МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Отчет № 12

по дисциплине «Информатика»

на тему: « Работа с массивами. Сложные сортировки. Сортировка Шелла. Сортировка Хоора. Сравнительные показатели производительности различных методов сортировки массивов»

Выполнил:

студент группы 3530902/90001 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Непушкин Сергей Александрович

Проверил:

Доцент ВШКФСиУ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Теплова Наталья Витальевна

Санкт-Петербург

2019 г.

Оглавление

1.Задание………………………………………………………………………………………….3

2.Блок-схема алгоритма…………………………………………………………………………4

3.Текст кода………………………………………………………………………………………6

4.Пример работы программы……………………………………………………………………9

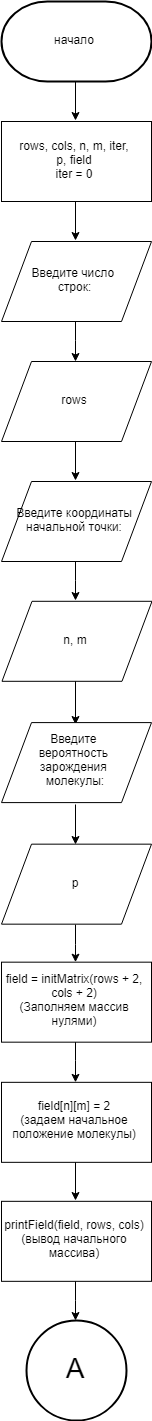
**1. Задание.**

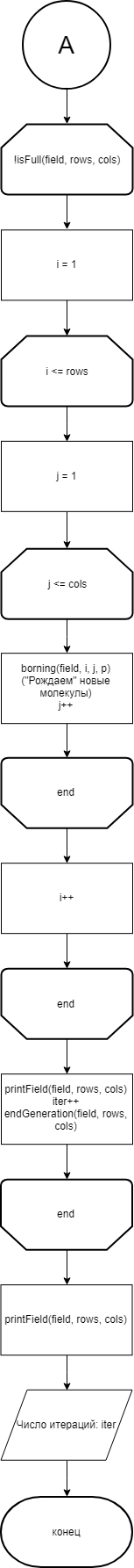
Написать программу, определяющую количество шагов, потребовавшихся для размножения молекул в заданном поле.

Входные файлы: число клеток по горизонтали, число клеток по вертикали, вероятность зарождения молекулы, начальные координаты молекулы.

При зарождении молекулы вместо 0 выводится 1, когда молекула проживает 1 шаг и способна размножаться 1 заменяется на 2.

**2.Блок-схема алгоритма**

****

****

**3.Текст кода.**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <conio.h>

#include <math.h>

#include <locale.h>

#include <stdlib.h>

#include <time.h>

int\*\* initMatrix(int n, int m)

{

int\*\* result = (int\*\*)calloc(n, sizeof(int\*));

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

{

result[i] = (int\*)calloc(m, sizeof(int));

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

{

result[i][j] = 0;

}

}

return result;

}

void deleteMatrix(int\*\* matrix, int rows)

{

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

{

free(matrix[i]);

}

free(matrix);

}

void printField(int\*\* matrix, int rows, int cols)

{

system("cls");

for (int i = 0; i <= rows + 1; i++)

{

for (int j = 0; j <= cols + 1; j++)

{

if (i == 0)

{

if (j == 0) printf("%c", 201);

else if (j == cols + 1) printf("%c", 187);

else printf("%c%c%c", 205, 205, 205);

}

else if (i == rows + 1)

{

if (j == 0) printf("%c", 200);

else if (j == cols + 1) printf("%c", 188);

else printf("%c%c%c", 205, 205, 205);

}

else if (j == 0 || j == cols + 1)

{

printf("%c", 186);

}

else {

printf(" %d ", matrix[i][j]);

}

}

printf("\n");

}

}

int isFull(int\*\* matrix, int rows, int cols)

{

for (int i = 1; i < rows + 1; i++)

{

for (int j = 1; j < cols + 1; j++)

{

if (matrix[i][j] == 0)

{

return 0;

}

}

}

return 1;

}

int countNeighbours(int\*\* matrix, int n, int m)

{

int count = 0;

for (int i = n - 1; i <= n + 1; i++)

{

for (int j = m - 1; j <= m + 1; j++)

{

if (matrix[i][j] > 1)

{

count += 1;

}

}

}

return count;

}

void endGeneration(int\*\* matrix, int rows, int cols)

{

for (int i = 1; i <= rows; i++)

{

for (int j = 1; j <= cols; j++)

{

if (matrix[i][j] == 1)

{

matrix[i][j] = 2;

}

}

}

}

int isBorn(float p, int neighbours)

{

if (rand() % 100 < (1 - pow(1 - p, neighbours)) \* 100)

{

return 1;

}

return 0;

}

void borning(int\*\* matrix, int n, int m, float p)

{

if (matrix[n][m] >= 1) {

return;

}

int neighbours = countNeighbours(matrix, n, m);

if (neighbours == 0) {

return;

}

if (isBorn(p, neighbours))

{

matrix[n][m] = 1;

}

}

int main()

{

srand(time(0));

setlocale(LC\_ALL, "Russian");

int rows, cols, n, m, iter;

iter = 0;

float p;

char c;

int\*\* field;

printf("Введите число строк: ");

scanf("%d", &rows);

printf("Введите число столбцов: ");

scanf("%d", &cols);

printf("Введите координаты начальной точки: ");

scanf("%d", &n);

scanf("%d", &m);

printf("Введите вероятность зарождения молекулы: ");

scanf("%f", &p);

setlocale(LC\_ALL, "C");

field = initMatrix(rows + 2, cols + 2);

field[n][m] = 2;

printField(field, rows, cols);

scanf("%c", &c);

while (!isFull(field, rows, cols))

{

for (int i = 1; i <= rows; i++)

{

for (int j = 1; j <= cols; j++)

{

borning(field, i, j, p);

}

}

printField(field, rows, cols);

iter++;

endGeneration(field, rows, cols);

scanf("%c", &c);

}

printField(field, rows, cols);

setlocale(LC\_ALL, "Russian");

printf("Число итераций: %d", iter);

deleteMatrix(field, rows + 2);

return 0;

}

**4.Пример работы программы**

