Министерство науки и высшего образования Российской федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт цифровых технологий, электроники и физики

Кафедра вычислительной техники и электроники

Курс «Математическое моделирование»
Отчет по лабораторной работе №3
«Игра “Жизнь”»

Выполнил: студент 585гр.

Роженцев А.К. _____

Проверил: ст.пр.

Уланов П.Н. _____

Барнаул 2020

1 Цель работы

Реализовать программу, моделирующую игру «Жизнь» с такими правилами:

- Клетки на квадратной доске могут находиться в двух состояниях: «живое» и «мертвое».
- «Живая» клетка выживает на очередном временном шаге, если имеет только 2 или 3 живых соседа.
- Если соседей меньше двух, клетка умирает из-за обособленности, а если больше трех, то из-за скученности (перенаселения).
- «Мертвая» оживает на очередном шаге только в том случае, если имеет 3 живых соседа.
- У каждой клетки 8 соседей: клетки, имеющие с ней общие стороны или вершины.
- Изменение состояния всех клеток происходит одновременно.

2 Задание

1. Реализовать программу, моделирующую игру “Жизнь”.
2. Задать центральную часть поля случайной конфигурацией. Размеры случайной области от 3x3 до 10x10.
3. Многократными запусками программы найти начальные конфигурации, приводящие к:
    - вымиранию
    - стабильной конфигурации
    - периодической конфигурации
    - конфигурации, развивающийся не менее 100 поколений

3 Программа

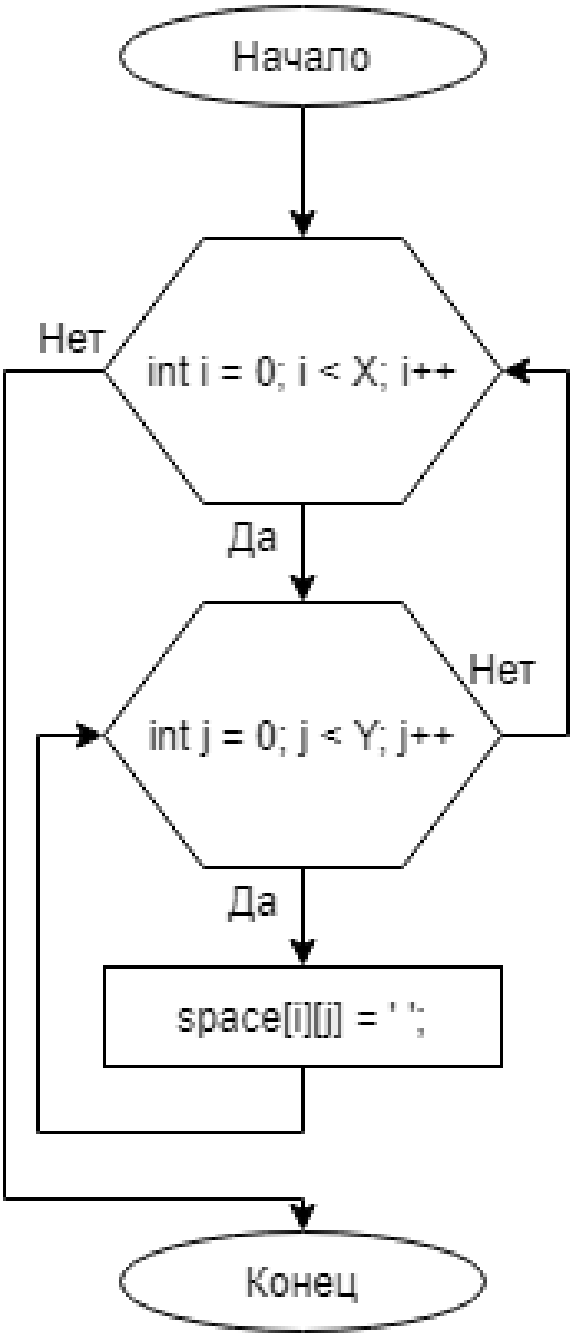

Рис. 1: Подпрограмма initialization

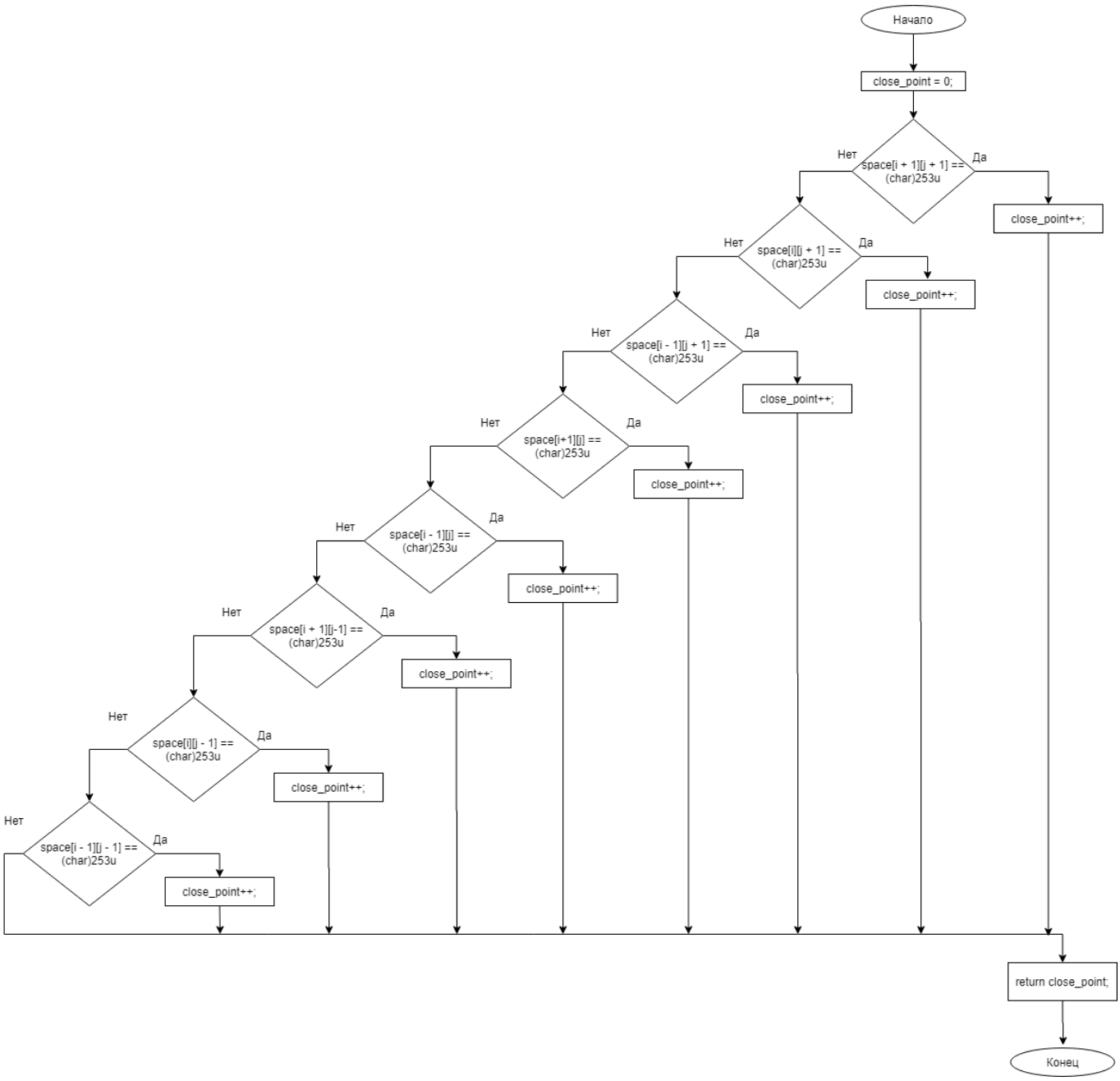

Рис. 2: Подпрограмма FIND_POINTS

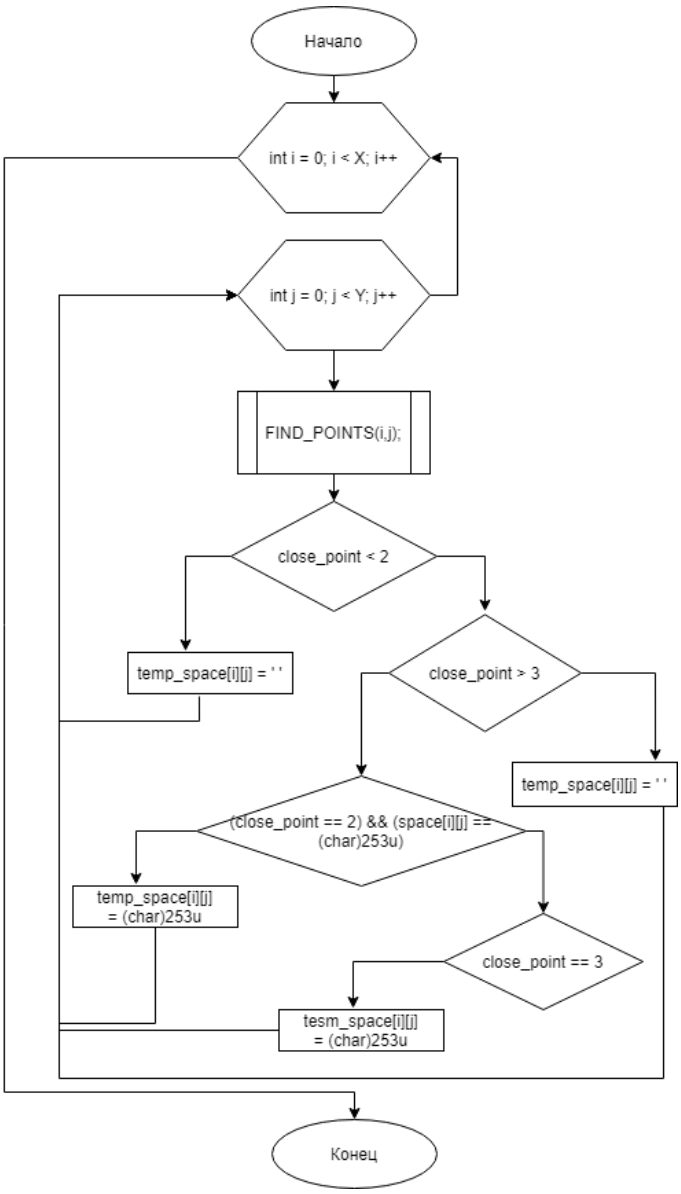

Рис. 3: Подпрограмма RULES

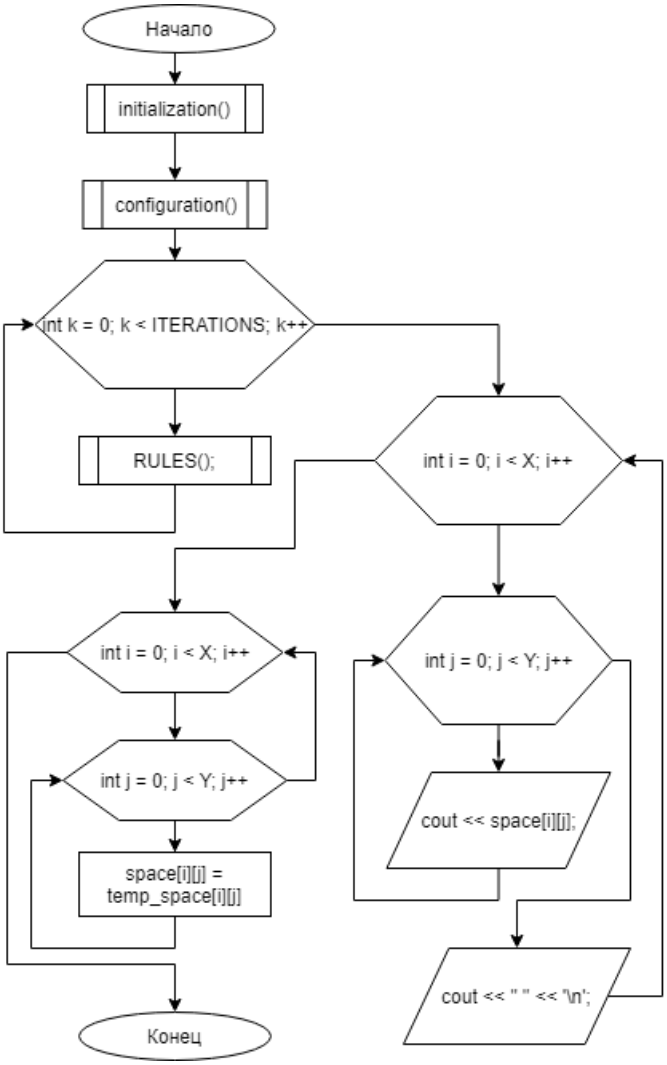

Рис. 4: Основная программа main

4 Конфигурации

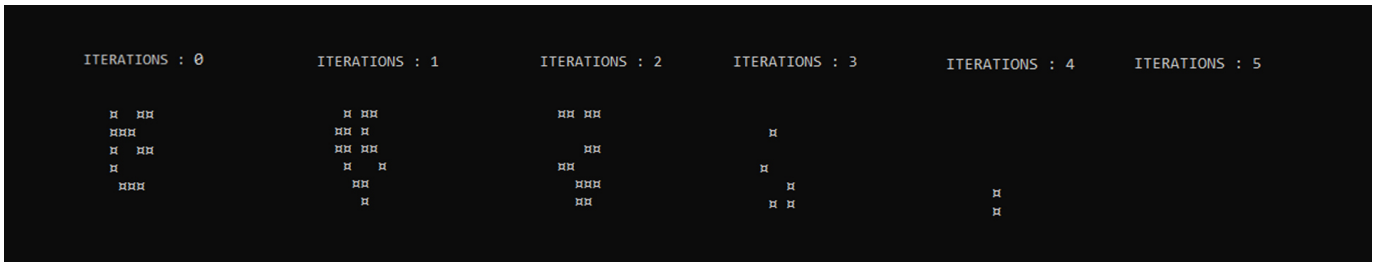

Рис. 5: Вымирание.

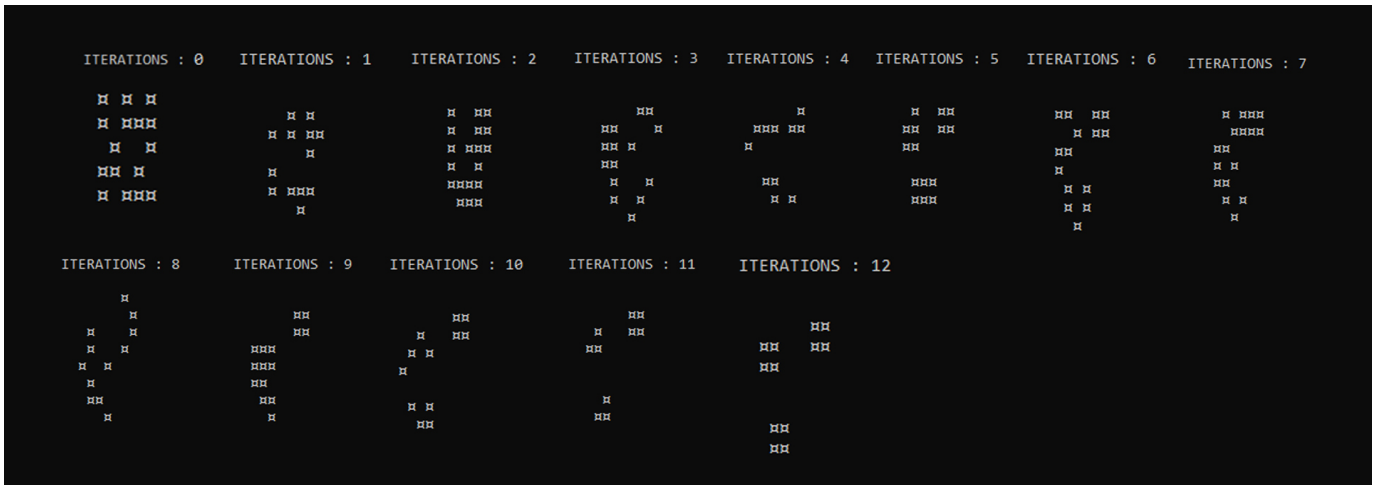

Рис. 6: Стабильная конфигурация.

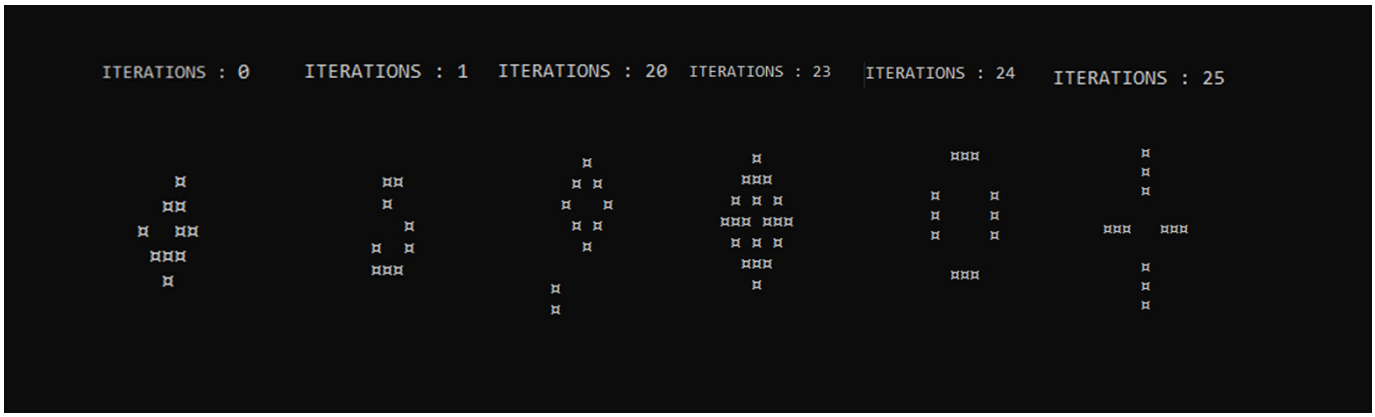

Рис. 7: Периодическая конфигурация.

5 Код программы

```
#include <iostream>
#include <conio.h>
#include <stdio.h>
#include<Windows.h>
#include <stdlib.h>
#include <time.h>


using namespace std;
#define ITERATIONS 1000
#define X 30
#define Y 30

char space[X][Y];
char temp_space[X][Y];
int close_point;

void initialization(){
  for (int i = 0; i < X; i++) {
    for (int j = 0; j < Y; j++) {
      space[i][j] = ' ';
    }
  }
}

void configurations(){
  space[3][6] = (char)253u;
  space[4][5] = (char)253u;
  space[4][6] = (char)253u;
  space[5][3] = (char)253u;
  space[5][6] = (char)253u;
  space[5][7] = (char)253u;
  space[6][4] = (char)253u;
  space[6][5] = (char)253u;
  space[6][7] = (char)253u;
  space[7][5] = (char)253u;
}


int FIND_POINTS(int i, int j){
  close_point = 0;
  if (space[i + 1][j + 1] == (char)253u)
  {
    close_point++;
  }
  if (space[i][j + 1] == (char)253u)
  {
    close_point++;
```

```
49    }
50    if (space[i - 1][j + 1] == (char)253u)
51    {
52      close_point++;
53    }
54    if (space[i+1][j] == (char)253u)
55    {
56      close_point++;
57    }
58    if (space[i - 1][j] == (char)253u)
59    {
60      close_point++;
61    }
62    if (space[i + 1][j-1] == (char)253u)
63    {
64      close_point++;
65    }
66    if (space[i][j - 1] == (char)253u)
67    {
68      close_point++;
69    }
70    if (space[i - 1][j - 1] == (char)253u)
71    {
72      close_point++;
73    }
74    return close_point;
75  }
76
77  void RULES(){
78    for (int i = 0; i < X; i++) {
79      for (int j = 0; j < Y; j++) {
80        FIND_POINTS(i,j);
81        if (close_point < 2)
82        {
83          temp_space[i][j] = ' ';
84        }
85        else if (close_point > 3)
86        {
87          temp_space[i][j] = ' ';
88        }
89        else if ((close_point == 2) && (space[i][j] == (char)253u))
90        {
91          temp_space[i][j] = (char)253u;
92        }
93        else if (close_point == 3)
94        {
95          temp_space[i][j] = (char)253u;
96        }
97      }
98    }
```

```
99  }
100
101 int main(){
102   initialization();
103   configurations();
104   for (int k = 0; k < ITERATIONS; k++){
105     RULES();
106     for (int i = 0; i < X; i++) {
107       for (int j = 0; j < Y; j++) {
108         cout << space[i][j];
109       }
110       cout << " " << '\n';
111     }
112     for (int i = 0; i < X; i++) {
113       for (int j = 0; j < Y; j++) {
114         space[i][j] = temp_space[i][j];
115       }
116     }
117     _getch();
118     system("cls");
119     cout<<"ITERATIONS : "<<k+1;
120   }
121   _getch();
122 }
```