

# Основы программирования

ФИСТ 1 курс

Власенко

Олег

Федосович

Лекция 10.

Двухмерные массивы. FOR. BREAK. Работа с файлами.

Простая игра на двумерном массиве

# 2D массив

```
int a0[3];
```

```
int a1[3];
```

```
int arr[2][3];
```

```
int a0_1[3] = {1, 2, 3};
```

```
int a1_1[] = {10, 20, 30};
```

```
int arr1[2][3] = {{ 1, 2, 3}, {10, 20, 30}};
```

# 2D массив – размещение в памяти

```
void main()
{
    int len = sizeof(int);
    int arr1[2][3] = { {1, 2, 3}, {10, 20, 30} };

    int * p00 = &arr1[0][0];
    int * p01 = &arr1[0][1];
    int * p02 = &arr1[0][2];
    int * p10 = &arr1[1][0];
    int * p11 = &arr1[1][1];
    int * p12 = &arr1[1][2];
}
```

# 2D массив – размещение в памяти (2)

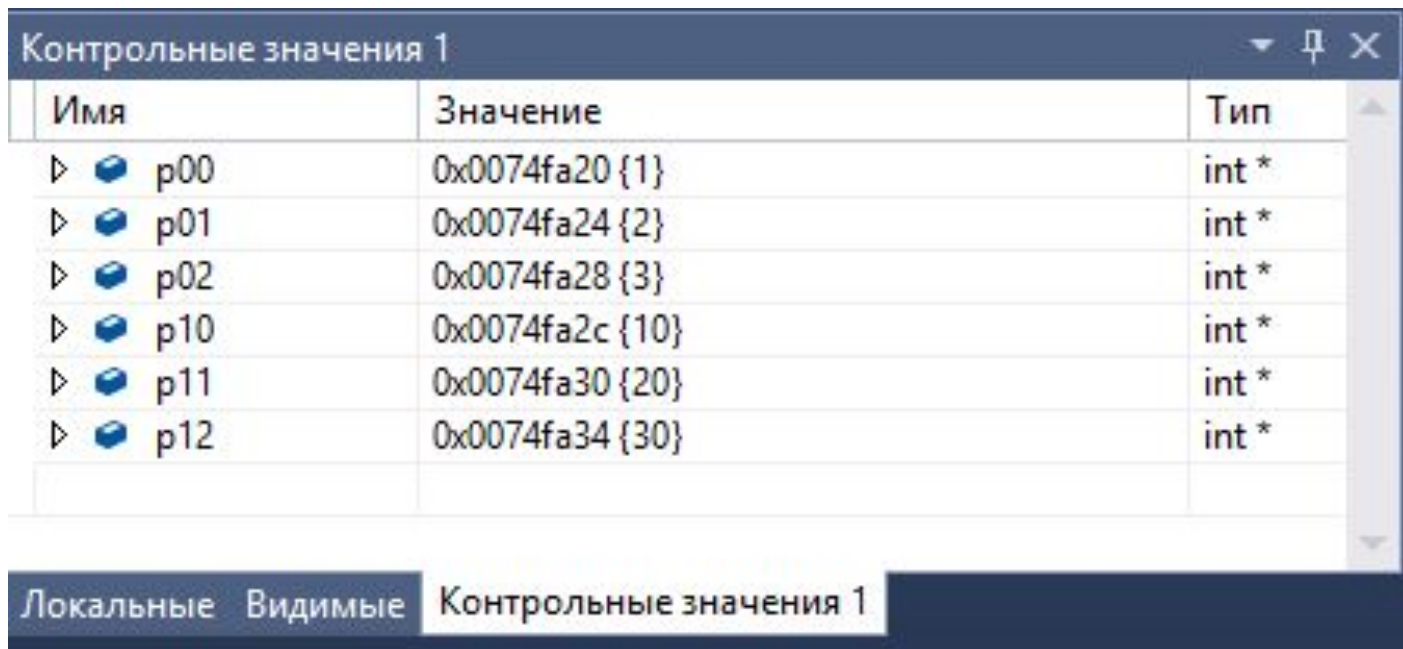
```
int main()
```

```
{
```

```
    int len = sizeof(int);
```

```
    int arr1[2][3] = { {1, 2, 3}, {10, 20, 30} };
```

```
    ...
```



Контрольные значения 1

Имя	Значение	Тип
p00	0x0074fa20 {1}	int *
p01	0x0074fa24 {2}	int *
p02	0x0074fa28 {3}	int *
p10	0x0074fa2c {10}	int *
p11	0x0074fa30 {20}	int *
p12	0x0074fa34 {30}	int *

Локальные Видимые Контрольные значения 1

# Вывод элементов 2D массива

```
int i = 0; // счетчик по строкам
```

```
while (i < 2) {
```

```
    int j = 0; // счетчик по столбцам
```

```
    while (j < 3) {
```

```
        printf("%5d ", arr1[i][j]);
```

```
        j++;
```

```
    }
```

```
    printf("\n");
```

```
    i++;
```

```
}
```

# Ввод элементов 2D массива

```
#define _CRT_SECURE_NO_WARNINGS
```

```
...
```

```
int i = 0;
```

```
while (i < 2) {
```

```
    int j = 0;
```

```
    while (j < 3) {
```

```
        scanf("%d", &arr1[i][j]);
```

```
        j++;
```

```
    }
```

```
    i++;
```

```
}
```

# Подсчет суммы элементов массива

```
int s = 0;
i = 0;
while (i < 2) {
    int j = 0;
    while (j < 3) {
        s += arr1[i][j];
        j++;
    }
    i++;
}
```

# Увеличение всех нечетных элементов в 10 раз

```
i = 0;
while (i < 2) {
    int j = 0;
    while (j < 3) {
        if (arr1[i][j] % 2 == 1) {
            arr1[i][j] *= 10;
        }

        j++;
    }
    i++;
}
```



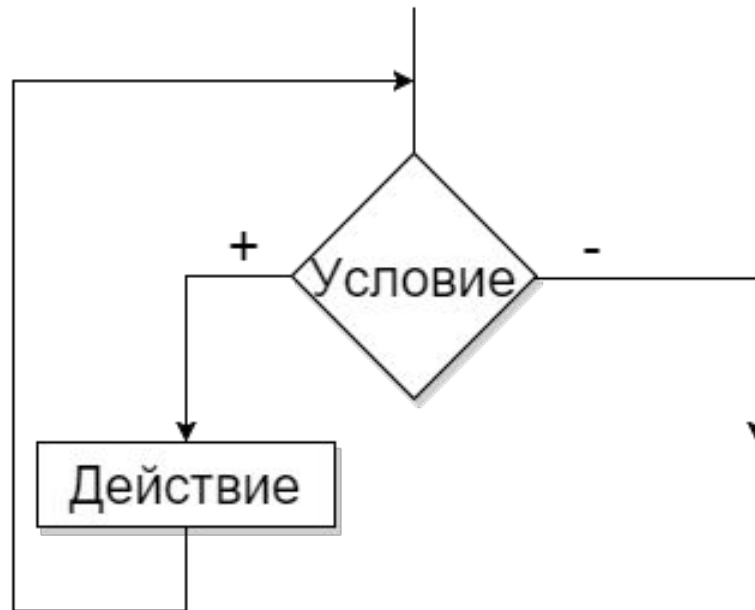
# Поиск минимального элемента

```
int min = a[0][0];
int iMin = 0;
int jMin = 0;
i = 0;
while (i < 2) {
    j = 0;
    while (j < 3) {
        if (a[i][j] < min) {
            min = a[i][j];
            iMin = i;
            jMin = j;
        }
        j++;
    }
    i++;
}
```

**FOR**

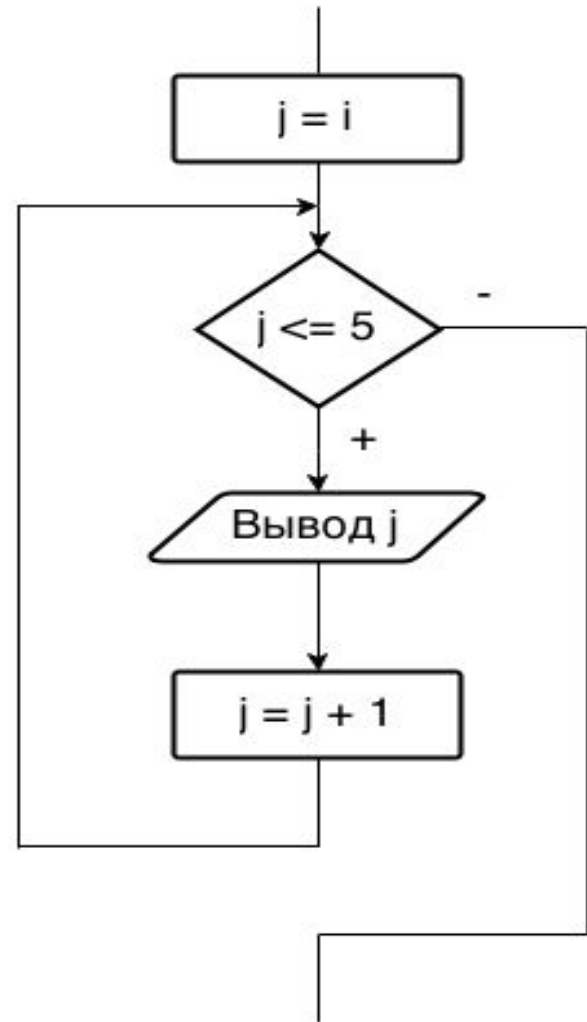
# Цикл с предусловием while

```
while (Условие) {  
    Действие;  
}
```



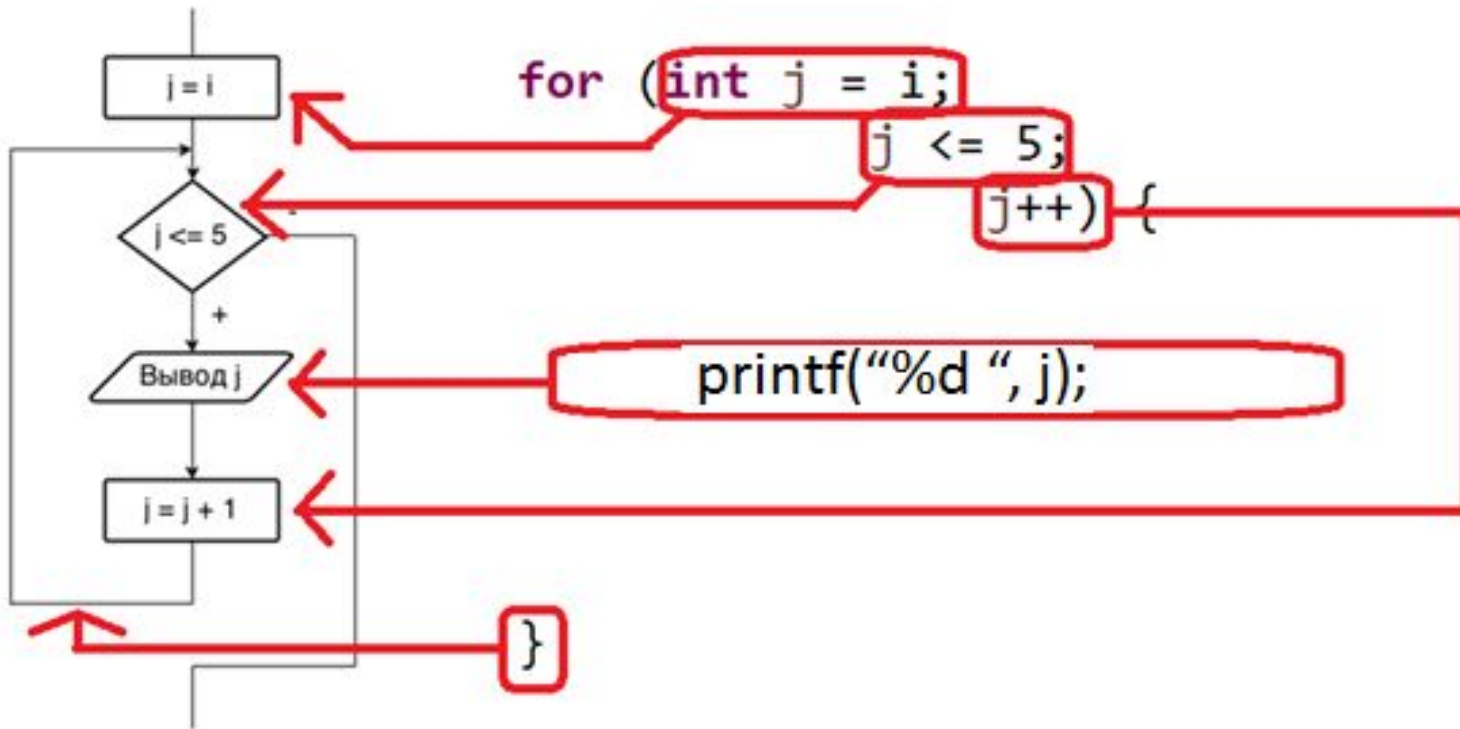
# Цикл for

```
int j = i; // инициализация счетчика цикла
while (j <= 5) { // условие продолжения цикла
    printf("%d ", j);
    j++; // изменение счетчика цикла
}
```



# Цикл for (2)

```
for (int j = i; j <= 5; j++) {  
    printf("%d ", j);  
}
```



# Цикл for – рисуем блок-схему!

```
f = 1;  
for (i = 1; i <= n; i++) {  
    f = f * i;  
}
```

# Цикл for – рисуем блок-схему и трассируем!

```
f = 1;  
for (i = 1; i <= n; i++) {  
    f = f * i;  
}
```

```
f = 1;  
i = 1;  
while (i <= n) {  
    f = f * i;  
    i++;  
}
```

# Какие варианты являются синтаксически некорректными?

1) for (;;) {printf("Hi");}

2) for (i=0;;) {printf("Hi");}

3) for (;i<n;) {printf("Hi");}

4) for (;i++) {printf("Hi");}

5) for (i=0;;i++) {printf("Hi");}

6) for (;i<n;i++) {printf("Hi");}

7) for (i=0;i<n;) {printf("Hi");}

8) for (i=0,j=10;i<j;i++,j--) {printf("Hi");}

9) for (i=0,j=10;i<j;i++,j--, printf("Hi"));

10) for (i = 0, j = 10, printf("Ups"); i < j; i++, j--, printf("Hi"));

11) for (;;);



# Вывод элементов 2D массива (WHILE)

```
int i = 0; // счетчик по строкам
```

```
while (i < 2) {
```

```
    int j = 0; // счетчик по столбцам
```

```
    while (j < 3) {
```

```
        printf("%5d ", arr1[i][j]);
```

```
        j++;
```

```
    }
```

```
    printf("\n");
```

```
    i++;
```

```
}
```

# Вывод элементов 2D массива (FOR)

```
for (int i = 0; i < 2; i++) {  
    for (int j = 0; j < 3; j++) {  
        printf("%5d ", arr1[i][j]);  
    }  
    printf("\n");  
}
```

# Ввод элементов 2D массива (WHILE)

```
#define _CRT_SECURE_NO_WARNINGS
```

```
...
```

```
int i = 0;
```

```
while (i < 2) {
```

```
    int j = 0;
```

```
    while (j < 3) {
```

```
        scanf("%d", &arr1[i][j]);
```

```
        j++;
```

```
    }
```

```
    i++;
```

```
}
```

# Ввод элементов 2D массива (FOR)

```
#define _CRT_SECURE_NO_WARNINGS
```

```
...
```

```
for (int i = 0; i < 2; i++) {  
    for (int j = 0; j < 3; j++) {  
        scanf("%d", &arr1[i][j]);  
    }  
}
```

# Подсчет суммы элементов массива (WHILE)

```
int s = 0;
i = 0;
while (i < 2) {
    int j = 0;
    while (j < 3) {
        s += arr1[i][j];
        j++;
    }
    i++;
}
```

# Подсчет суммы элементов массива (FOR)

```
int s = 0;  
for (int i = 0; i < 2; i++) {  
    for (int j = 0; j < 3; j++) {  
        s += arr1[i][j];  
    }  
}
```

# Увеличение всех нечетных элементов в 10 раз (WHILE)

```
i = 0;
while (i < 2) {
    int j = 0;
    while (j < 3) {
        if (arr1[i][j] % 2 == 1) {
            arr1[i][j] *= 10;
        }

        j++;
    }
    i++;
}
```

# Увеличение всех нечетных элементов в 10 раз (FOR)

```
for (int i = 0; i < 2; i++) {  
    for (int j = 0; j < 3; j++) {  
        if (arr1[i][j] % 2 == 1) {  
            arr1[i][j] *= 10;  
        }  
    }  
}
```



# Поиск минимального элемента (WHILE)

```
int min = a[0][0];
int iMin = 0;
int jMin = 0;
i = 0;
while (i < 2) {
    j = 0;
    while (j < 3) {
        if (a[i][j] < min) {
            min = a[i][j];
            iMin = i;
            jMin = j;
        }
        j++;
    }
    i++;
}
```

# Поиск минимального элемента (FOR)

```
int min = a[0][0];
int iMin = 0;
int jMin = 0;
for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 3; j++) {
        if (a[i][j] < min) {
            min = a[i][j];
            iMin = i;
            jMin = j;
        }
    }
}
```



# файл

<https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB>

**Файл** ([англ.](#) *file*) — именованная область данных на [носителе информации.](#)

# Текстовый файл

Текстовый файл содержит последовательность СИМВОЛОВ (в основном печатных знаков, принадлежащих тому или иному набору СИМВОЛОВ). Эти символы обычно сгруппированы в строки (англ. *lines, rows*). В современных системах строки разделяются разделителями строк

[https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9\\_%D1%84%D0%B0%D0%B9%D0%BB](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB)

# Работа с файлом – общий алгоритм

- 1) Открыть файл
- 2) Работать с файлом
- 3) Закрывать файл

**Задача 1 – прочитать из файла 2 целых  
числа, подсчитать их сумму, вывести в  
другой файл**

// Чтение из входного файла

// Обработка

// Запись в выходной файл

Входной файл:

3 12

Выходной файл:

15

# Задача 1 – прочитать из файла 2 целых числа, подсчитать их сумму, вывести в другой файл

// Чтение из входного файла

```
FILE *fin;
```

```
int a, b, s;
```

```
fin = fopen("c:\\Temp\\Files\\in1.txt", "rt");
```

```
if (fin == NULL) {
```

```
    printf("File in1.txt is not found");
```

```
    return;
```

```
}
```

```
fscanf(fin, "%d%d", &a, &b);
```

```
fclose(fin);
```



# Задача 1 (2)

// Обработка

**s = a + b;**

# Задача 1 (3)

// Запись в выходной файл

```
FILE *fout;
```

```
fout = fopen("c:\\Temp\\Files\\out1.txt", "wt");
```

```
if (fout == NULL) {
```

```
    printf("File out1.txt cannot be created");
```

```
    return;
```

```
}
```

```
fprintf(fout, "s = %d", s);
```

```
fclose(fout);
```

## Задача 2

Ввести с клавиатуры массив из  $N$  строк по  $M$  элементов каждая ( $1 \leq N \leq 10$ ,  $1 \leq M \leq 10$ ).  $N$  и  $M$  вводятся с клавиатуры.

Переставить столбцы, содержащие минимальный и максимальный элементы.

Получившийся массив вывести в консоль и в файл "out.txt".

# Задача 2

```
// перестановка столбцов с минимальным и максимальным элементами
for (int i = 0; i < n; i++) {
    int tmp = a[i][jMin];
    a[i][jMin] = a[i][jMax];
    a[i][jMax] = tmp;
}
```

```
// Запись в выходной файл
FILE *fout = fopen("d:\\Temp\\out2.txt", "wt");
if (fout == NULL) {
    printf("File out2.txt cannot be created");
    return;
}
for (int i = 0; i < n; i++) {
    for (int j = 0; j < m; j++)
        fprintf(fout, "%5d ", a[i][j]);
    fprintf(fout, "\n");
}
fclose(fout);
```

## Задача 3

Загрузить из файла “in3.txt” массив из N строк по M элементов каждая ( $1 \leq N \leq 10$ ,  $1 \leq M \leq 10$ ). N и M вводятся с клавиатуры.

(Загруженный массив вывести в консоль для контроля).

Удалить столбцы, в которых есть хотя бы один четный элемент.

Получившийся массив вывести в консоль и в файл “out3.txt”.

# Задача 3 – загрузка из файла

```
// Чтение из входного файла
// Открытие файла
FILE *fin;
fin = fopen("d:\\Temp\\in3.txt", "rt");
if (fin == NULL) {
    printf("File in3.txt is not found");
    return;
}
```

```
// Ввод массива ИЗ ФАЙЛА
fscanf(fin, "%d", &n);
fscanf(fin, "%d", &m);
for (int i = 0; i < n; i++)
    for (int j = 0; j < m; j++)
        fscanf(fin, "%d", &a[i][j]);
fclose(fin);
```

# Задача 3 – удаление столбцов с четными элементами

```
for (j = 0; j < m; ) {
    int flag = 0;
    for (i = 0; i < n; i++)
        if (a[i][j] % 2 == 0) {
            flag = 1;
            break;
        }

    if (flag) {
        for (int j2 = j; j2 < m - 1; j2++)
            for (i = 0; i < n; i++)
                a[i][j2] = a[i][j2 + 1];
        m--;
    }
    else {
        j++;
    }
}
```

**BREAK**



```
#include <stdio.h>
```

```
void main() {  
    int i = 1;  
    int a = 1, b = 2, c = 3, d = 4, e = 5, f = 6;  
    do {  
        printf("%d ", i);  
        if (a < b) {  
            for (b = d; b < f; b++) {  
                a = c;  
                while (a < f) {  
                    d += a;  
                    a++;  
                }  
                c = a;  
            }  
            e += d;  
        }  
    }
```



```
else {  
    for (f = e; f > a; f--) {  
        if (c < a) {  
            c = a;  
            d++;  
            break;  
        }  
        f += a;  
    }  
    }  
    i++;  
} while (i <= 5);  
  
printf("%d %d %d", d, e, f);  
}
```



# Задача 4

Загрузить из файла “in4.txt” массив из N строк по M элементов каждая ( $1 \leq N \leq 10$ ,  $1 \leq M \leq 10$ ). N и M вводятся с клавиатуры.

(Загруженный массив вывести в консоль для контроля).

Продублировать строки, в которых есть отрицательные элементы.

Получившийся массив вывести в консоль и в файл “out4.txt”.

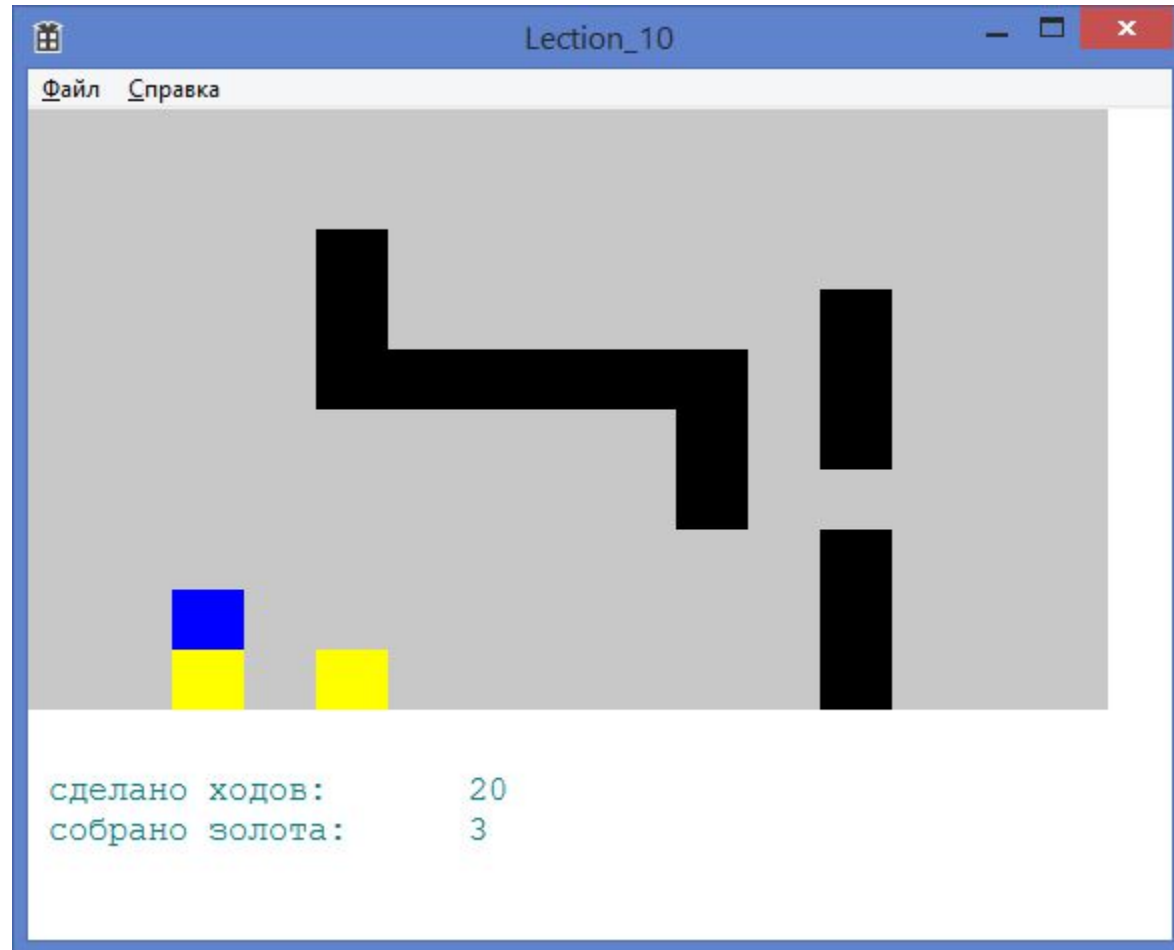
# Задача 4 – вставка строк

```
for (i = n - 1; i >= 0; i--) {
    int flag = 0;
    for (j = 0; j < m; j++)
        if (a[i][j] < 0) {
            flag = 1;
            break;
        }

    if (flag) {
        // вставка i-ой строки дублированием
        for (int i2 = n; i2 > i; i2--)
            for (j = 0; j < m; j++)
                a[i2][j] = a[i2 - 1][j];
        n++;
    }
}
```



# Делаем игру на основе 2D массива



# Кодируем состояние игры в 2D массиве

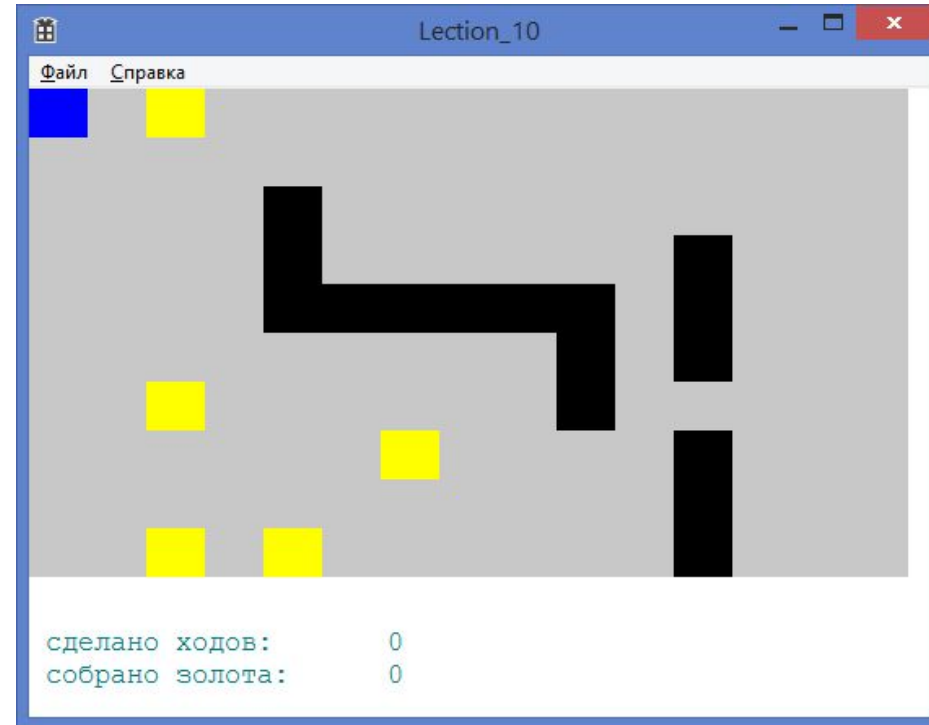
```
#define N 10
```

```
#define M 15
```

```
int a[N][M] = {  
{ 3, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  
{ 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  
{ 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0 },  
{ 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 0, 2, 0, 0, 0 },  
  
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 0, 0, 0 },  
{ 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0 },  
{ 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 2, 0, 0, 0 },  
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0 },  
{ 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0 }  
};
```

```
int steps = 0;
```

```
int gold = 0;
```



**Коды ячеек**

// 0 - ???

// 1 - ???

// 2 - ???

// 3 - ???

# Кодируем состояние игры в 2D массиве

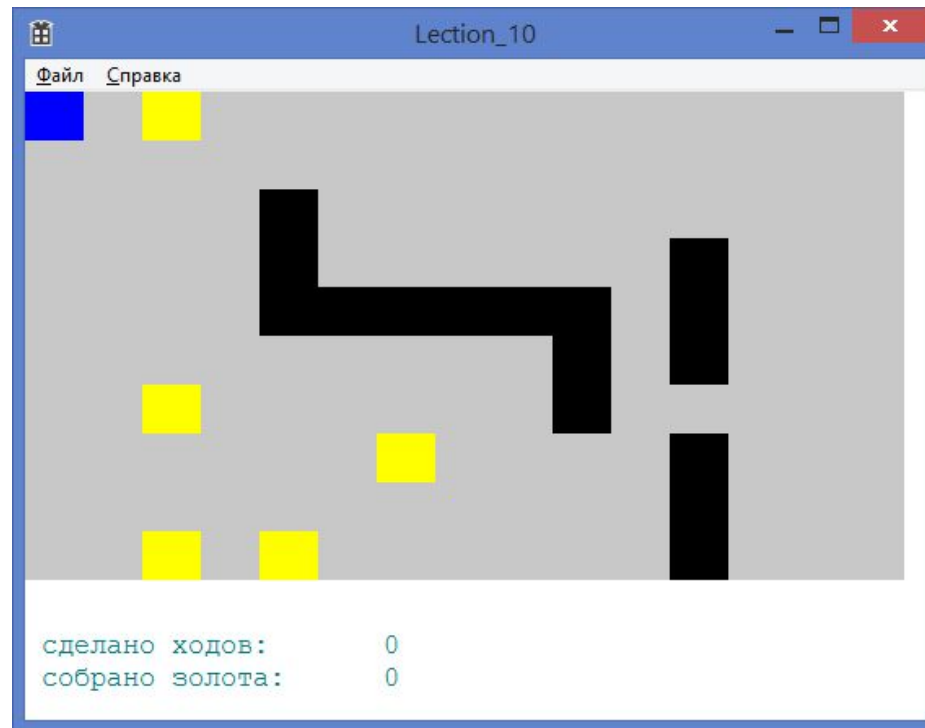
```
#define N 10
```

```
#define M 15
```

```
int a[N][M] = {  
{ 3, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  
{ 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  
{ 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0 },  
{ 0, 0, 0, 0, 2, 2, 2, 2, 2, 0, 2, 0, 0, 0, 0 },  
  
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 0, 0, 0 },  
{ 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0 },  
{ 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 2, 0, 0, 0 },  
{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0 },  
{ 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0 }  
};
```

```
int steps = 0;
```

```
int gold = 0;
```



## Коды ячеек

// 0 - свободно

// 1 - золото

// 2 - стена

// 3 - игрок



# Код функции WndProc

```
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;
```

# Код функции WndProc

```
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;  
    }  
    break;
```

# Изменение состояния игры: двигаем игрока

## ВЛЕВО

```
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++;
    }
}
```

# Изменение состояния игры: двигаем игрока вправо

```
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++;
    }
}
```

# Изменение состояния игры: двигаем игрока вверх

```
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++;
    }
}
```

# Изменение состояния игры: двигаем игрока вниз

```
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--;
    }
}
```

# Отрисовка состояния игры

```
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);

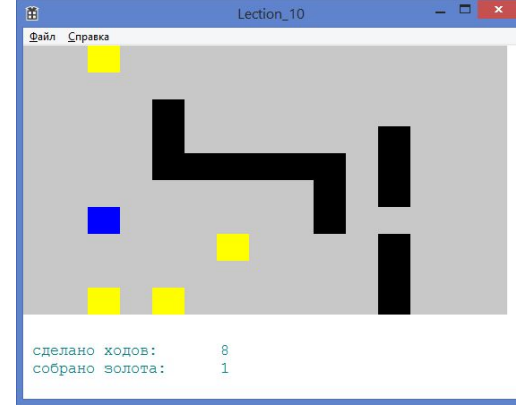
            EndPaint(hWnd, &ps);
        }
        break;
        ...
        return 0;
    }
}
```

# Отрисовка состояния игры (2)

```
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
```

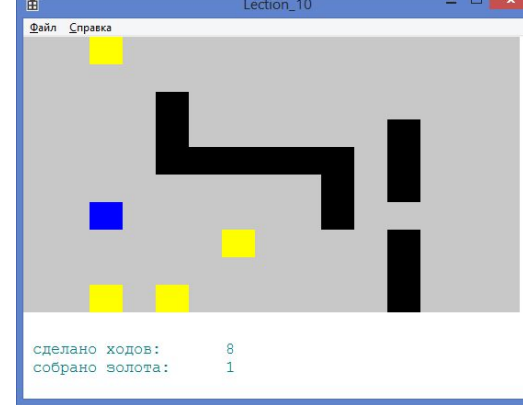
```
HBRUSH hBrushWall; //создаём кисть для стены  
hBrushWall = CreateSolidBrush(RGB(0, 0, 0)); // черный
```

```
HBRUSH hBrushMan; //создаём кисть для игрока  
hBrushMan = CreateSolidBrush(RGB(0, 0, 255)); // синий
```



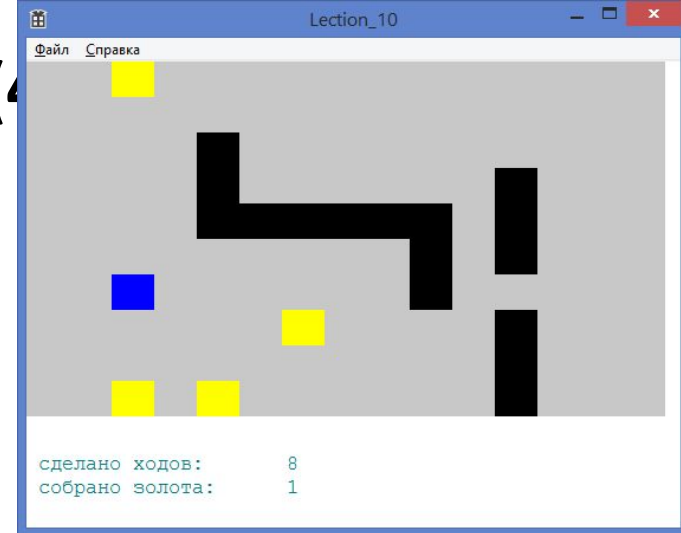
# Отрисовка состояния игры (3)

```
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++;
}
```



# Отрисовка состояния игры (4)

```
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));
```



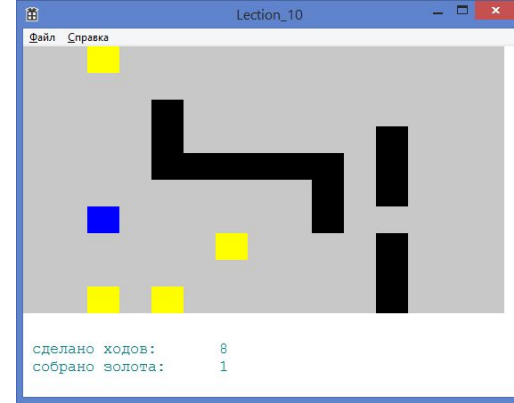
# Отрисовка состояния игры (5)

```
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);

} // конец функции void DrawField(HDC hdc)
```



# Домашнее задание

ЕСЛИ

хотите плюсы в карму

И

У вас есть лишнее время (т.е. нет долгов по другим предметам!!!)

ТО

Выберите себе игру из предложенных вариантов – и сделайте её!

# Источники информации

- msdn
- google