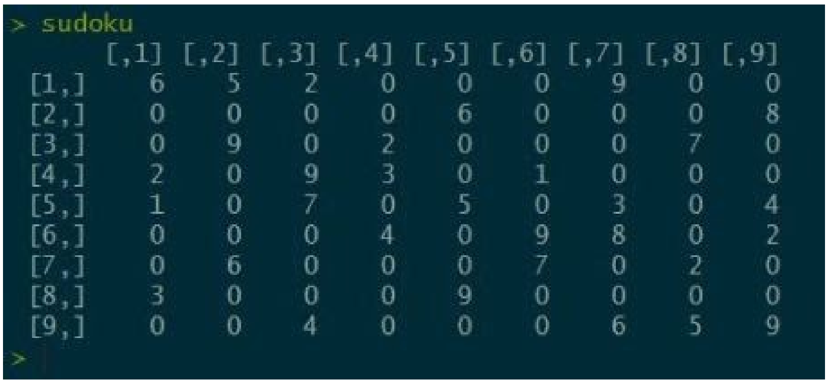
Sudoku is a classical logical game based on combinatorial number replacement puzzle. Objective is to to fill 9×9 matrix with digits so that each column, each row, and each box (3×3 sub-grid) of nine contain all of the digits from 1 to 9.

Solving sometimes can be a nagging process. For this purpose, here is the R helper function for you to solve this with R.



```
> sudoku
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
 [1,]    6    5    2    0    0    0    9    0    0
 [2,]    0    0    0    0    6    0    0    0    8
 [3,]    0    9    0    2    0    0    0    7    0
 [4,]    2    0    9    3    0    1    0    0    0
 [5,]    1    0    7    0    5    0    3    0    4
 [6,]    0    0    0    4    0    9    8    0    2
 [7,]    0    6    0    0    0    7    0    2    0
 [8,]    3    0    0    0    9    0    0    0    0
 [9,]    0    0    4    0    0    0    6    5    9
>
```

Let’s get the Sudoku grid we want to solve into R:

```
sudoku <- matrix(data=c(
6,0,0,2,1,0,0,3,0,
5,0,9,0,0,0,6,0,0,
2,0,0,9,7,0,0,0,4,
0,0,2,3,0,4,0,0,0,
0,6,0,0,5,0,0,9,0,
0,0,0,1,0,9,7,0,0,
9,0,0,0,3,8,0,0,6,
0,0,7,0,0,0,2,0,5,
0,8,0,0,4,2,0,0,9), nrow=9, ncol=9, byrow=FALSE
)
```

Now, we will need a function that will find all the 0 values – these are the values we need to work on.

```
get_zeros <- function(board_su){
  #empty df
  df <- data.frame(i=NULL,j=NULL)
  for (i in 1:nrow(board_su)){
    for (j in 1:ncol(board_su)){
      if (board_su[i,j] == 0) {
        a <- data.frame(i,j)
        #names(a) <- c("i", "j")
        #df <- rbind(df, a)
        df <- a
        return(df)
      }
    }
  }
}
```

In addition we will need a function to solve and validated the solution.

Function validater will validate for the sudoku board a particular solution at a particular position:

```
validater(sudoku, 1, c(1,4))
```

In matrix, at position x=1, y=4, where there is 0, it will test if number 1 is valid or not. If the number is valid, it returns TRUE (number) to outer function for finding complete solution.

This function iterates through all the possible 0-positions and iterates through solutions that are still available based on the rules:

- each row can contain only one number in range of 1..9
- each column can contain only one numer in range of 1..9
- each sub-grid of 3×3 can contain only one number in range of 1..9

And the nuts and bolts of the validater function:

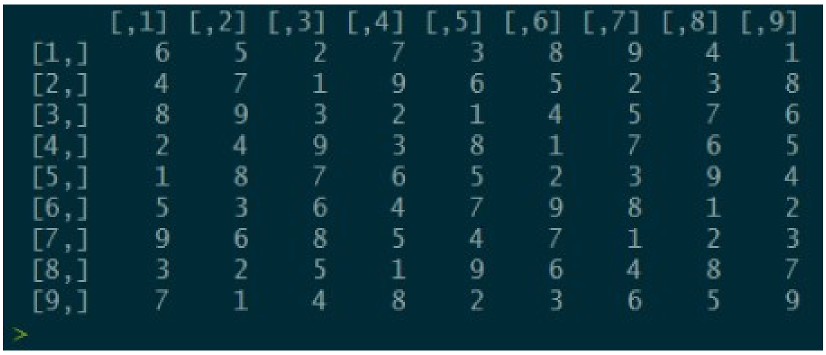
```
validater <- function(board_su, num, pos=c(NULL,NULL)){
  status <- FALSE
  a <- as.integer(pos[1])
  b <- as.integer(pos[2])
  num <- as.integer(num)
  while (status == FALSE) {
    for (i in 1:9) {
      if ((board_su[a,i] == num & b != i) == TRUE) {
        status <- FALSE
        return(status)
      }
    }

    for (i in 1:9) {
      if ((board_su[i,b] == num & a != i) == TRUE) {
        status <- FALSE
        return(status)
      }
    }

    #which box are we in
    boxNx <- as.integer(ifelse(as.integer(b/3)==0, 1, as.integer(b/3)))
    boxNy <- as.integer(ifelse(as.integer(a/3)==0, 1, as.integer(a/3)))

    #looping through the box
    for (i in boxNy*3:(boxNy*3 + 3)) {
      for (j in  boxNx * 3 : (boxNx*3 + 3)) {
        if ((board_su[i, j] == num &  i != a & j != b) == TRUE){
          status <- FALSE
        }
      }
    }
    status <- TRUE
    return(status)
  }
}
```

With the following solution:



```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
 [1,]    6    5    2    7    3    8    9    4    1
 [2,]    4    7    1    9    6    5    2    3    8
 [3,]    8    9    3    2    1    4    5    7    6
 [4,]    2    4    9    3    8    1    7    6    5
 [5,]    1    8    7    6    5    2    3    9    4
 [6,]    5    3    6    4    7    9    8    1    2
 [7,]    9    6    8    5    4    7    1    2    3
 [8,]    3    2    5    1    9    6    4    8    7
 [9,]    7    1    4    8    2    3    6    5    9
>
```

For sure, this is not to be taken seriously, as you get the application on your mobile phone where you make a photo of your grid to be solved and the phone solves it for you, using library like OpenCV. The code was created only and just for fun (and because the Advent of Code for 2019 is over).