

K9

Федеральное государственное бюджетное образовательное учреждение высшего
образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра вычислительных систем

Курсовая работа

по теме «Пятнашки»

Выполнил:

студент гр. ИВ-222

Перегоедов К.И.

Проверил:

ст. Преподаватель

Токмашева Е.И.

.

Новосибирск, 2023

ВВЕДЕНИЕ.

Сегодня большое влияние на все сферы общественной жизни оказывают информатизация и информационные технологии. В частности все больше пользуются спросом программные приложения и продукты. Для их написания требуются определенные навыки работы в различных программах, предназначенных специально для создания приложений на каком-либо языке программирования.

Данная работа, посвященная написанию программного приложения «Пятнашки», поспособствует изучению одной из таких программ.

Для достижения этой цели необходимо решить ряд задач:

- Изучить аналоги игры «Пятнашки»;
- Определить функциональность будущего приложения;
- Изучить ранее неизвестные алгоритмы и структуры, необходимые для написания программы или разработать собственные;
- Проработать удобный пользовательский интерфейс.

Приведенные выше задачи будут более подробно рассмотрены в основном тексте курсовой работы.

ГЛАВА 1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ.

1.1 Правила игры.

Пятнашки — популярная головоломка, придуманная в 1878 году Ноем Чепменом. Представляет собой набор одинаковых квадратных костяшек с

нанесёнными числами, заключённых в квадратную коробку. Длина стороны коробки в четыре раза больше длины стороны костяшек для набора из 15 элементов (и в три раза больше для набора в 8 элементов), соответственно в коробке остаётся незаполненным одно квадратное поле.

Цель игры — перемещая костяшки по коробке добиться упорядочивания их по номерам, желательно сделав как можно меньше перемещений. Существуют многочисленные алгоритмы решения головоломки, гарантирующие получение результата за наименьшее количество ходов.

1.2 Аналогии.

Большинство найденных мной приложений представляли собой простую реализацию данной игры. Многие во время исполнения выдавали различного рода ошибки, связанные с исчезновением цифр или перемещением полей, содержащих цифры.

Встречались и довольно специфичные варианты, предлагавшие игроку собрать рисунок из фрагментов изображения, наложенных на костяшки.

Также следует упомянуть упрощенные варианты с меньшим количеством полей (3 на 3) и усложненные (4 на 4 и более).

В общем виде данное табло можно представить в виде таблицы 1:

Таблица 1 – Образец табло.

5	7	3	8
15	1	13	2
14	10	6	4
9	11	12	

- Игрок должен перемещать по одной клетки с цифрой на пустое место. - Так происходит до тех пор, пока пользователь не выстроит последовательную комбинацию цифр (Таблица 2), и лишь после этого игрок считается победителем.

Таблица 2 – Правильное заполнение табло.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

ГЛАВА 2 ПОСТАНОВКА ЗАДАЧИ.

Целью данной курсовой работы является развитие навыков программирования, путем написания игры «Пятнашки», для реализации которой нужно выполнить ряд задач, перечисленных ранее.

Основываясь на правилах игры и изученных аналогов можно выделить следующие функциональные требования:

- Реализация алгоритма таким образом, чтобы начальное расположение чисел было случайным с наименьшей вероятностью повторения.
- Возможности прервать игру, для этого нужно закрыть текущую и начать новую.
- Удобный для пользователя интерфейс.
- Работа с графическими возможностями .

В итоге получилась программа – игра, в которой использовались основные возможности программы С.

ГЛАВА 3 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Игра «Пятнашки», представленная в данной работе написана на языке С, с помощью интегрированной среды разработки Embarcadero Dev-C.

3.1 Описание приложения.

Разработанное программное приложение является стандартной, оригинальной игрой «Пятнашки», не использующей элементы графических изображений или рознящиеся с подлинными размерами игрового поля.

Любой не ознакомленный с игрой человек, сможет интуитивно разобраться в принципе работы приложения.

4. Используемые методы

Программа выполнена в среде программирования С++. Сама программа - приложение Windows. Программой обрабатываются события от нажатия клавиш на клавиатуре.

Для того чтобы не усложнять листинг программы, в ней имеется основная процедура.

- Основная процедура, т.е. та процедура, которая отвечает за саму игру и взаимодействие с пользователем во время игры;

- Основная программа.

К основным процедурам относятся:

- Табло;

- Вывод;

- Направление;

- Поиск;
- Замена;
- Проверка;
- Игра15

В данном разделе опишем основные приемы используемых процедур.

4.1 Основные процедуры.

4.1.1 процедура Табло ;

Данная процедура формирует табло, заполненное случайными, неповторяющимися цифрами от 1 до 15 и одной пустой клеткой. Процедура реализована с использованием двух массивов: одномерный и двумерный.

Одномерный массив с 16 целыми числами заполняется случайным образом, причем, учитывается, чтобы цифры в данном массиве не повторялись.

Каждой цифре из одномерного массива присваивается, аналогичный ей строковый элемент и вводится в двумерный массив. К примеру, цифре 5, присваивается строковый элемент '5 '. Исключение составляет цифра 16, ей присваивается символ "***".

4.1.2 процедура Вывод;

Процедура вывода на экран табло с цифрами сформированное на момент отображения. Если программа только запущенна, тогда на экран выводится таблица заполненное случайным образом. Если игра уже идет то данная процедура выводит на экран ту комбинацию цифр, которая определена пользователем во время игры.

4.1.3 процедура Направление ;

В данной процедуре пользователю, при помощи клавиш-стрелок, предлагается ввести направление перехода. В данной процедуре считывается код нажатой клавиши, чтоб в дальнейшем можно было осуществлять передвижение.

4.1.4 процедура Поиск ;

В этой процедуре осуществляется поиск пустого элемента. Это необходимо для того, чтоб в дальнейшем пользователь смог относительно пустого элемента

сделать свой ход. Процедура считывает каждый элемент двумерного массива и сравнивает его с пустым. После того как пустой найден процедура запоминает координаты пустого элемента, а именно строку и столбец.

4.1.5 процедура Замена ;

Программа в зависимости от выбора направления осуществляет перестановку элементов в двумерном массиве.

4.1.6 процедура Проверка;

После совершения перестановки цифр, а иными словами после очередного хода, программа сравнивает расстановку цифр в двумерной матрицы. Если текущая комбинация является правильной, тогда игрок считается победителем.

4.1.7 процедура Игра 15;

Эта процедура является основной. В ней подключается графический модуль и происходит основной процесс игры.

Игра будет продолжаться до тех пор, пока не будет разложен правильный расклад.

5 Интерфейс.

При запуске генерируется игровое поле 4 на 4.

6. Реализация.

Главной задачей было создать работающую игру в удобной пользователю оболочке. Под простым на вид интерфейсом скрыт алгоритм, создающий поле с цифрами и отслеживающий их положение.

Сначала проводятся операции с двумерными массивами, так как, по сути, мы имеем матрицу значений от 1 до 16. При запуске приложения заполняется массив объектов.

Следующий фрагмент программы перемешивает числа, кроме пустой, создавая случайную последовательность чисел, которую пользователь может увидеть сразу после начала игры.

При нажатии определённой клавиши-стрелки, процедура проверяет, является ли нажатая кнопка той, при помощи которой можно переместить, путем проверки её положения в массиве относительно пустой ячейки. Далее, если условие выполнено, число и пустая ячейка меняются местами.

В программе имеется логическая переменная, которая каждый раз при совершении хода делает проверку последовательности чисел. Если пользователь успешно расставил фишки в возрастающей последовательности, он получает сообщение об успехе. ЗАКЛЮЧЕНИЕ

Для запуска игры достаточно любого компьютера, на котором установлена хоть какая-нибудь операционная система, семейство Windows. Было создано программное приложение «Пятнашки», в ходе разработки которого были изучены новые алгоритмы и структуры языка C, что и являлось основной целью данной курсовой работы.

Список литературы

Открытые Интернет источники.

Совершенный алгоритм. Графовые алгоритмы и структуры данных. Автор: Рафгарден Тим.

Алгоритмы. Руководство по разработке. Автор: Стивен С. Скиена.

Приложение 1: Листинг программы

```
// игра "Пятнашки"
```

```
// управление стрелками
```



```
#include <iostream>
```

```
#include <vector>
```

```
#include <cstdlib>
```

```
#include <iomanip>
```

```
#include <conio.h>
```

```
using std::vector; using
```

```
std::cout;
```

```
const unsigned short SIZE = 4; // размер игрового поля
```

```
vector<int> in_game_map(SIZE); vector<vector<int>>
```

```
game_map(SIZE, in_game_map); // игровая карта
```

```
vector<int> in_right_map(SIZE); vector<vector<int>> right_map(SIZE,
```

```
in_right_map); // правильная итоговая карта
```

```
struct coordinate // хранилище координат нулевого элемента
```

```
{ unsigned
```

```
x; unsigned
```

```
y;
```

```
} zero; // объект
```

```
void create_right_map() // создаем правильную карту заполненную по порядку
```

```
{ unsigned right_value = 1; for
```

```
(unsigned i = 0; i < SIZE; i++)
```

```

    {      for (unsigned j = 0; j < SIZE;
j++)      right_map[i][j] =
right_value++; }    right_map[SIZE-
1][SIZE-1] = 0; // нулевой элемент в
нижний правый угол }

```

void create_game_map() // рандомно создаем игровую карту

```

{    unsigned limit =
SIZE*SIZE;

```

vector<int> temporary; // временный массив из которого будем брать значения в игровую карту

```

    for (unsigned i = 0; i < limit; i++)
temporary.push_back(i);

```

```

    int value;    for (unsigned i = 0; i
< SIZE; i++)

```

```

    {      for (unsigned j = 0; j < SIZE;
j++)
    {

```

```

        value = rand() % limit--;
game_map[i][j] = temporary[value];

```

```

        if (temporary[value] == 0) // сохраняем координаты нулевого элемента
        {
zero.x = j;
zero.y = i;
        }

```

```

        temporary.erase(temporary.begin() + value);

```

```
    }  
    }  
}
```

bool check_map() // сравнение игровой и правильной карты для определения
конца игры

```
{    if (game_map ==  
right_map)  
    return true;  
return false;  
}
```

void up_move() // ход вверх (нулевой элемент вниз)

```
{    if (zero.y >  
0)  
    {  
        game_map[zero.y][zero.x] = game_map[zero.y - 1][zero.x];  
zero.y--;    game_map[zero.y][zero.x] = 0;  
    }  
}
```

void down_move() // ход вниз (нулевой элемент вверх)

```
{    if (zero.y < SIZE -  
1)  
    {  
        game_map[zero.y][zero.x] = game_map[zero.y + 1][zero.x];  
zero.y++;    game_map[zero.y][zero.x] = 0;  
    }  
}
```

```
    }  
}
```

```
void right_move() // ход вправо (нулевой элемент влево)  
{    if (zero.x < SIZE -  
1)  
{  
    game_map[zero.y][zero.x] = game_map[zero.y][zero.x + 1];  
zero.x++;    game_map[zero.y][zero.x] = 0;  
}  
}
```

```
void left_move() // ход влево (нулевой элемент вправо)  
{    if (zero.x >  
0)  
{  
    game_map[zero.y][zero.x] = game_map[zero.y][zero.x - 1];  
zero.x--;    game_map[zero.y][zero.x] = 0;  
}  
}
```

```
void get_direction() // определяем нажатую игроком стрелку  
{  
    int move = static_cast<int> (_getch()); // UP = 72, DOWN = 80, RIGHT = 77,  
LEFT = 75  
    switch (move)
```

```

    {
case 72:
    {
        up_move(); break;
    }
case 80:
    {
        down_move(); break;
    }
case 77:
    {
        right_move(); break;
    }
case 75:
    {
        left_move();
break;
    }
    default:
    {
get_direction();
    }
    }
}

void screen() // выводим массив на экран
{
system("cls");
for (unsigned i

```

```

= 0; i < SIZE;
i++)

    {        for (unsigned j = 0; j < SIZE;
j++)
        {
            if (game_map[i][j] != 0)        cout << std::setw(2) <<
std::setfill('0') << game_map[i][j] << ' ';        else
                cout << "*** "; // нулевой элемент
        }        cout
<< '\n';
    }
}

```

```

int main() {
    srand(static_cast<int>(time(NULL)));

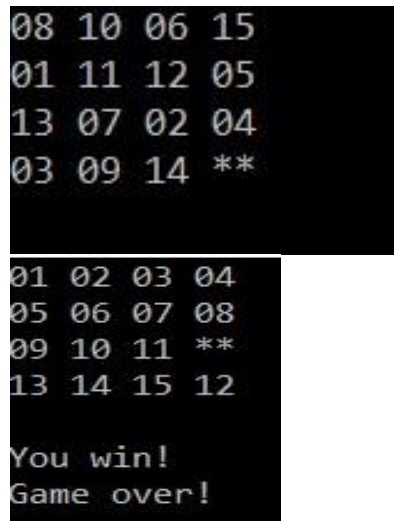
    create_right_map(); // создание игровых карт
do
    {
        create_game_map();
    } while (check_map());

do // игровой цикл
    {
        screen();
get_direction(); }
while
(!check_map());

```

```
cout << "\nYou win!\nGame over!\n";  
_getch();  
}
```

Приложение 2: Результаты работы



```
08 10 06 15  
01 11 12 05  
13 07 02 04  
03 09 14 **  
  
01 02 03 04  
05 06 07 08  
09 10 11 **  
13 14 15 12  
  
You win!  
Game over!
```

Репозиторий:

https://github.com/trpo2023/cw-iv-222_15-puzzle