**Пример 1.**

**Из листа клетчатой бумаги размером NxM удалили некоторые клетки размером 1 х 1. Определите, на сколько связных кусков распадается оставшаяся часть листа. К - количество удалённых клеток, (х1; у1), …, (хk; yk) - координаты удалённых клеток.**

**Для решения данной задачи рассмотрим двумерный массив NxM , где удалённые клетки имеют значение -1, а неудалённые - 0.**

**В нашем алгоритме мы будем находить первую клетку со значением 0, помечаем её определённым образом, а потом находим и помечаем клетки, имеющие общую сторону с найденной клеткой и к тому же со значением 0. Потом находим и помечаем «нулевых» соседей последних найденных клеток и т.д. Процесс заканчивается, когда никаких новых клеток не удаётся найти. Значит, мы получили единый кусок листа.**

**Далее выбираем новую начальную клетку и опять повторяем процесс нахождения её «соседей», равных 0.**

**Алгоритм закончится, когда на «листе» не останется клеток со значением 0.**

**Для решения задачи будем использовать очередь. В очереди будем хранить вновь помеченные клетки.**

**Для примера рассмотрим лист размером 4 х 5, где удалёнными (со значением -1) считаются клетки с координатами (1;5), (4;5), (3;4), (2;3), (2;2), (1;1), (3;1). Итак,**

|  |
| --- |
|  |
| i \ j | 1 | 2 | 3 | 4 |  |
| 1 | -1 | 0 | -1 | 0 |  |
| 2 | 0 | -1 | 0 | 0 |  |
| 3 | 0 | -1 | 0 | 0 |  |
| 4 | 0 | 0 | -1 | 0 |  |
| 5 | -1 | 0 | 0 | -1 |  |
|  |  |  |  |  |  |

**Теперь найдём первую клетку со значением 0 и занесём её координаты в очередь. При этом в матрицу на соответствующее место ставится номер куска, который рассматривается (в данном случае 1). Первой найдена клетка с координатами (i=1;j=2).**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| Индексы | | | | 1 | 2 | | 3 | | 4 | | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |  |
| i \ j | 1 | 2 | 3 | | | 4 | |  | |
| 1 | -1 | 1 | -1 | | | 0 | |  | |
| 2 | 0 | -1 | 0 | | | 0 | |  | |
| 3 | 0 | -1 | 0 | | | 0 | |  | |
| 4 | 0 | 0 | -1 | | | 0 | |  | |
| 5 | -1 | 0 | 0 | | | -1 | |  | |
|  |  |  |  | | |  | |  | |

**Рядом с этой клеткой нет других, со значением 0. Удаляем эту клетку из очереди. Потом ищем следующую начальную клетку. Это будет клетка с координатами (1;4). Её в таблице помечаем номером 2 (второй кусок). В очередь вносим её координаты и координаты её нулевых «соседей»-клеток, а потом их «соседей», и т.д. Получим следующее:**

|  |
| --- |
|  |
| Индексы | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |  |
| i | 1 | 1 | 2 | 2 | 3 | 3 | 3 | 4 |  |  |  |  |  |  |  |
| j | 2 | 4 | 4 | 3 | 4 | 3 | 4 | 4 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**На каждом шаге после проверки элемента на наличие нужных «соседей» и его пометке в массиве, удаляем его из очереди. Продолжаем процесс до тех пор, пока очередь не станет пуста.**

**Массив-лист примет следующий вид:**

|  |
| --- |
|  |
| i \ j | 1 | 2 | 3 | 4 |  |
| 1 | -1 | 1 | -1 | 2 |  |
| 2 | 0 | -1 | 2 | 2 |  |
| 3 | 0 | -1 | 2 | 2 |  |
| 4 | 0 | 0 | -1 | 2 |  |
| 5 | -1 | 0 | 0 | -1 |  |
|  |  |  |  |  |  |

**Следующий начальный нулевой элемент с координатами (2;1) будет уже помечаться номером 3 (третий кусок). Вносим его координаты в очередь, и действуем по знакомому алгоритму.**

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

**Массив примет вид:**

|  |
| --- |
|  |
| i \ j | 1 | 2 | 3 | 4 |  |
| 1 | -1 | 1 | -1 | 2 |  |
| 2 | 3 | -1 | 2 | 2 |  |
| 3 | 3 | -1 | 2 | 2 |  |
| 4 | 3 | 3 | -1 | 2 |  |
| 5 | -1 | 3 | 3 | -1 |  |
|  |  |  |  |  |  |

**Больше нулевых элементов нет. Поэтому алгоритм будет завершен. Количество кусков равно 3.**

//Из листа клетчатой бумаги размером N x M удалили некоторые клетки.

//Определить, на сколько связных кусков распадается оставшаяся часть листа. // К - количество удалённых клеток.

#include <iostream>#include <fstream>#include <iomanip>#include <math.h>

using namespace std;

struct queue //динамическая очередь

{

int x; //номер строки матрицы

int y; //номер столбца матрицы

queue\* next; //нссылка на след элемент

};

queue \*head,\*tail;

void InitQueue() //инициализация очереди

{

head = 0; tail = 0;

}

bool isEmpty() //проверка на пустоту

{

return head == 0 ? true : false;

}

void peek( int \*a,int \*b) //взятие из очереди первого элемента без удаления

{

if (!isEmpty())

{ \*a = head->x; \*b = head->y; }

}

void pop() // удаление из очереди первого элемента

{

if (!isEmpty())

{

queue\* tmp = head; head = head->next;

delete tmp;

}}

void push(int a,int b) // добавление в очередь элемента

{

queue\* tmp = new queue; tmp->x=a;

tmp->y=b; tmp->next = 0;

if(head == 0) {

head = tmp;

tail = head;

return;

}

tail->next = tmp;

tail = tmp; tail->next=0;

}

void main()

{ setlocale(LC\_ALL,"rus");

ifstream f1;

f1.open("input1.txt");

if(!f1.is\_open())

{cerr<<"Open file failed."<<endl;

return; }

int i,j,n,m,k,x,y,x1,y1;

int \*\*Lst; //матрица для хранения листа бумаги

f1>>n>>m;

Lst=new int \*[n];

for(i=0;i<n;i++)

Lst[i]=new int[m];

cout<<"Лист бумаги дырки -1 "<<endl;

for(i=0;i<n;i++) {

for(j=0;j<m;j++) {f1>>Lst[i][j];cout<<setw(3)<<Lst[i][j]<<" ";

cout<<endl<<endl; }

InitQueue(); k=0;

for(i=0;i<n;i++) {

for(j=0;j<m;j++)

{ if (Lst[i][j]==0)

{

k++; push(i,j); Lst[i][j]=k;

while (!isEmpty())

{ while (!isEmpty())

{ peek(&x,&y);

if ( x>0 && Lst[x-1][y]==0 ) { x1=x-1; push(x1,y); Lst[x1][y]=k; }

if ( x<n-1 && Lst[x+1][y]==0 ) { x1=x+1; push(x1,y); Lst[x1][y]=k; }

if ( y>0 && Lst[x][y-1]==0 )

{ y1=y-1; push(x,y1); Lst[x][y1]=k; }

if ( y<m-1 && Lst[x][y+1]==0 )

{ y1=y+1; push(x,y1); Lst[x][y1]=k; }

pop();

}// for

}//while is\_empty

}//if

} //for j

} //for i

cout<<"Количество кусков k="<<k<<endl;}

Задача “Дорога”

Имеется план местности, разбитой на квадраты, заданный матрицей размером M x N. Каждый квадрат имеет одно из обозначений : 0, если в квадрате твёрдая почва, и -1, если в квадрате болото. Необходимо определить, имеется ли в маршрут робота из позиции (ХС, УС) в позицию (ХФ, УФ), если робот может двигаться только по ровной местности и только в горизонтальном и вертикальном направлениях. При этом вдоль границ квадратов двигаться нельзя.

Решение задачи похоже на решение задачи “Лист бумаги” с несколькими отличиями:

1) нам не нужно больше искать все куски по которым ходит робот, перемещаться нужно будет только в пределах одного куска;

2) необходимо сделать проверку исходной точки - старта робота, и конечной - финиша, не являются ли они координатами болота. Если это так - нет смысла искать маршрут.

3) необходимо каждой паре координат, попадающей в очередь поставить в соответствие число, означающее номер шага робота,