Федеральное агентство связи
Сибирский Государственный Университет
Телекоммуникаций и Информатики

Кафедра вычислительных систем

Курсовая работа
«Пятнашки»

Выполнил: Коробов А.П.
Группа: ЗП-202
Номер студенческого билета: 73200171
Адрес электронной почты:
sasha-korobov@list.ru
Проверил: _____

Новосибирск, 2021 г

Оглавление

Техническое задание:

Создать законченный и полностью рабочий программный продукт – игру «пятнашки». Приложение должно иметь интуитивно понятный интерфейс и покрытие тестами.

Описание проекта:

Игроку предоставляется поле, на котором сеткой 4x4 в случайном порядке расположены 15 чисел – от 1 до 15 (+1 пустое поле). Целью игры является упорядочить числа в порядке возрастания (в левой верхней ячейке минимальное число, в правой нижней пустая ячейка).

Для написания приложения использовался язык программирования C. Для тестирования использовалась библиотека Google Testing Framework.

Игровой процесс:

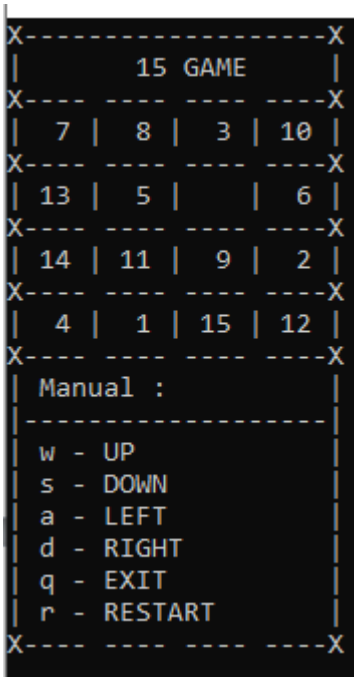
После запуска программы, генерируется игровое поле со случайно расположенными числами. Ячейки с числами можно передвигать на место пустой ячейки путем нажатия клавиш w, a, s, d. В результате числа должны расположиться в порядке возрастания.

Описание выполненного проекта:

Игра была написана интровертом-минималистом, поэтому в ней нет никаких таблиц лидеров, красивого дизайна и в принципе ничего лишнего. Весь геймплей происходит в консоле.

При запуске ярлыка 15.exe открывается консоль с приветствием:

После нажатия любой клавиши появляется игровое поле:



Описание используемых функций:

Объявление структуры игрового поля происходит при помощи typedef struct:

```
*/
typedef struct Game15Field
{
    int cells[FIELD_WIDTH][FIELD_HEIGHT];
} Game15Field;
```

где WIDTH – ширина, а HEIGT – высота поля. cells – массив игровых клеток.

Для перемешивания игрового поля используется функция shuffle_field. В случайном порядке игровое поле заполняется ячейками с числами.

```
void shuffle_field(Game15Field* field)
{
    for (int i = 0; i < FIELD_HEIGHT; ++i)
    {
        for (int j = 0; j < FIELD_WIDTH; ++j)
        {
            swap_values(&field->cells[i][j],
                &field->cells[rand() % FIELD_HEIGHT][rand() % FIELD_WIDTH]);
        }
    }
}
```

Вывод игрового поля в консоль осуществляет функция display_game_field:

```
void display_game_field(Game15Field field);
```

```
X------------------X
|       15 GAME    |
X---- ---- ---- ----X
|  7 |  8 |  3 | 10 |
X---- ---- ---- ----X
| 13 |  5 |    |  6 |
X---- ---- ---- ----X
| 14 | 11 |  9 |  2 |
X---- ---- ---- ----X
|  4 |  1 | 15 | 12 |
X---- ---- ---- ----X
```

Для проверки условия окончания игры используется is_end_game:

```
int is_end_game(Game15Field field)
{
    int counter = 1;

    if(field.cells[FIELD_HEIGHT-1][FIELD_WIDTH-1] != 0)
        return 0;

    for(int i=0;i<FIELD_HEIGHT;i++)
    {
        for(int j=0;j<FIELD_WIDTH;j++)
        {
            if(i != FIELD_HEIGHT - 1 || j != FIELD_WIDTH - 1)
            {
                if(counter++ != field.cells[i][j])
                    return 0;
            }
        }
    }

    return 1;
}
```

Если выводится 1, игра завершается.

Описание тестов:

Проверка перемешивания игрового поля:

```
TEST(ShuffleFieldTest, CheckShuffiling)
{
    Game15Field field;

    int inc = 0;
    for(int i=0;i<4;i++)
        for(int j=0;j<4;j++)
            field.cells[i][j] = inc++;

    shuffle_field(&field);

    inc = 0;

    bool isShuffeled = false;
    for(int i=0;i<4;i++)
    {
        for(int j=0;j<4;j++)
        {
            if(field.cells[i][j] != inc)
            {
                isShuffeled = true;
            }
            inc++;
        }
    }

    EXPECT_EQ(true, isShuffeled);
}
```

Проверка создания поля с уникальными клетками:

```
TEST(CreateFieldTest, CheckIfFieldCreatingSuccessufully)
{
    Game15Field field = create_field();

    bool passed = true;
    for(int k=0;k<16;k++)
    {
        bool isPresent = false;
        for(int i=0;i<4;i++)
        {
            for(int j=0;j<4;j++)
            {
                if(field.cells[i][j] == k)
                {
                    isPresent = true;
                }
            }
        }

        if(!isPresent)
        {
            passed = false;
            break;
        }
    }

    EXPECT_EQ(true, passed);
}
```

Проверка функции окончания игры:

```
TEST(EndTest, EndAccepted)
{
    int init[4][4] = {
        { 1, 2, 3, 4},
        { 5, 6, 7, 8},
        { 9,10,11,12},
        {13,14,15, 0}
    };

    int expected = true;

    Game15Field field;

    for(int i=0;i<4;i++)
    {
        for(int j=0;j<4;j++)
        {
            field.cells[i][j] = init[i][j];
        }
    }

    int acually = is_end_game(field);

    EXPECT_EQ(expected, acually);
}

TEST(EndTest, EndDeclined)
{
    int init[4][4] = {
        { 1, 2, 3, 4},
        { 5, 0, 7, 8},
        { 9,10,11,12},
        {13,14,15, 6}
    };

    int expected = 0;

    Game15Field field;

    for(int i=0;i<4;i++)
    {
        for(int j=0;j<4;j++)
        {
            field.cells[i][j] = init[i][j];
        }
    }

    int acually = is_end_game(field);
    EXPECT_EQ(expected, acually);
}
```

Листинг программы:

Так как листинг программы предоставлен в репозитории, а все основные функции были описаны выше, сам код программы в отчете отсутствует.