**Vinnytsia IT-Academy**

«Jumping Snake

video game»

Realized by Stepayko Sergey

group A16-green,

Vinnitsya IT-Academy

Vinnytsia 2016

CONTENTS

1. Presentation………………………………………………….……………3
2. System requirements………………………………………….……….….5
3. User guide…………………………………………………….…………...5
4. Programmer instruction……………………………………………….….6
5. About author…………………………………………..………….…........8
6. **Presentation**

I am proud to present you my project.

This video game is based on classic snake game from a year 1977.

But my reincarnation of that game has much significant features, such as:

- the snake can move in any angle (not only Up, Down, Left and Right), because the direction is a float type variable that contains angle in radiants.

- screen borders are not walls;

- and this snake can jump over the walls, food or bonuses on map;

So, my game is named - Jumping Snake.

The game could be played with one or two players.

To win the game, a player should collect all food and bonuses on the map and don`t hit the walls, his own body or another player’s snake.

Each player has three control buttons, Left, Right and Jump.

The game code was written in such a way, that a user can set almost any display resolution, from 100\*100 and up to Full HD, 4K or 8K and game will work in it, pixel in pixel. No blur or shape distortion even in not standard resolutions and screen aspect ratios.

All ingame graphics are vector type, every element looks clear and sharp so in any game resolution.

The size of the game elements also could be changed in a wide range from ten up to fifty pixels.

All game events are sounded. Game and menu have different background music.

The menu could be operated with the keyboard keys or using a mouse.

As a prize, a game winner (or winners) can see actual replay of his (their) game.

In the end of the game it is possible to replay this level again or randomly generate another level.

Ingame menu have such settings that could be changed:

* a screen mode (full screen or window);
* a window width and height;
* a game difficulty (speed);
* a unit size;
* a background color.

Also there are “help” and “credits” screens.

1. **System requirements**

Operating system: Windows 7 or higher;

Processor: 1 GHz of faster;

Video adapter RAM: 32 Mb or more;

RAM: 256 Mb or more;

Free HDD space: 7.5 Mb or more.

**3. User guide**

You should start the game executing JumpingSnake.exe file.

After, using arrows keys on keyboard or a mouse - choose HELP menu item and select it with ENTER or left mouse button.

Read future instructions and explanations on screen.

**4. Programmer instruction**

Used programming language is C++.

C++ is a general-purpose [programming language](https://en.wikipedia.org/wiki/Programming_language). It has [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) and [generic](https://en.wikipedia.org/wiki/Generic_programming) programming features, while also providing facilities for [low-level](https://en.wikipedia.org/wiki/Low-level_programming) [memory](https://en.wikipedia.org/wiki/Memory_(computing)) manipulation.

Used [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) is Microsoft Visual Studio 2010.

Microsoft Visual Studio is an IDE from [Microsoft](https://en.wikipedia.org/wiki/Microsoft). It is used to develop [computer programs](https://en.wikipedia.org/wiki/Computer_program) for [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), as well as [web sites](https://en.wikipedia.org/wiki/Web_site), [web applications](https://en.wikipedia.org/wiki/Web_application) and [web services](https://en.wikipedia.org/wiki/Web_service). Visual Studio uses Microsoft software development platforms such as [Windows API](https://en.wikipedia.org/wiki/Windows_API), [Windows Forms](https://en.wikipedia.org/wiki/Windows_Forms), [Windows Presentation Foundation](https://en.wikipedia.org/wiki/Windows_Presentation_Foundation), [Windows Store](https://en.wikipedia.org/wiki/Windows_Store) and [Microsoft Silverlight](https://en.wikipedia.org/wiki/Microsoft_Silverlight). It can produce both [native code](https://en.wikipedia.org/wiki/Native_code) and [managed code](https://en.wikipedia.org/wiki/Managed_code).

Used Additional library is Allegro v.5.0.2.

This is a cross-platform library mainly aimed at video game and multimedia programming. It handles common, low-level tasks such as creating windows, accepting user input, loading data, drawing images, playing sounds, etc. and generally abstracting away the underlying platform. However, Allegro is not a game engine, programmer is free to design and structure his program as he like.

Used notation – Hungarian.

Hungarian notation is an [identifier naming convention](https://en.wikipedia.org/wiki/Identifier_naming_convention) in [computer programming](https://en.wikipedia.org/wiki/Computer_programming), in which the name of a [variable](https://en.wikipedia.org/wiki/Variable_(computer_science)) or [function](https://en.wikipedia.org/wiki/Function_(computer_science)) indicates its [type](https://en.wikipedia.org/wiki/Data_type) or intended use.

Hungarian notation was designed to be language-independent, and found its first major use with the [BCPL](https://en.wikipedia.org/wiki/BCPL) programming language. Because BCPL has no data types other than the machine word, nothing in the language itself helps a programmer remember variables' types. Hungarian notation aims to remedy this by providing the programmer with explicit knowledge of each variable's data type.

The project has almost two thousands lines of code placed in 6 files, three header files and three .cpp files.

Main.h and Main.cpp files contain main program code.

Menu.h and Menu.cpp files contain a class, an extensible program-code-template for creating [objects](https://en.wikipedia.org/wiki/Object_(object-oriented_programming)) (menus), providing initial values for state ([member variables](https://en.wikipedia.org/wiki/Member_variable)) and implementations of behavior (member functions or [methods](https://en.wikipedia.org/wiki/Method_(computer_programming))), such as:

* initMenu;
* initMenuBlock;
* changeSelectedItem;
* draw;
* checkInput;
* getSelectedItem;
* setDataInputMode;
* getInputedData;
* setMenuItemName.

Csnake.h and Csnake.cpp files contain a class, for work with snakes objects.

It contains such [methods](https://en.wikipedia.org/wiki/Method_(computer_programming)):

* initSnake;
* moveSnake;
* directionChange;
* drawSnake;
* increase;
* setSnakeSpeed;
* getSnakeBodySize;
* reduse;
* getSnakeHeadStruct;
* getSecondSegmentStruct;
* getLastSegmentStruct;
* SpeedUpBonusPicked;
* setJump;
* rotate\_On;
* rotate\_Off;

All significant code blocks are commented.

**5. About author**

This project was realized by Stepayko Sergey, student of Vinnytsia IT-Academy, in October 2016.