数独をCプログラムで解く

# 数独のルール

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 0 |  |  |  |  |  |  |  |  |  | Low |
| 1 |  |  |  |  |  |  |  |  |  | Middle |
| 2 |  |  |  |  |  |  |  |  |  | High |
| 3 |  |  |  |  |  |  |  |  |  | Low |
| 4 |  |  |  |  |  |  |  |  |  | Middle |
| 5 |  |  |  |  |  |  |  |  |  | High |
| 6 |  |  |  |  |  |  |  |  |  | Low |
| 7 |  |  |  |  |  |  |  |  |  | Middle |
| 8 |  |  |  |  |  |  |  |  |  | High |
|  | L | M | H | L | M | H | L | M | H | L:Low, M:Middle, H:High |

図 1　数独の表

* 数独の表は9x9の行列である。グリッドと呼ぶ
* 行と列の数字でマスの場所を特定する。
  + 左上隅のマスは(0,0)、右下隅のマスは(8,8)など
* 3x3の行列が9つある。9x9の行列に隙間やずれなく敷き詰められている。
* 各マスには1-9までの数字が1つ入る。
* 各行の各マスには1-9までの数字が必ず1つ入り、重複や抜けはない。
* 各列の各マスには1-9までの数字が必ず1つ入り、重複や抜けはない。
* 各3x3の行列の各マスには1-9までの数字が必ず1つ入り、重複や抜けはない。
* マス目にあらかじめ数字が入っている場合がある。

## 英語表記

|  |  |  |
| --- | --- | --- |
| 9x9行列：grid | 確定した：fixed |  |
| 1-9の数字：digit | 候補：candidate |  |
| マス1つ：square | 有効：valid |  |
| 3x3行列：box | 未確定の：unfixed |  |
| 行：row | 仮の：temporary |  |
| 列：column | 共通の：common |  |

# データ構造

* グリッドのデータ構造
  + マスの確定数字 (square\_fixed\_digits)
  + マス無効数字 (square\_invalid\_digits)
  + マス候補 (square\_candidate\_digits)
  + 行の確定数字 (row\_fixed\_digits)
  + 行候補 (row\_candidate\_digits)
  + 列の確定数字 (column\_fixed\_digits)
  + 列候補 (column\_candidate\_digits)
  + 3x3行列の確定数字 (box\_fixed\_digits)
  + 3x3行列候補 (box\_candidate\_digits)
  + 候補が存在しないマスの有無 (is\_no\_candidate\_square)
  + 未確定のマスの数 (num\_blank\_squares)
  + 仮確定したマス (square\_fixed\_temporary)
  + 仮確定した数字 (fixed\_temporary\_digit)

# 解法

## マスを埋める処理 (FillSquares)

1. すべてのマスのマス候補を求める
   1. 数値候補が1つもないマスが出た -> 処理を抜ける
2. あるマスの候補がマス確定する条件を満たしたら、その数字を確定数字とする。
   1. 確定マスの数 + 1, 未確定のマスの数 - 1
3. 確定したマスの数が0
   1. 未確定のマスの数が0 -> 完成・終了
   2. 数値候補が1つに定まるマスが一つもない -> 仮確定処理 -> マスを埋める処理を実行（再帰処理）
   3. 仮確定取り消し処理
4. 1に戻る

## 各マスの候補を求める処理　(CalculateCandidates)

マス(0,0)の候補を求めるには行と列と3x3行列それぞれで求めた候補をすべて考慮する必要がある。

* 行候補の計算 (CalculateRowCandidates)
  + 0行目に含まれない数字 (GetUnfixedDigitsInRow)
* 列候補の計算 (CalculateColumnCandidates)
  + 0列目に含まれない数字 (GetUnfixedDigitsInColumn)
* 3x3行列候補の計算 (CalculateBoxCandidates)
  + 3x3行列に含まれない数字 (GetUnfixedDigitsInBox)
* 無効数字invalid\_digits)の計算
  + 無効数字に含まれない数字
* マスに共通する候補の計算 (GetCommonCandidates)
  + 行候補、列候補、3x3行列候補に共通する数字
* マス候補 (CalculateSquareCandidates)
  + マスに共通する候補のうちマス無効数字に該当しない数字

## あるマスの数値を確定する処理 (FixSquareDigit)

マス(0,0)の候補が確定数字になる条件

1. マス候補が1つの場合、そのマス候補が確定数字になる。
2. マス候補が2つ以上ある場合は、そのマスを含む

* 同じ行のすべてのマスのマス候補 (GetUniqueRowCandidate)、または
* 同じ列のすべてのマスのマス候補(GetUniqueColumnCandidate)、または
* 同じ3x3行列のすべてのマスのマス候補 (GetUniqueBoxCandidate)

の中で、唯一となるマス候補が存在すればそれが確定数字になる。

## 数値候補が1つに定まるマスが一つもない場合の処理

1. 仮確定処理を実行する。
   * マス候補の数字の中から1つを確定させる。
2. マスを埋める処理を実行する。
   * 数値候補をすべて求めた上で、数値候補が1つに定まるマスが一つもない場合の処理を実行する。
   * 言いかえれば、マスを埋める処理を最後まで実行し、埋まっていないマスがあればそのマスはすべて数値候補が1つに定まらないマスである。

### 仮確定処理 (FixTemporaryDigit)

1. 確定していないマス (i, j)のうち、iが最も小さいマスの中でjが最も小さいマスを選ぶ (GetUnfixedSquare)
2. 選択したマスのマス候補のうち最も小さい値を確定数字にする (ChooseTemporaryDigitFromSquareCandidates)

### 仮確定取り消し処理 (CancelTemporaryDigit)

1. 確定数字を削除する (RemoveFixedDigit)
2. 確定数字をマス候補有効数字から除外する (RemoveSquareValidDigit)

## 数値候補が1つもないマスが存在する場合の処理

数値候補が1つもないマスが存在する場合、その状態では解けないことになる。

1. 各マスの数値候補を求める中で、数値候補の数が0になる。
2. 数値候補が0となるマスがなくなる直前のマスの値確定を戻し、その仮確定の値を数値候補から削除する。
3. 仮確定する数字を別の数字に変える。もし、仮確定する数字が存在しなければ、2に戻る。仮確定したマスが存在しなければ、数独が解けないことが決まる。
4. マスを埋める処理を実行する。

# プログラムフロー

## 数独を解くプログラム

* 開始
* 数独の問題をファイルから読み込む
* マスを埋める処理 (FillSquares)
* 結果の表示
  + 成功：解けた旨表示
  + 失敗：解けなかった旨表示
  + 9x9行列の結果を表示
* 終了

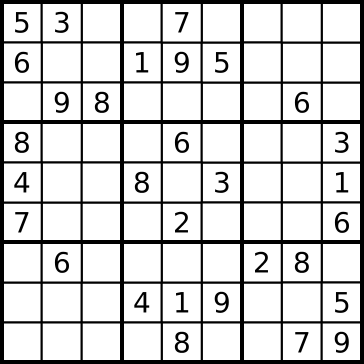
## グリッドデータの初期化

* 数字の表現 ：ビット
* 初期値
  + マスの確定数字 (square\_fixed\_digits)：0
  + マス候補有効数字 (square\_valid\_digits)：0x1FF
  + マス候補 (square\_candidate\_digits)：0
  + 行の確定数字 (row\_fixed\_digits)：0
  + 行候補 (row\_candidate\_digits)：0
  + 列の確定数字 (column\_fixed\_digits)：0
  + 列候補 (column\_candidate\_digits)：0
  + 3x3行列の確定数字 (box\_fixed\_digits)：0
  + 3x3行列候補 (box\_candidate\_digits)：0
  + 未確定のマスの数 (num\_unfixed\_squares)：81 （行数×列数）

### グリッドデータの出力

項目のみ

## 数独表をファイルから読み込む処理 (ReadSudokuFIle)

* 数独表のフォーマット
  + CSV形式 （Excelで問題を作れるように）
  + 空白マス:(文字無し)
  + 数字：1-9
  + マス区切り文字：,
* ファイル名：sudoku\_sample.txt
* フォーマットサンプル

5,3,,,7,,,,

6,,,1,9,5,,,

,9,8,,,,,6,

8,,,,6,,,,3

4,,,8,,3,,,1

7,,,,2,,,,6

,6,,,,,2,8,

,,,4,1,9,,,5

,,,,8,,,7,9

* + Excel2007で数独問題を作成するとき9x9の枠を罫線で描いておくと、csv形式で出力したときに9x9の行列としてcsv出力してくれる。
  + 問題の表と解答の表を読み取れるようにする。
* 初期化処理
  + 読み取りアルゴリズム（要記述）
  + 初期配置数字を確定数字をとして以下 のマスの確定数字にビットを立てる
    - マスの確定数字 (square\_fixed\_digits)
    - 行の確定数字 (row\_fixed\_digits)
    - 列の確定数字 (column\_fixed\_digits)
    - 3x3行列の確定数字 (box\_fixed\_digits)
  + 以下の値を設定する。
    - 未確定のマスの数 (num\_unfixed\_squares) - 1

## できた機能

* ビットフラグを10進数にして表示する機能。

次バージョンに入れ込む案

## 欲しい機能

* マス目の指定はrow,columnを個別に指定するのではなく、座標の構造体にする。
* マス目の構造体はrow, column, box, candidate, fixedにする。
* boxを3x3行列だけでなく、任意の形に対応できるようにする。
  + 問題があるか？
* 様々なナンプレの制約に対応できるようにする。
* gridの入力をやりやすくしたい。
  + ExcelでVBAを使う？専用ツールを作る？

## 反省点

* 関数の分け方は重要。関数名にない処理を中でやらせると、データがどこで変えられたのか分かりづらくなる。構造体の場合、どの関数でどのメンバが変更したのか分からなくなるとプログラムを追うのがとても大変になる。
  + グローバル変数の問題点としてどこで値が変えられたのか分かりづらくなるというのがあるが、値渡しでも同様の問題が生じる。
* データ構造と構造体は似て非なる物。データ構造は構造体よりはざっくりした物。今回上で挙げたgridのデータ構造位の情報量で十分データ構造と言える。
* 構造体はメンバを追加しても構造体を操作する処理に影響を与えない。これは構造体の大きな利点。今回の場合gridに持たせたい状態をgrid構造体に追加すればすぐに対応できた。ただし、メンバを削除したり変更したりするとプログラムの変更が生じる可能性があるので、注意が必要。
* 仕様書はとても大事。プログラムはよりどころがあった方が圧倒的に組みやすい。
* 速度は気にするほどではない。人間が解くのに目標で120分かかる問題を0.008秒で解いた。時間のかかる処理ではないので、今回のプログラムは速度の優先順位は低い。