s+v) then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(i,j+1);end;
 18) with pole[i,j-1] do if (f<>1)and(s>pole[i,j].s+v) then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(i,j-1);end;
end;

begin
 19) Rread('dina1.txt');Pathfinder(xs,ys); println(pole[xe,ye].s);
 20) var s:='->('+xe.ToString+','+ye.ToString+')';
 21) while ((xe,ye)<>(xs,ys)) do (s,xe,ye):=('->('+xe.ToString+','+ye.ToString+')'+s, pole[xe,ye].x,pole[xe,ye].y);
 22) writeln('(',xs,',',ys,')',s);
end.
```

```
type pp=record
       x,y:integer;//координаты предшественника
       v:integer;//"вес" клетки
       s:integer;//протяженность пути до этой клетки (сумма "весов" всех клеток маршрута)
      end;
var pole:array [,] of pp;//динамический двумерный массив записей
    xs,ys,xe,ye,m,n:integer;

procedure Rread(s:string); begin
  var f:=openread(s);   //открываем файл для чтения;
  readln(f,m,n); //читаем из него m,n - количество строк и столбцов поля
  readln(f,xs,ys);//читаем xs,ys - координаты стартовой клетки (начало координат: [1,1])
  readln(f,xe,ye); //читаем из файла xe,ye - координаты конечной клетки
  readln(f);readln(f);
  setlength(pole,m+2,n+2);//здесь строки поля 1..m, а колонки 1..n. Остальное - стенка
  for var i:=0 to n+1 do begin pole[0,i].v:=1001; pole[m+1,i].v:=1001; end;//окружаем поле стеной
  for var i:=0 to m+1 do begin pole[i,0].v:=1001; pole[i,n+1].v:=1001; end;//...
  for var i:=1 to m do begin
   read(f,pole[i,1].v);//читаем вспомогательный столбец i-й строки
   for var j:=1 to n do //читаем i-ю строку, присваиваем s,v,f (x,y присваивать не надо)
    with pole[i,j] do begin read(f,v); s:=1000;if v=-1 then v:=1001; end;
   readln(f);//читаем конец строки (для перехода на следующую)
  end; f.close; pole[xs,ys].s:=0;
end;

procedure Pathfinder(i,j:integer); begin
  with pole[i+1,j] do if s>pole[i,j].s+v then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(i+1,j);end;
  with pole[i-1,j] do if s>pole[i,j].s+v then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(i-1,j);end;
  with pole[i,j+1] do if s>pole[i,j].s+v then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(i,j+1);end;
  with pole[i,j-1] do if s>pole[i,j].s+v then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(i,j-1);end;
end;

begin
 Rread('dina1.txt');Pathfinder(xs,ys); println(pole[xe,ye].s);
 var s:='->('+xe.ToString+','+ye.ToString+')';
 while ((xe,ye)<>(xs,ys)) do (s,xe,ye):=('->('+xe.ToString+','+ye.ToString+')'+s,pole[xe,ye].x,pole[xe,ye].y);
 writeln('(',xs,',',ys,')',s);
end.
```

```
type pp=record
      x,y:integer;//координаты предшественника
      v:integer;//"вес" клетки
      s:integer;//протяженность пути до этой клетки (сумма "весов" всех клеток маршрута)
     end;
var pole:array [,] of pp;//динамический двумерный массив записей
    xs,ys,xe,ye,m,n:integer;

procedure Rread(s:string); begin
  var f:=openread(s);   //открываем файл для чтения;
  readln(f,m,n); //читаем из него m,n - количество строк и столбцов поля
  readln(f,xs,ys);//читаем xs,ys - координаты стартовой клетки (начало координат: [1,1])
  readln(f,xe,ye); //читаем из файла xe,ye - координаты конечной клетки
  readln(f);readln(f);
  setlength(pole,m+2,n+2);//здесь строки поля 1..m, а колонки 1..n. Остальное - стенка
  for var i:=0 to n+1 do begin pole[0,i].v:=1001; pole[m+1,i].v:=1001; end;//окружаем поле стеной
  for var i:=0 to m+1 do begin pole[i,0].v:=1001; pole[i,n+1].v:=1001; end;//...
  for var i:=1 to m do begin
   read(f,pole[i,1].v);//читаем вспомогательный столбец i-й строки
   for var j:=1 to n do //читаем i-ю строку, присваиваем s,v,f (x,y присваивать не надо)
    with pole[i,j] do begin read(f,v); s:=1000;if v=-1 then v:=1001; end;
   readln(f);//читаем конец строки (для перехода на следующую)
  end; f.close; pole[xs,ys].s:=0;
end;

procedure Pathfinder(i,j:integer);
  procedure St1(a,b:integer):=with pole[a,b] do if s>pole[i,j].s+v then begin (s,x,y):=(pole[i,j].s+v,i,j);Pathfinder(a,b);end;
begin
  St1(i-1,j); St1(i+1,j); St1(i,j+1); St1(i,j-1);
end;

begin
 Rread('dina1.txt');Pathfinder(xs,ys); println(pole[xe,ye].s);
 var s:='->('+xe.ToString+','+ye.ToString+')';
 while ((xe,ye)<>(xs,ys)) do (s,xe,ye):=('->('+xe.ToString+','+ye.ToString+')'+s,pole[xe,ye].x,pole[xe,ye].y);
 writeln('(',xs,',',ys,')',s);
end.
```

# Проход «в ширину» (от Сорокиной М.А., C++)…

```
1  #include <iostream>
2  #include <string>
3  #include <vector>
4  #include <algorithm>
5  #include <queue>
6  #include <math.h>
7  #include "cmath"
8  #include "deque"
9  using namespace std;
10 int main() {
11   vector<vector<int> > a(17, vector <int> (17, 0)), used(16, vector <int> (17, 0)), d(16, vector <int> (16, -1));
12   vector<vector<pair<int, int> > > p(16, vector <pair <int, int> > (16));
13   for (int i = 0; i<16; ++i)  for (int j = 0; j<16; ++j)  cin >> a[i][j];
14   pair<int, int> s = {11,7};
15   queue<pair<int, int> > q;
16   q.push (s);  used[11][7] = 1;  p[11][7] = {-1,-1};
17   while (!q.empty()) {
18     pair<int, int> v = q.front();  q.pop();
19     if (v.first == 3 && v.second == 9) break;
20     if (!used[v.first][v.second+1] && a[v.first][v.second+1]>0) {
21        used[v.first][v.second+1] = 1;  q.push ({v.first, v.second+1});
22        d[v.first][v.second+1] = d[v.first][v.second] + 1; p[v.first][v.second+1] = {v.first, v.second};   }
23     if (!used[v.first-1][v.second] && a[v.first-1][v.second]>0) {
24        used[v.first-1][v.second] = 1;  q.push ({v.first-1, v.second});
25        d[v.first-1][v.second] = d[v.first][v.second] + 1;  p[v.first-1][v.second] = {v.first, v.second};   }
26     if (!used[v.first][v.second-1] && a[v.first][v.second-1]>0) {
27        used[v.first][v.second-1] = 1;  q.push ({v.first, v.second-1});
28        d[v.first][v.second-1] = d[v.first][v.second] + 1; p[v.first][v.second-1] = {v.first, v.second};   }
29     if (!used[v.first+1][v.second] && a[v.first+1][v.second]>0) {
30        used[v.first+1][v.second] = 1;  q.push ({v.first+1, v.second});
31        d[v.first+1][v.second] = d[v.first][v.second] + 1;  p[v.first+1][v.second] = {v.first, v.second}; }
32    } cout << d[3][9]; return 0; }
```

# Проход «в ширину»…

**Задания по программе:**

1)Скопировать текст программы

2)Вычислить и вывести на печать
количество проверок (25 строка),
а также количество добавлений
элементов в очередь (26 строка)

3)Выделить поля **xo, yo** в отдельную структуру данных

4) Убрать их из структуры **pp**

5) Внести необходимые изменения в программу, заменив, где это необходимо, работу с элементами массива **pole** работой с новой структурой

*Итоговые программы прислать*
*в мой E-mail – адрес*

```
1  type pp=record
2        x,y,xo,yo:integer;//координаты предшественника
3        v,s:integer;//"вес" клетки b протяженность пути до нее (сумма "весов" всех клеток маршрута)
4       end;
5  var pole:array [,] of pp;//динамический двумерный массив записей
6      xs,ys,xe,ye,m,n:integer; w:=new Queue<pp>; //создаем очередь для поиска в ширину
7  procedure rread(s:string); begin
8    var f:=openread(s);   //открываем файл для чтения;
9    readln(f,m,n); //читаем из него m,n - количество строк и столбцов поля
10   readln(f,xs,ys);  //читаем из файла xs,ys - координаты стартовой клетки (начало координат: [1,1])
11   readln(f,xe,ye); //читаем из файла xe,ye - координаты конечной клетки
12   readln(f);readln(f);
13   setlength(pole,m+2,n+2);//здесь строки поля 1..m, а колонки 1..n. Остальное - стенка
14   for var i:=0 to n+1 do begin pole[0,i].v:=1001; pole[m+1,i].v:=1001; end;//окружаем поле стеной
15   for var i:=0 to m+1 do begin pole[i,0].v:=1001; pole[i,n+1].v:=1001; end;//...
16   for var i:=1 to m do begin read(f,pole[i,1].v);//читаем вспомогательный столбец i-й строки
17    for var j:=1 to n do begin //читаем i-ю строку (read) и присваиваем компоненты s,v,f (x,y присваивать не надо)
18      read(f,pole[i,j].v); pole[i,j].s:=1000;if pole[i,j].v=-1 then pole[i,j].v:=1001;
19      pole[i,j].xo:=i;pole[i,j].yo:=j; //храним координаты поля!!!
20    end;readln(f);//читаем конец строки (для перехода на следующую)
21   end; f.close; pole[xs,ys].s:=0;
22 end;
23 procedure pathfinder;
24  procedure moving(vv:pp;a,b:integer):=with vv do
25   if s+pole[xo+a,yo+b].v<pole[xo+a,yo+b].s then begin
26    pole[xo+a,yo+b].s:=s+pole[xo+a,yo+b].v; pole[xo+a,yo+b].x:=xo;pole[xo+a,yo+b].y:=yo; w.Enqueue(pole[xo+a,yo+b])
27   end;
28 begin
29  w.Enqueue(pole[xs,ys]); var vv:pp;
30  while w.Count>0 do begin
31   vv:=w.Dequeue; if (vv.xo=xe) and (vv.yo=ye) then break;
32   moving(vv,1,0);moving(vv,-1,0);moving(vv,0,1);moving(vv,0,-1);
33  end;
34 end;
35 begin
36  rread('dina1.txt'); pathfinder; println(pole[xe,ye].s);
37  var s:='->('+xe.ToString+','+ye.ToString+')';
38  while ((xe,ye)<>(xs,ys)) do (s,xe,ye):=('->('+xe.ToString+','+ye.ToString+')'+s,pole[xe,ye].x,pole[xe,ye].y);
39  writeln('(',xs,',',ys,')',s);
40 end.
```