# 数独程序实现

## 1需求分析

需求分析

1.生成终局 格式：sudoku.exe -c n

1）不重复

2）1<=n<=1000000，能处理非法参数

4）n在1000以内时，要求程序在 60 s 内给出结果

5）输出生成的数独终盘至sudoku.txt

2. 求解数独 格式：sudoku.exe -s path

1）从指定文件读取，每读81个数字视作一道数独题，忽略其他字符

2）要求文件内的数独题目是合法的

3）文件内数独个数在1000以内时，要求程序在 60 s 内给出结果

4）输出已读入数独的答案至sudoku.txt。若存在未满81个的数字，在已解出的答案后输出“存在错误格式!”

## 2数独游戏的基本解决方法

数独横九竖九共八十一个格子，同时又分为9个九宫格。规则很简单——需要每一个格中的数字，都保证与其所在横排和竖排以及九宫格内无相同数字。

所以我们的大概思路就是，从第一个空格开始试着填数，从 1 开始填，如果 1 不满足横排竖排九宫格无重复的话，就再填入 2 ，以此类推，直到填入一个暂时满足规则的数，中断此格，移动到下一个空格重复这个过程。

如果到达某个空格发现已经无数可选了，说明前面某一格填错了，那就返回上一格，从上一格的中断处继续往 9 尝试，直到这样回朔到填错的那一格。

这样的话，我们就可以整理出重要的步骤了：

* 寻找到下一个空格
* 轮流填入格中数字 1 到 9
* 递归判断填入数是否符合规则

## 3设计实现

* 输入处理类：

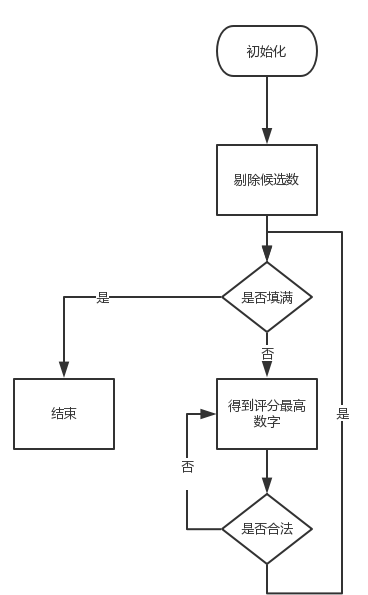
根据参数调用下列函数进行相应处理（包括参数合法性判断）

* 终盘生成类：

种子生成函数、交换组合函数、行列交换函数、转换输出函数

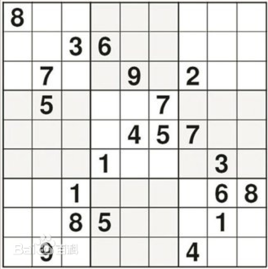
* 数独求解类：

初始化函数、记录函数函数、检错函数、恢复函数、深度优先遍历函数、评价函数、减少候选数函数。



## 4.程序

首先测试数独使用的是芬兰数学家因卡拉花费3个月时间设计出的世界上迄今难度最大的数独。如下



将空格用 0 表示，同时将数独表示成嵌套的列表，这样每格的行数和列数就正好是列表中每个对应数的索引。

程序如下：

from createFun import \*

from IsSDFun import \*

import wx

import random,math

createSD() #创建数独矩阵 return matrix

print("完整矩阵")

print\_grid(matrix)

#matrix1用于储存残缺矩阵,赋初值0; matrix2用于作改变的中间矩阵

#matrix2和matrix1设置一样的初值，但是直接matrix2=matrix1，因为这样赋值是指针赋值，更改一个时另一个也变化

matrix1=[]

matrix2=[]

for i in range(9):

matrix1.append([0] \* 9)

matrix2.append([0] \* 9)

n=70 #设置显示个数

t=0

while t<n:

i=random.randint(0,8)

j=random.randint(0,8)

if matrix1[i][j]==0:

matrix1[i][j]=matrix[i][j] #从matrix中给matrix1对应位置赋值

matrix2[i][j]=matrix[i][j]

t+=1

print("残缺矩阵")

print\_grid(matrix1)

#生成一个字典保存文本框对象，以（0到80）为键，81个文本框对象为值，

#用于残缺矩阵和窗口文本框之间传值

dic={}

class Frame1(wx.Frame):

def \_\_init\_\_(self,superior):

wx.Frame.\_\_init\_\_(self,parent=superior,title="数独",size=(800,600))

panel=wx.Panel(self)

for i in range(9): #循环显示出matrix1,并给字典赋值

for j in range(9):

dic[i\*9+j]=wx.TextCtrl(panel,value=str(matrix1[i][j]),pos=(55\*j,55\*i),size=(50,50),style=wx.TE\_CENTER)

#dic[0].SetValue("10")

#print(dic[0].GetValue())

#print(dic)

self.btnCS=wx.Button(parent=panel,label=u"测 试",pos=(600,100),size=(100,30))

self.btnCL=wx.Button(parent=panel,label=u"重 来",pos=(600,150),size=(100,30))

self.TSText=wx.TextCtrl(parent=panel,pos=(550,250),size=(200,200))

#按钮点击事件绑定函数

self.Bind(wx.EVT\_BUTTON,self.CS,self.btnCS)

self.Bind(wx.EVT\_BUTTON,self.CL,self.btnCL)

def CS(self,event):

for i in range(9): #循环用把文本框的内容赋值给matrix2

for j in range(9):

matrix2[i][j]=eval(dic[i\*9+j].GetValue())

flag,ts=isSD(matrix2)

self.TSText.SetValue(ts)

def CL(self,event):

for i in range(9): #循环显示出matrix1，并给字典赋值

for j in range(9):

num=matrix1[i][j]

dic[i\*9+j].SetValue(str(num))

if \_\_name\_\_=="\_\_main\_\_":

app=wx.App()

frame=Frame1(None)

frame.Show()

app.MainLoop()