МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра информационных систем

Отчет по дисциплине Основы алгоритмизации и программирование

Домашняя работа №1

Выполнил студент гр. ИСЭбд-11

Гапон Николай Иванович

Преподаватель

Власенко Олег Федосович

ССЫЛКА НА GIT <https://github.com/NikGapon/1year/tree/main/lab8>

Домашнее задание

Добавить еще 3-5 действий при нажатии на выбранные вами клавиши!

void moveToLeft() {

int i, j;

i = 0;

while (i < N) {

j = 1;

while (j < M) {

if (a[i][j] == 3) {

if (a[i][j - 1] == 0) {

a[i][j - 1] = 3;

a[i][j] = 0;

steps++;

}

else if (a[i][j - 1] == 1) {

a[i][j - 1] = 3;

a[i][j] = 0;

steps++;

gold++;

}

}

j++;

}

i++;

}

}

void moveToRight() {

int i = 0;

while (i < N) {

int j = M - 2;

while (j >= 0) {

if (a[i][j] == 3) {

if (a[i][j + 1] == 0) {

a[i][j + 1] = 3;

a[i][j] = 0;

steps++;

}

else if (a[i][j + 1] == 1) {

a[i][j + 1] = 3;

a[i][j] = 0;

steps++;

gold++;

}

}

j--;

}

i++;

}

}

void moveUp() {

int i = 1;

while (i < N) {

int j = 0;

while (j < M) {

if (a[i][j] == 3) {

if (a[i - 1][j] == 0) {

a[i - 1][j] = 3;

a[i][j] = 0;

steps++;

}

else if (a[i - 1][j] == 1) {

a[i - 1][j] = 3;

a[i][j] = 0;

steps++;

gold++;

}

}

j++;

}

i++;

}

}

void UPST() {

int i = 0;

while (i < N) {

int j = 0;

while (j < M) {

if (a[i][j] == 3) {

a[i - 1][j] = 2;

steps++;

}

j++;

}

i++;

}

}

void UPST\_0() {

int i = 0;

while (i < N) {

int j = 0;

while (j < M) {

if (a[i][j] == 3) {

a[i - 1][j] = 0;

steps++;

}

j++;

}

i++;

}

}

void moveDown() {

int i = N;

while (i >= 0) {

int j = 0;

while (j < M) {

if (a[i][j] == 3) {

if (a[i + 1][j] == 0) {

a[i + 1][j] = 3;

a[i][j] = 0;

steps++;

}

else if (a[i + 1][j] == 1) {

a[i + 1][j] = 3;

a[i][j] = 0;

steps++;

gold++;

}

}

j++;

}

i--;

}

}

int sizeX = 36;

int sizeY = 30;

void DrawField(HDC hdc) {

HBRUSH hBrushEmptyCell; //создаём кисть для пустого поля

hBrushEmptyCell = CreateSolidBrush(RGB(200, 200, 200)); // серый

HBRUSH hBrushGold; //создаём кисть для поля с золотом

hBrushGold = CreateSolidBrush(RGB(255, 255, 0)); // желтый

HBRUSH hBrushWall; //создаём кисть для стены

hBrushWall = CreateSolidBrush(RGB(0, 0, 0)); // черный

HBRUSH hBrushMan; //создаём кисть для игрока

hBrushMan = CreateSolidBrush(RGB(0, 0, 255)); // синий

int i, j;

i = 0;

while (i < N) {

j = 0;

while (j < M) {

RECT rect = { j \* sizeX,i \* sizeY, (j + 1) \* sizeX,(i + 1) \* sizeY };

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

FillRect(hdc, &rect, hBrushEmptyCell);

}

else if (a[i][j] == 1) {

FillRect(hdc, &rect, hBrushGold);

}

else if (a[i][j] == 2) {

FillRect(hdc, &rect, hBrushWall);

}

else if (a[i][j] == 3) {

FillRect(hdc, &rect, hBrushMan);

}

else {

// тут никогда не должны оказаться

}

j++;

}

i++;

}

HFONT hFont;

hFont = CreateFont(20,

0, 0, 0, 0, 0, 0, 0,

DEFAULT\_CHARSET,

0, 0, 0, 0,

L"Courier New"

);

SelectObject(hdc, hFont);

SetTextColor(hdc, RGB(0, 128, 128));

TCHAR string1[] = \_T("сделано ходов:");

TCHAR string2[] = \_T("собрано золота:");

TextOut(hdc, 10, sizeY \* (N + 1), (LPCWSTR)string1, \_tcslen(string1));

TextOut(hdc, 10, sizeY \* (N + 1) + 20, (LPCWSTR)string2, \_tcslen(string2));

char sSteps[5];

TCHAR tsSteps[5];

sprintf(sSteps, "%d", steps);

OemToChar(sSteps, tsSteps);

TextOut(hdc, 220, sizeY \* (N + 1), (LPCWSTR)tsSteps, \_tcslen(tsSteps));

char sGold[5];

TCHAR tsGold[5];

sprintf(sGold, "%d", gold);

OemToChar(sGold, tsGold);

TextOut(hdc, 220, sizeY \* (N + 1) + 20, (LPCWSTR)tsGold, \_tcslen(tsGold));

DeleteObject(hFont);

DeleteObject(hBrushEmptyCell);

DeleteObject(hBrushGold);

DeleteObject(hBrushWall);

DeleteObject(hBrushMan);

}

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

switch (message)

{

case WM\_PAINT:

{

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hWnd, &ps);

DrawField(hdc);

EndPaint(hWnd, &ps);

}

break;

case WM\_KEYDOWN:

switch (wParam)

{

case VK\_DOWN:

moveDown();

InvalidateRect(hWnd, NULL, TRUE);

break;

case VK\_LEFT:

moveToLeft();

InvalidateRect(hWnd, NULL, TRUE);

break;

case VK\_UP:

moveUp();

InvalidateRect(hWnd, NULL, TRUE);

break;

case VK\_RIGHT:

moveToRight();

InvalidateRect(hWnd, NULL, TRUE);

break;

case VK\_TAB:

UPST();

InvalidateRect(hWnd, NULL, TRUE);

break;

case VK\_CAPITAL:

UPST\_0();

InvalidateRect(hWnd, NULL, TRUE);

break;

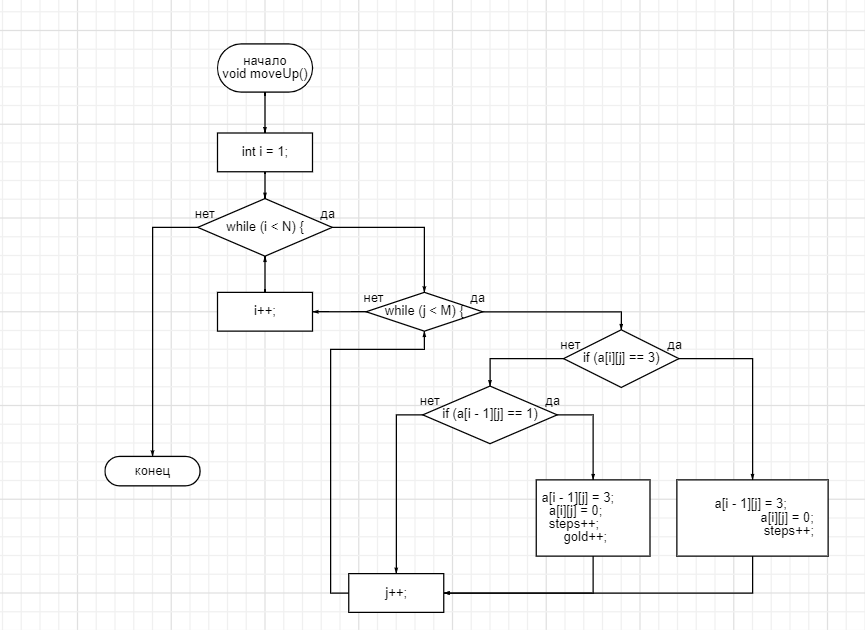
}

break;

return 0;

}

}



ВЫВОД

В этой лабораторной работе мы детально познакомились с двухмерными масивами

И развили навыки с WM\_KEYDOWN