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 №2
«Модель Ва-тор»

Выполнил: студент 585гр.

Роженцев А.К. _____

Проверил: ст.пр.

Уланов П.Н. _____

Барнаул 2020

1 Цель работы

Написать программу, реализующую модель Ва-Тор, с возможностью сохранения в файл количества рыб и акул на каждом шаге. Выполнить исследование поведения популяций рыб и акул.

2 Описание модели

Рыбы: Начальное количество рыб и акул помещается случайным образом в узлы прямоугольной сетки.Всем рыбам и акулам приписывается случайный возраст. На очередном временном шаге рассматривается по очереди каждая рыба. Определяется число ближайших незанятых соседних узлов и рыба передвигается в один из незанятых узлов случайным образом. Если все узлы заняты, рыба не перемещается. Акулы: На очередном временном шаге рассматривается по очереди каждая акула. Если все ближайшие к акуле соседние узлы свободны, она перемещается в один из них случайным образом. Если хоть в одном из них находится рыба, акула перемещается в такой узел случайным образом и съедает рыбу. Если за Na шагов акула ничего не съедает, то она погибает. Если акула выживает в течение Ma шагов, у нее появляется потомок. Новая акула помещается в предыдущую позицию родителя.

3 Программа

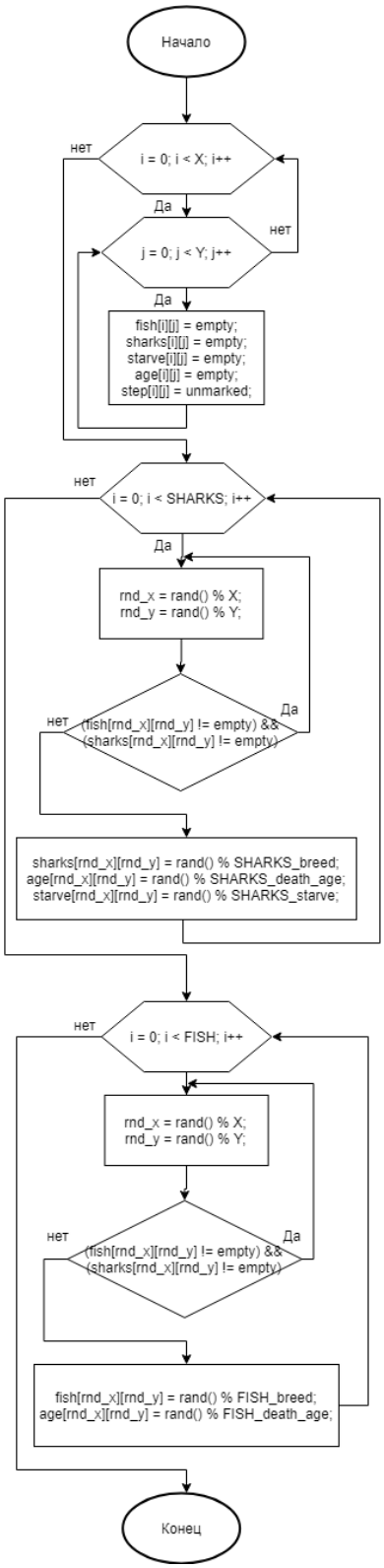

Рис. 1: Подпрограмма initialization

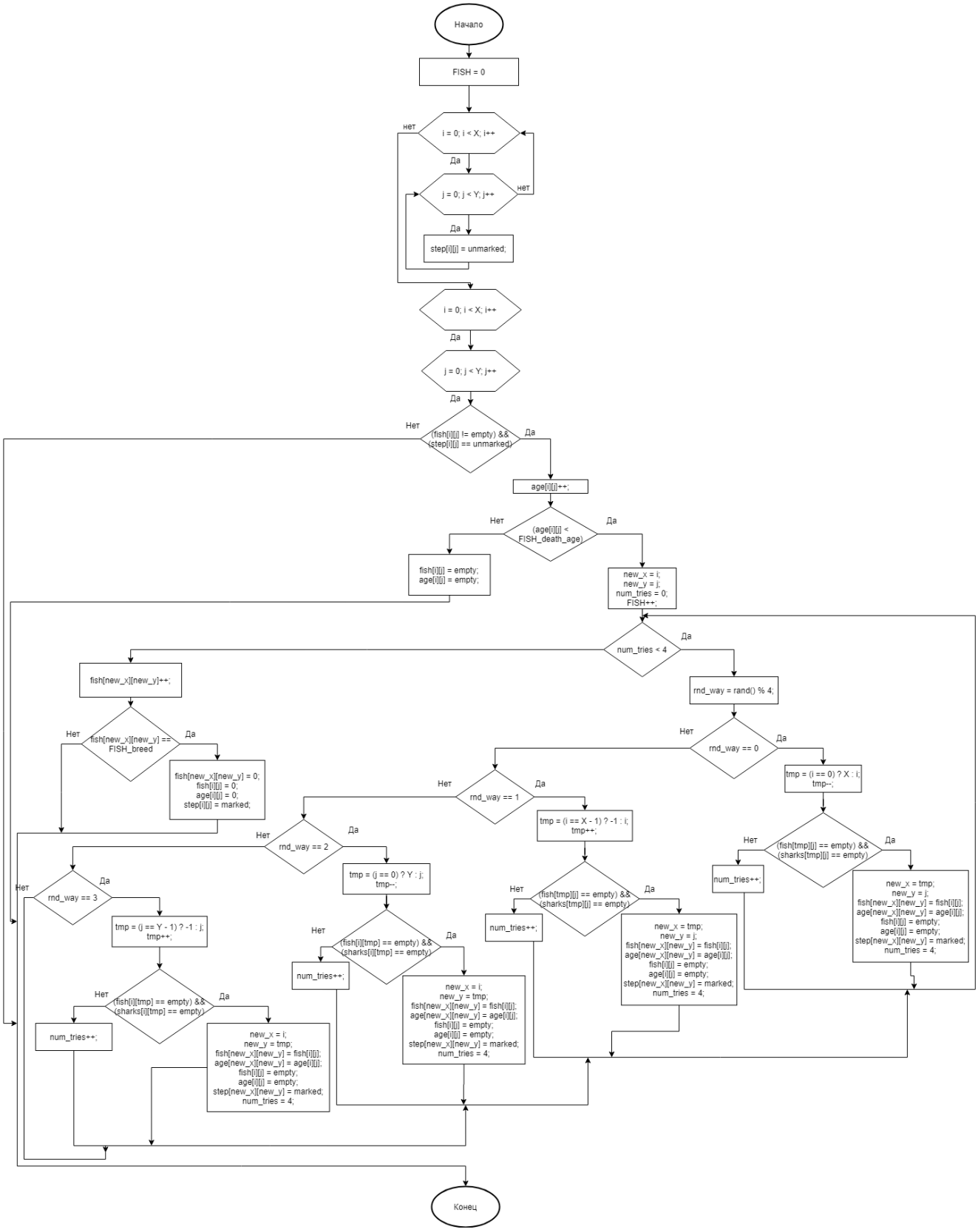

Рис. 2: Подпрограмма Fishstep

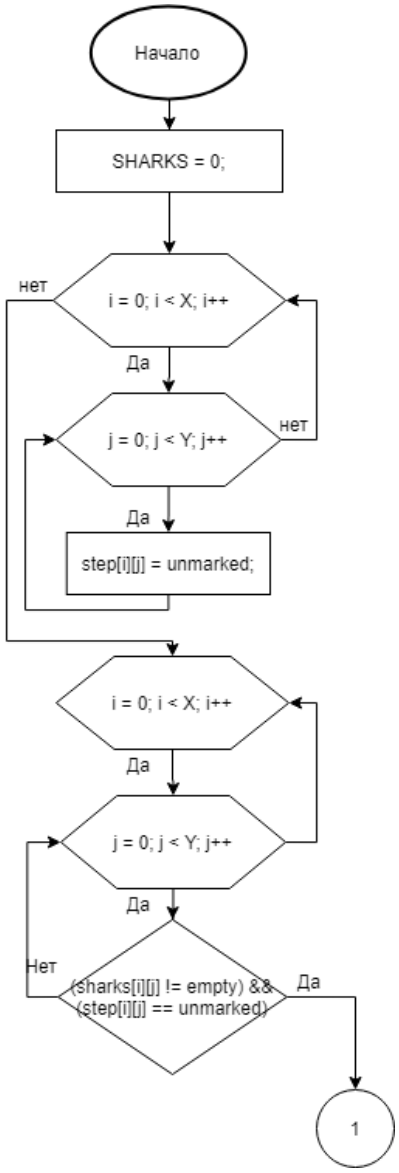

Рис. 3: Подпрограмма Sharkstep

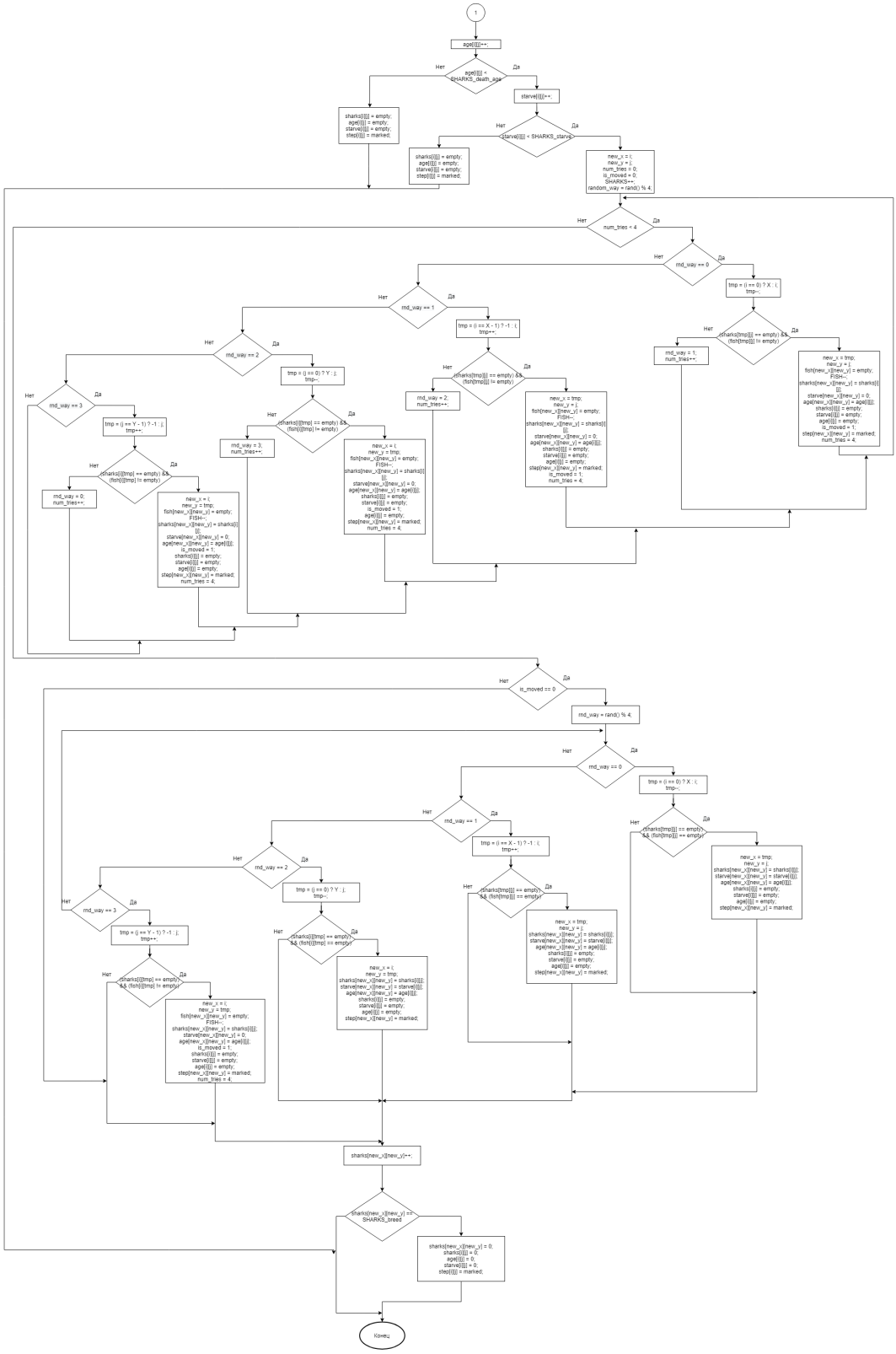

Рис. 4: Подпрограмма Sharkstep

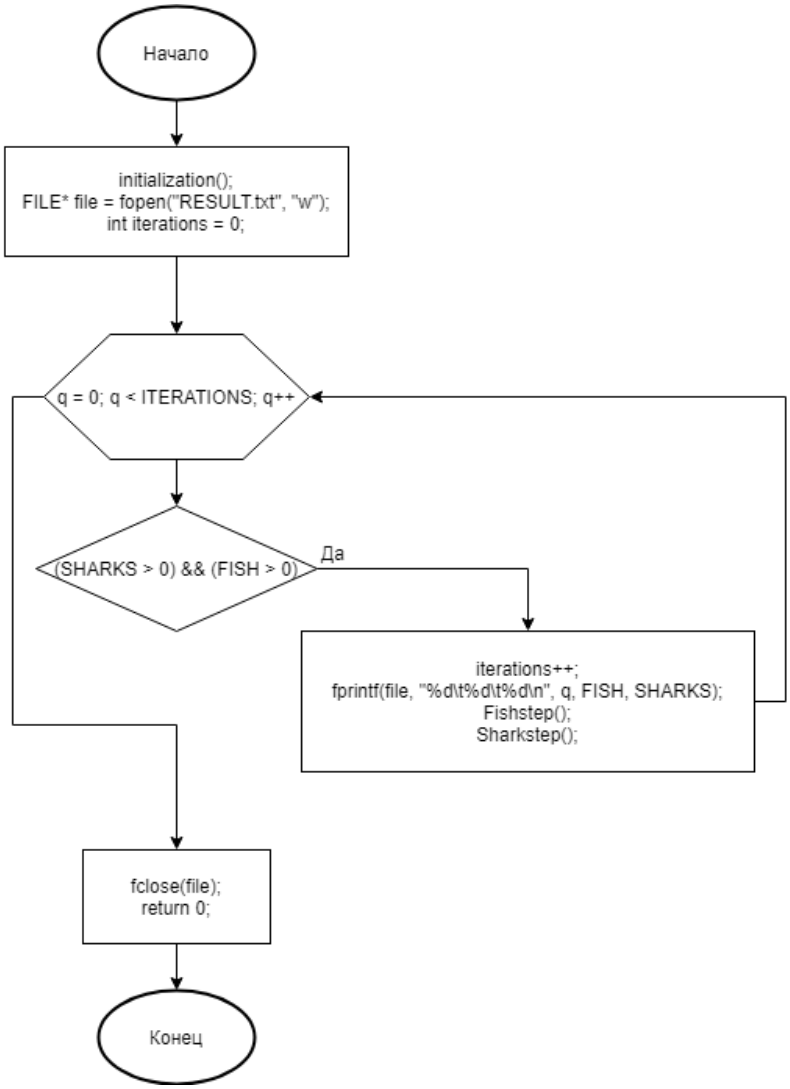

Рис. 5: Подпрограмма main

Зависимость численности акул и рыб от времени получилась следующая.

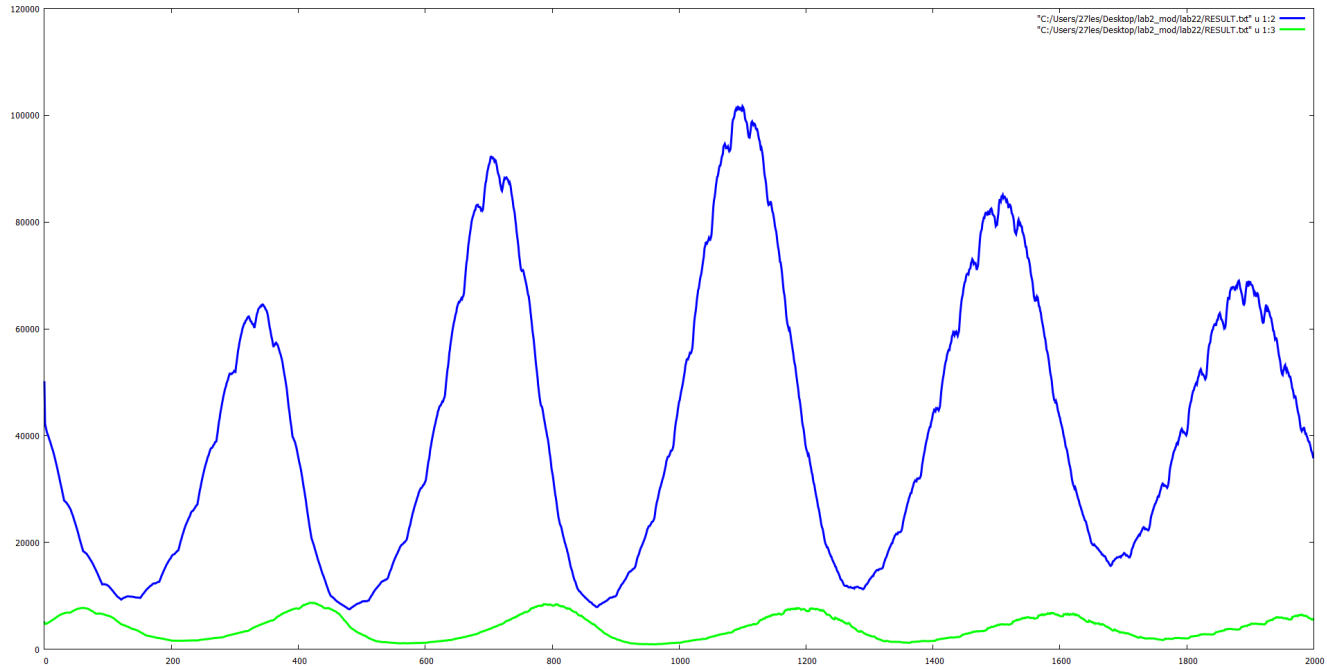

Рис. 6: Синие - рыбы, зеленые - акулы

Приложение

```
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define ITERATIONS 2000
#define X 500
#define Y 500
int FISH = 50000;
int SHARKS = 5000;
int FISH_breed = 30;
int SHARKS_breed = 40;
int SHARKS_starve = 25;
int FISH_death_age = 100;
int SHARKS_death_age = 200;
int step[X][Y];
int fish[X][Y];
int sharks[X][Y];
int starve[X][Y];
```

```
19 int age[X][Y];
20 int i, j;
21 enum { empty = -1, marked, unmarked };
22 void initialization()
23 {
24 int random_x, random_y;
25 for (i = 0; i < X; i++) {
26 for (j = 0; j < Y; j++) {
27 fish[i][j] = empty;
28 sharks[i][j] = empty;
29 starve[i][j] = empty;
30 age[i][j] = empty;
31 step[i][j] = unmarked;
32 }
33 }
34 for (i = 0; i < SHARKS; i++) {
35 do {
36 random_x = rand() % X;
37 random_y = rand() % Y;
38 } while ((fish[random_x][random_y] != empty) &&
39 (sharks[random_x][random_y] != empty));
40 sharks[random_x][random_y] = rand() % SHARKS_breed;
41 age[random_x][random_y] = rand() % SHARKS_death_age;
42 starve[random_x][random_y] = rand() % SHARKS_starve;
43 }
44 for (i = 0; i < FISH; i++) {
45 do {
46 random_x = rand() % X;
47 random_y = rand() % Y;
48 } while ((fish[random_x][random_y] != empty) &&
49 (sharks[random_x][random_y] != empty));
50 fish[random_x][random_y] = rand() % FISH_breed;
51 age[random_x][random_y] = rand() % FISH_death_age;
52 }
53 }
54 void Fishstep()
55 {
56 static int new_x, new_y;
57 static int random_way, num_tries, tmp;
58 FISH = 0;
59 for (i = 0; i < X; i++) {
```

```
for (j = 0; j < Y; j++) {
step[i][j] = unmarked;
}
}
for (i = 0; i < X; i++) {
for (j = 0; j < Y; j++) {
if ((fish[i][j] != empty) && (step[i][j] == unmarked)) {
age[i][j]++;
if ((age[i][j] < FISH_death_age)) {
new_x = i;
new_y = j;
num_tries = 0;
FISH++;
while (num_tries < 4) {
random_way = rand() % 4;
if (random_way == 0) {
tmp = (i == 0) ? X : i;
tmp--;
if ((fish[tmp][j] == empty) && (sharks[tmp][j] == empty)) {
new_x = tmp;
new_y = j;
fish[new_x][new_y] = fish[i][j];
age[new_x][new_y] = age[i][j];
fish[i][j] = empty;
age[i][j] = empty;
step[new_x][new_y] = marked;
num_tries = 4;
}
else {
num_tries++;
}
}
else if (random_way == 1) {
tmp = (i == X - 1) ? -1 : i;
tmp++;
if ((fish[tmp][j] == empty) && (sharks[tmp][j] == empty)) {
new_x = tmp;
new_y = j;
fish[new_x][new_y] = fish[i][j];
age[new_x][new_y] = age[i][j];
fish[i][j] = empty;
```

```
age[i][j] = empty;
step[new_x][new_y] = marked;
num_tries = 4;
}
else {
num_tries++;
}
}
else if (random_way == 2) {
tmp = (j == 0) ? Y : j;
tmp--;
if ((fish[i][tmp] == empty) && (sharks[i][tmp] == empty)) {
new_x = i;
new_y = tmp;
fish[new_x][new_y] = fish[i][j];
age[new_x][new_y] = age[i][j];
fish[i][j] = empty;
age[i][j] = empty;
step[new_x][new_y] = marked;
num_tries = 4;
}
else {
num_tries++;
}
}
else if (random_way == 3) {
tmp = (j == Y - 1) ? -1 : j;
tmp++;
if ((fish[i][tmp] == empty) && (sharks[i][tmp] == empty)) {
new_x = i;
new_y = tmp;
fish[new_x][new_y] = fish[i][j];
age[new_x][new_y] = age[i][j];
fish[i][j] = empty;
age[i][j] = empty;
step[new_x][new_y] = marked;
num_tries = 4;
}
else {
num_tries++;
}
```

```
142 }
143 }
144 fish[new_x][new_y]++;
145 if (fish[new_x][new_y] == FISH_breed) {
146 fish[new_x][new_y] = 0;
147 fish[i][j] = 0;
148 age[i][j] = 0;
149 step[i][j] = marked;
150 }
151 }
152 else {
153 fish[i][j] = empty;
154 age[i][j] = empty;
155 }
156 }
157 }
158 }
159 }
160 void Sharkstep()
161 {
162 static int new_x, new_y;
163 static int random_way, num_tries, tmp;
164 static int is_moved;
165 SHARKS = 0;
166 for (i = 0; i < X; i++) {
167 for (j = 0; j < Y; j++) {
168 step[i][j] = unmarked;
169 }
170 }
171 for (i = 0; i < X; i++) {
172 for (j = 0; j < Y; j++) {
173 if ((sharks[i][j] != empty) && (step[i][j] == unmarked)) {
174 age[i][j]++;
175 if (age[i][j] < SHARKS_death_age) {
176 starve[i][j]++;
177 if (starve[i][j] < SHARKS_starve) {
178 new_x = i;
179 new_y = j;
180 num_tries = 0;
181 is_moved = 0;
182 SHARKS++;
```

```
183 random_way = rand() % 4;
184 while (num_tries < 4) {
185 if (random_way == 0) {
186 tmp = (i == 0) ? X : i;
187 tmp--;
188 if ((sharks[tmp][j] == empty) && (fish[tmp][j] != empty)) {
189 new_x = tmp;
190 new_y = j;
191 fish[new_x][new_y] = empty;
192 FISH--;
193 sharks[new_x][new_y] = sharks[i][j];
194 starve[new_x][new_y] = 0;
195 age[new_x][new_y] = age[i][j];
196 sharks[i][j] = empty;
197 starve[i][j] = empty;
198 age[i][j] = empty;
199 is_moved = 1;
200 step[new_x][new_y] = marked;
201 num_tries = 4;
202 }
203 else {
204 random_way = 1;
205 num_tries++;
206 }
207 }
208 else if (random_way == 1) {
209 tmp = (i == X - 1) ? -1 : i;
210 tmp++;
211 if ((sharks[tmp][j] == empty) && (fish[tmp][j] != empty)) {
212 new_x = tmp;
213 new_y = j;
214 fish[new_x][new_y] = empty;
215 FISH--;
216 sharks[new_x][new_y] = sharks[i][j];
217 starve[new_x][new_y] = 0;
218 age[new_x][new_y] = age[i][j];
219 sharks[i][j] = empty;
220 starve[i][j] = empty;
221 age[i][j] = empty;
222 step[new_x][new_y] = marked;
223 is_moved = 1;
```

```
num_tries = 4;
}
else {
random_way = 2;
num_tries++;
}
}
else if (random_way == 2) {
tmp = (j == 0) ? Y : j;
tmp--;
if ((sharks[i][tmp] == empty) && (fish[i][tmp] != empty)) {
new_x = i;
new_y = tmp;
fish[new_x][new_y] = empty;
FISH--;
sharks[new_x][new_y] = sharks[i][j];
starve[new_x][new_y] = 0;
age[new_x][new_y] = age[i][j];
sharks[i][j] = empty;
starve[i][j] = empty;
is_moved = 1;
age[i][j] = empty;
step[new_x][new_y] = marked;
num_tries = 4;
}
else {
random_way = 3;
num_tries++;
}
}
else if (random_way == 3) {
tmp = (j == Y - 1) ? -1 : j;
tmp++;
if ((sharks[i][tmp] == empty) && (fish[i][tmp] != empty)) {
new_x = i;
new_y = tmp;
fish[new_x][new_y] = empty;
FISH--;
sharks[new_x][new_y] = sharks[i][j];
starve[new_x][new_y] = 0;
age[new_x][new_y] = age[i][j];
```

```
265 is_moved = 1;
266 sharks[i][j] = empty;
267 starve[i][j] = empty;
268 age[i][j] = empty;
269 step[new_x][new_y] = marked;
270 num_tries = 4;
271 }
272 else {
273 random_way = 0;
274 num_tries++;
275 }
276 }
277 }
278 if (is_moved == 0) {
279 random_way = rand() % 4;
280 if (random_way == 0) {
281 tmp = (i == 0) ? X : i;
282 tmp--;
283 if ((sharks[tmp][j] == empty) && (fish[tmp][j] == empty)) {
284 new_x = tmp;
285 new_y = j;
286 sharks[new_x][new_y] = sharks[i][j];
287 starve[new_x][new_y] = starve[i][j];
288 age[new_x][new_y] = age[i][j];
289 sharks[i][j] = empty;
290 starve[i][j] = empty;
291 age[i][j] = empty;
292 step[new_x][new_y] = marked;
293 }
294 }
295 else if (random_way == 1) {
296 tmp = (i == X - 1) ? -1 : i;
297 tmp++;
298 if ((sharks[tmp][j] == empty) && (fish[tmp][j] == empty)) {
299 new_x = tmp;
300 new_y = j;
301 sharks[new_x][new_y] = sharks[i][j];
302 starve[new_x][new_y] = starve[i][j];
303 age[new_x][new_y] = age[i][j];
304 sharks[i][j] = empty;
305 starve[i][j] = empty;
```

```
306 age[i][j] = empty;
307 step[new_x][new_y] = marked;
308 }
309 }
310 else if (random_way == 2) {
311 tmp = (j == 0) ? Y : j;
312 tmp--;
313 if ((sharks[i][tmp] == empty) && (fish[i][tmp] == empty)) {
314 new_x = i;
315 new_y = tmp;
316 sharks[new_x][new_y] = sharks[i][j];
317 starve[new_x][new_y] = starve[i][j];
318 age[new_x][new_y] = age[i][j];
319 sharks[i][j] = empty;
320 starve[i][j] = empty;
321 age[i][j] = empty;
322 step[new_x][new_y] = marked;
323 }
324 }
325 else if (random_way == 3) {
326 tmp = (j == Y - 1) ? -1 : j;
327 tmp++;
328 if ((sharks[i][tmp] == empty) && (fish[i][tmp] == empty)) {
329 new_x = i;
330 new_y = tmp;
331 sharks[new_x][new_y] = sharks[i][j];
332 starve[new_x][new_y] = starve[i][j];
333 age[new_x][new_y] = age[i][j];
334 sharks[i][j] = empty;
335 starve[i][j] = empty;
336 age[i][j] = empty;
337 step[new_x][new_y] = marked;
338 }
339 }
340 }
341 sharks[new_x][new_y]++;
342 if (sharks[new_x][new_y] == SHARKS_breed) {
343 sharks[new_x][new_y] = 0;
344 sharks[i][j] = 0;
345 age[i][j] = 0;
346 starve[i][j] = 0;
```

```
347 step[i][j] = marked;
348 }
349 }
350 else {
351 sharks[i][j] = empty;
352 age[i][j] = empty;
353 starve[i][j] = empty;
354 step[i][j] = marked;
355 }
356 }
357 else {
358 sharks[i][j] = empty;
359 age[i][j] = empty;
360 starve[i][j] = empty;
361 step[i][j] = marked;
362 }
363 }
364 }
365 }
366 }
367 int main(int argc, char** argv)
368 {
369 srand(time(NULL));
370 initialization();
371 printf("\nCalculating...\n");
372 FILE* file = fopen("RESULT.txt", "w");
373 int iterations = 0;
374 for (int q = 0; q < ITERATIONS; q++) {
375 if ((SHARKS > 0) && (FISH > 0)) {
376 iterations++;
377 fprintf(file, "%d\t%d\t%d\n", q, FISH, SHARKS);
378 Fishstep();
379 Sharkstep();
380 }
381 }
382 fclose(file);
383 return 0;
384 }
```