**Sudoku概要设计**

1. **文档概述**

此概要说明对Sudoku软件的功能做了概要性设计的说明，为软件详细设计奠定了基础。

1. **模块详细概述**
   1. **Sudoku类模块**

需求分析中大概设计了两个类，后来觉得设置一个类应该就可以了，3×3矩阵的判断可以转化成一个二维数组。

Sudoku类中包含四个字段及其相应属性分别是：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 属性 | 类型 | 含义 |
| shudu | Shudu | Char [9,9]数组 | 数独矩阵 |
| hang | Hang | Bool [9,9]数组 | 行判定，每一行代表数独每一行，从上至下依次标号，每一列代表数独该行该数是否出现 |
| lie | Lie | Bool [9,9]数组 | 列判定，每一行代表数独每一列，从左至右依次标号，每一列代表数独该列该数是否出现 |
| sansan | Sansan | Bool [9,9]数组 | 行判定，每一行代表数独每一个3\*3小矩阵，从左至右从上至下依次标号，每一列代表数独该小矩阵该数是否出现 |

包含两个函数分别是：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数名 | GenerateSudokuEnding | | | |
| 文件名 | Mysudoku/Mysudoku/Sudoku.cs | | | |
| 功能 | 生成数独终局 | | | |
| 记述形式 | public void GenerateSudokuEnding() | | | |
| 参数 无 | | | | |
| 返回值 | | 类型 | 无 | 说明 |
|  | | 值 | 无 | 无 |
| 详细说明 | | | | |
| 通过一定的算法生成数独终局。 | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数名 | SolveSudoku | | | |
| 文件名 | Mysudoku/Mysudoku/Sudoku.cs | | | |
| 功能 | 求解数独 | | | |
| 记述形式 | public void SolveSudoku( ) | | | |
| 参数 无 | | | | |
| 返回值 | | 类型 | 无 | 说明 |
|  | | 值 | 无 | 无 |
| 详细说明 | | | | |
| 通过一定的算法求解导入的数独谜题。 | | | | |

两个函数之间没有直接关联，是两种关于数独的功能，不过内部会调用相关的字段，比如判断改行是否有重复数字的时候会查看hang，lie，sansan数组，同时会将数字写入shudu数组。

* 1. **控制模块（输入输出模块）**

软件主操作界面为控制台，所有的参数输入都通过控制台传输给主函数，由主函数来判断具体的命令操作（同时也会判断命令是否正确），根据不同命令参数去执行不同板块：

1. 在控制台输入指令“Sudoku.exe -c *有效数字*”可以输出改有效数字数量的数独终局到txt文件中。
2. 输入“Sudoku.exe -s *绝对路径*”可以读取绝对路径中txt文本中的数独，求解并输出结果

当判断为命令1时，会调用Sudoku类的GenerateSudokuEnding函数来生成数独；当命令为2时，会调用Sudoku类的SolveSudoku函数来求解，否则，判定为无效命令，结束程序。

同时设置了一个输出函数OutputToTxt，统一将生成的数独输出到TXT文件中。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 函数名 | OutputToTxt | | | | |
| 文件名 | Mysudoku/Mysudoku/Program.cs | | | | |
| 功能 | 输出数独写入txt文件 | | | | |
| 记述形式 | static void OutputToTxt(Sudoku sudoku,StreamWriter streamWriter) | | | | |
| 参数 | | | | | |
| 类型 | | 变量名称 | | I/O | 说明 |
| Sudoku | | Sudoku | | I | 要输出的数独 |
| StreamWriter | | StreamWriter | | I | 写入流 |
| 返回值 | | 类型 | 无 | 说明 | |
|  | | 值 | 无 | 无 | |
| 详细说明 | | | | | |
| 将生成的数独写入指定的txt文件中。 | | | | | |

1. **数据结构定义**

class Sudoku

{

public void GenerateSudokuEnding()

{

//填充部分

}

public void SovleSudoku()

{

//填充部分

}

private char[,] shudu = new char[9, 9];

public char[,] Shudu

{

get { return shudu; }

set { shudu = value; }

}

private bool[,] hang = new bool[9, 9];

public bool[,] Hang

{

get { return hang; }

set { hang = value; }

}

private bool[,] lie = new bool[9, 9];

public bool[,] Lie

{

get { return lie; }

set { lie = value; }

}

private bool[,] sansan = new bool[9, 9];

public bool[,] Sansan

{

get { return sansan; }

set { sansan = value; }

}

}

}

class Program

{

static void Main(string[] args)

{

//填充部分

}

static void OutputToTxt(Sudoku sudoku,StreamWriter streamWriter)

{

//填充部分

}