**Программа «Жизнь»**

Подготовлено Торяником Александром, институт искусственного интеллекта, группа КВБО-01-21

Представленная мною программа выполняет условия программы «Жизнь»

Сначала создаётся файл work.dat (жилище микробов) и файл work.out (описывающий текущее поколение микробов, их развитие). Файл work.dat состоит из различных символов,среди которых «обитают» микробы. 2. Необходимо создать массивы «настоящее» и «будущее». В массиве «настоящее» записывается текущее поколение микробов, а в файл «будущее»- следующее. Массивы создаются размером 21х21. 3. Программа выдаёт пользователю на экран запрос: «Введите количество поколений». Именно столько поколений программа будет описывать. 4. Теперь программа создаёт в файле work.out поколение под номером 1. Для этого программа открывает и проверяет на наличие микробов файл (каждый символ, среди которых может затеряться микроб). Если под символом скрывается микроб (например, символ «Х»), то в массив «настоящее» записывается единица (1, т.е. микроб только что родился), а если не микроб – то ноль (0, т. е. там никто не живёт). Теперь в массиве «настоящее» находиться поле из 1 и 0 (он состоит из новорожденных младенцев и пустых мест). Все последующие поколения тоже записываются в файл work.out. Файл work.dat закрывается, и работа теперь ведётся только с файлом work.out. 5. Теперь описывается следующее поколение. Оно создаётся после проверки массива «настоящее». Проверяются микробы и их соседи (результат записывается в массив «будущее»): - если жизненный уровень микроба от 1 до 11 то: --если соседей 2 или 3, то микроб продолжает жить и подрастает (т.е. его жизненный уровень возрастает на 1), --иначе микроб погибает (=0), т.к. он задыхается или умирает от скуки; - если жизненный уровень микроба 0, то микроб рождается заново (т.е. его жизненный уровень равен 1); - если жизненный уровень микроба равен 12, то микроб погибает (от старости). 6. После проверки подсчитывается количество жизней данного поколения (т.е. сколько единиц). Затем программа проверяет, есть ли кто «живой на поле» или все погибли (т.е. нули). 7. Теперь программа производит замену поколений: массив «будущее» становится «настоящим». 100 8. Программа продолжает работу с пункта 5 до тех пор, пока количество поколений (которые описывает программа) не станет равным введенным пользователем в пункте 3 или пока не погибнут все микробы (т.е. везде одни нули).

**В моей вариации программы только микробы возрождаются вне зависимости от наличия соседей, как и было указано в pdf файле.**

План работы программы:

1.srand(unsigned(time(NULL)))

устанавливает начальное состояние генератора случайных чисел по текущему времени (иначе он будет выдавать одни и те же значения при повторных запусках программы);

2.файл work.dat заполняется рандомными буквами

ofstream outf("work.dat");

for (int i = 0; i < 21 \* 21; i++)

{

outf << char('a' + rand() % ('e' - 'a'));

}

outf.close();

3.выбирается рандомная буква, которая будет являться микробом

char z= char('a' + rand() % ('e' - 'a'));

4.далее создаются два двумерных массива 21x21

int nast[21][21] = { 0 };

int budu[21][21] = { 0 };

5. cin >> pokolenie проверяет правильность ввода пользователем количества поколений, которые необходимо воспроизвести

6.далее из файла заполнение массива nast 0 и 1 в зависимости от того, является ли буква микробом

for (int x = 0; x <= 441; x++)

{

int stroka = x % 21;

int stolbec = x / 21;

if (str[x] == z)

{

nast[stolbec][stroka] = 1;

}

else if (str[x] != z)

{

nast[stolbec][stroka] = 0;

}

}

7.далее данные из массива переносятся в файл

for (int o = 0; o <= 20; o++)

{

for (int p = 0; p <= 21; p++)

{

if (p != 21)

{

fout << nast[o][p] << " ";

}

else

fout << endl;

}

}

8.далее следует основной цикл while, внутри которого происходят все операции с микробами (наличие соседей, возраст,проверка есть ли кто-то живой в поколении,задается массив, который запоминает умерших микробов)

while (k <= pokolenie)

{

for (int c = 0; c <= 20; c++)

{

for (int d = 0; d <= 21; d++)

{

int sosedi = 0;

if (nast[c][d] == 12)

{

budu[c][d] = 0;

mertvie[c][d] = 1;

}

if (nast[c][d] < 12 and nast[c][d] > 0 and nast[c][d] != 12)

{

//проверка, не выходит ли "сосед" за пределы массива, чтобы можно было к нему обратиться

if (d - 1 > 0) {

if (nast[c][d - 1]) {

sosedi++;

}

}

if (c + 1 < 20 && d - 1 > 0) {

if (nast[c + 1][d - 1]) {

sosedi++;

}

}

if (c + 1 < 20) {

if (nast[c + 1][d]) {

sosedi++;

}

}

if (c + 1 < 20 && d + 1 < 21) {

if (nast[c + 1][d + 1]) {

sosedi++;

}

}

if (d + 1 < 21) {

if (nast[c][d + 1]) {

sosedi++;

}

}

if (c - 1 > 0 && d + 1 < 21) {

if (nast[c - 1][d + 1]) {

sosedi++;

}

}

if (c - 1 > 0) {

if (nast[c - 1][d]) {

sosedi++;

}

}

if (c - 1 > 0 && d - 1 > 0) {

if (nast[c - 1][d - 1]) {

sosedi++;

}

}

// проверка на необходимое количество соседей

if (sosedi == 2 or sosedi == 3)

{

budu[c][d] =nast[c][d] +1;

sosedi = 0;

}

Else //если соседей мало или много, то микроб погибает

{

budu[c][d] = 0;

mertvie[c][d] = 1;

sosedi = 0;

}

}

}

} //воскрешение микроба

for (int y = 0; y < 20; y++)

{

for (int e = 0; e <= 21; e++)

{

if (mertvie[y][e] == 1 and nast[y][e] == 0)

{

budu[y][e] = 1;

mertvie[y][e] = 0;

}

}

} //проверка есть ли живые микробы

int live = 0;

for (int i = 0; i < 20; i++)

{

for (int j = 0; j <= 21; j++)

{

if (budu[i][j] > 0)

{

live++;

}

}

} //если есть жизнь, то данные массива записываются в файл

if (live > 0)

{

for (int r = 0; r <= 20; r++)

{

for (int s = 0; s <= 21; s++)

{

if (s != 21)

{

fout << budu[r][s] << " ";

}

else

fout << endl;

}

}

}

Else //если жизни нет, то программа останавливается

{

cout << "Популяция погибла" << endl;

break;

} // смена массивов местами и обнуление массива budu

for (in q = 0; q <= 20; q++)

{

for (int t = 0; t <= 21; t++)

{

nast[q][t] = budu[q][t];

}

}

budu[21][21] = { 0 };

k++;

9.после выполнения условий, из файла work.out выводятся данные в консоль

ifstream fin("work.out");

cout << "Поколение 1:" << endl;

while (fin)

{

string strinput;

fin >> strinput;

cout << strinput << " ";

if (w <= pokolenie and q < 441 \* pokolenie - 2)

{

q++;

if (q % 21 == 0)

{

cout << endl;

}

if (q % 441 == 0)

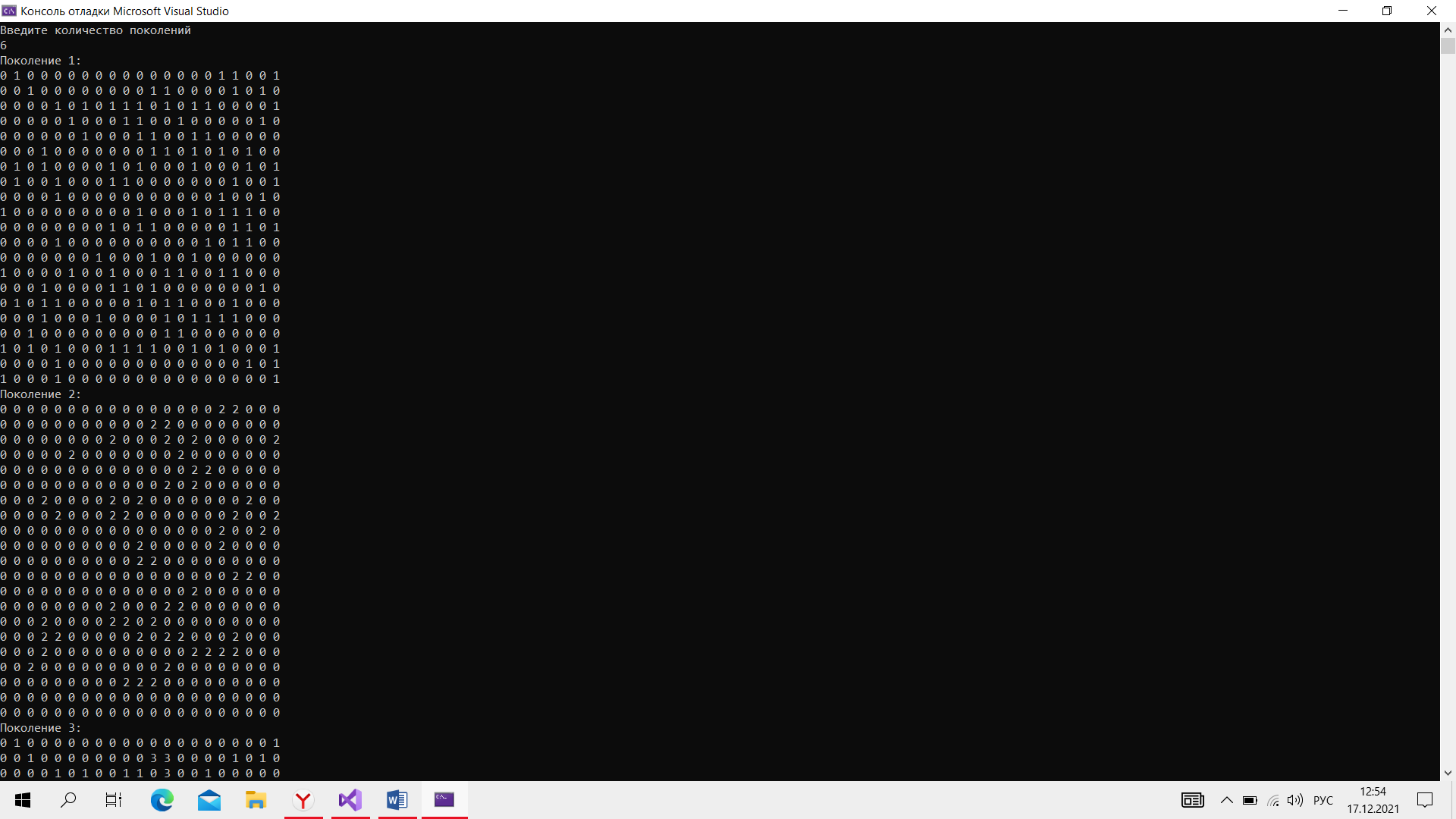
{

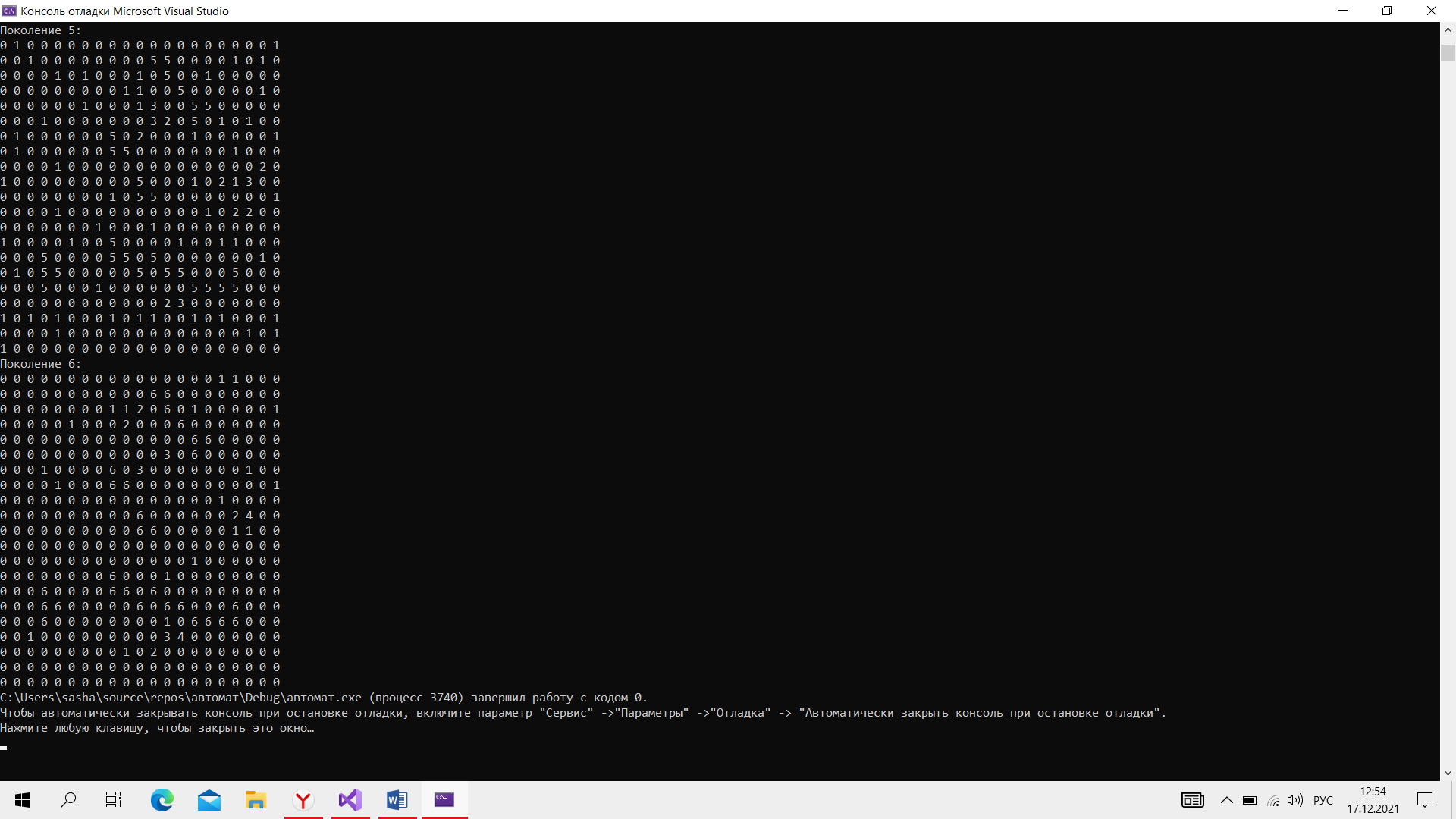
w++;

cout << "Поколение " << w << ":" << endl;

}

**Пример работы программы:**





Код программы на С++:

#include <fstream>

#include <ctime>

#include <string>

#include <iostream>

using namespace std;

int main()

{

srand(unsigned(time(NULL)));

setlocale(LC\_ALL, "RU");

ofstream outf("work.dat");

for (int i = 0; i < 21 \* 21; i++)

{

outf << char('a' + rand() % ('e' - 'a'));

}

outf.close();

char z= char('a' + rand() % ('e' - 'a'));

int nast[21][21] = { 0 };

int budu[21][21] = { 0 };

int pokolenie;

cout << "Введите количество поколений" << endl;

if (cin >> pokolenie)

{

ifstream inf("work.dat");

string str;

getline(inf, str);

inf.close();

ofstream fout("work.out");

for (int x = 0; x <= 441; x++)

{

int stroka = x % 21;

int stolbec = x / 21;

if (str[x] == z)

{

nast[stolbec][stroka] = 1;

}

else if (str[x] != z)

{

nast[stolbec][stroka] = 0;

}

}

for (int o = 0; o <= 20; o++)

{

for (int p = 0; p <= 21; p++)

{

if (p != 21)

{

fout << nast[o][p] << " ";

}

else

fout << endl;

}

}

int k = 2;

int mertvie[21][21] = { 0 };

while (k <= pokolenie)

{

for (int c = 0; c <= 20; c++)

{

for (int d = 0; d <= 21; d++)

{

int sosedi = 0;

if (nast[c][d] == 12)

{

budu[c][d] = 0;

mertvie[c][d] = 1;

}

if (nast[c][d] < 12 and nast[c][d] > 0 and nast[c][d] != 12)

{

if (d - 1 > 0) {

if (nast[c][d - 1]) {

sosedi++;

}

}

if (c + 1 < 20 && d - 1 > 0) {

if (nast[c + 1][d - 1]) {

sosedi++;

}

}

if (c + 1 < 20) {

if (nast[c + 1][d]) {

sosedi++;

}

}

if (c + 1 < 20 && d + 1 < 21) {

if (nast[c + 1][d + 1]) {

sosedi++;

}

}

if (d + 1 < 21) {

if (nast[c][d + 1]) {

sosedi++;

}

}

if (c - 1 > 0 && d + 1 < 21) {

if (nast[c - 1][d + 1]) {

sosedi++;

}

}

if (c - 1 > 0) {

if (nast[c - 1][d]) {

sosedi++;

}

}

if (c - 1 > 0 && d - 1 > 0) {

if (nast[c - 1][d - 1]) {

sosedi++;

}

}

if (sosedi == 2 or sosedi == 3)

{

budu[c][d] =nast[c][d] +1;

sosedi = 0;

}

else

{

budu[c][d] = 0;

mertvie[c][d] = 1;

sosedi = 0;

}

}

}

}

for (int y = 0; y < 20; y++)

{

for (int e = 0; e <= 21; e++)

{

if (mertvie[y][e] == 1 and nast[y][e] == 0)

{

budu[y][e] = 1;

mertvie[y][e] = 0;

}

}

}

int live = 0;

for (int i = 0; i < 20; i++)

{

for (int j = 0; j <= 21; j++)

{

if (budu[i][j] > 0)

{

live++;

}

}

}

if (live > 0)

{

for (int r = 0; r <= 20; r++)

{

for (int s = 0; s <= 21; s++)

{

if (s != 21)

{

fout << budu[r][s] << " ";

}

else

fout << endl;

}

}

}

else

{

cout << "Популяция погибла" << endl;

break;

}

for (int q = 0; q <= 20; q++)

{

for (int t = 0; t <= 21; t++)

{

nast[q][t] = budu[q][t];

}

}

budu[21][21] = { 0 };

k++;

}

int q = 0; int w = 1;

ifstream fin("work.out");

cout << "Поколение 1:" << endl;

while (fin)

{

string strinput;

fin >> strinput;

cout << strinput << " ";

if (w <= pokolenie and q < 441 \* pokolenie - 2)

{

q++;

if (q % 21 == 0)

{

cout << endl;

}

if (q % 441 == 0)

{

w++;

cout << "Поколение " << w << ":" << endl;

}

}

}

}

else

cout << "Вы ввели не число" << endl;

}