

Forum:[Hack Forums](#)

Thread: [Square Matrix Program Generator](#)

Computronium - April 10, 2018, 8:45 pm

Hello all. I am attempting to create a square matrix that can do the following in my self-teaching:

I will prompt the user to enter the length of a square matrix.

After that, it will randomly fill in 0s and 1s into the matrix

Next we system.out.print the square matrix

Finally, find and print all of the rows, columns and diagonals with all 0s or 1s

Please help. I am totally lost. I could really use your help.

m0noxide - April 10, 2018, 10:39 pm

I get that you're totally lost but can you at least start it, perhaps the first two steps or so?

Which step are you stuck at, exactly?

Then we can properly help you instead of just giving you the answer

GenericSkid - April 10, 2018, 11:18 pm

Expected output:

Code Enter grid size: 3

```
111
100
111
```

```
Row[1]: 111
Row[3]: 111
```

```
col[1]: 111Code Enter grid size: 3
```

```
011
011
101
```

```
col[3]: 111
```

```
Diagonal-Backward: 111
```

Without diving straight into the code this can be broken into a few parts.

Reading grid size

Looping through grid to set values between 0-1

Looping through grid to find rows that have the same value all the way through

Looping through grid to find columns that have the same value all the way through

Looping through diagonals to find diagonals that have the same value all the way through

1: Reading grid size

This is the easiest part (can be done in one line of code). Input is usually done through the `java.util.Scanner` class. In this case since we want a number, we can use the method `nextInt()`. Store the value in some variable.

2: Populating the grid

Next up we make the grid and populate it. To make a square grid you need only make a new array of some type such as `Type[N][N]`. You can think of this as equivalent to `Type[Rows][Columns]`.

To populate the grid you can use two for loop to iterate over rows and columns.

PHP Code

```
<?php
for (r : rows)
    for(c : columns)
        grid[r][c] = X;
```

Since we want X in this case to be a random number between 0 and 1 we can use the `java.util.Random` class, which has the method `nextInt(K+1)` where K is the max value you want to have (Such as one).

You can print the grid in these loops too. After setting the value you can call `System.out.print(grid[r][c])` in the column loop. If our square size is 3 an example would look like: 110
To print the next row, place an empty `System.out.println()` after the column loop ends. Again, if our square size is 3 we could see something like:

Code 110

011

101

3 & 4: Printing rows and columns with matching numbers

I'm going to include parts 3 & 4 from the part list together. The code for each is nearly identical. Anyways I'll get on that later.

PHP Code

```
<?php
// print rows with same values, such as: 000, 111
for (r : rows)
    // row-specific variables
    for(c : columns)
        // check if grid[r][c] == grid[r][c - 1]
        // note: there are many ways to check if the current value is = to the last. I leave this up to you.
```

Here we have our row/column loops and we want to know which rows have consecutive values. In this case we should have a row-specific value for the value of the repeated number. For example we start with column c, row 0: `val = grid[c][0]`. Our column loop only needs to iterate over the remaining rows, for example if our grid size is 3, then we need only check 1 and 2 (So basically, same loop as before, but `c = 1` instead of `c = 0`)

We can then check if the grid at `[row][column]` is equal to our val. If it is not this row is invalid. If we finish the loop and it has been the same each time, we just need to print our val as many times as our grid is wide/tall. For example:

Code 001

111

100

row[2]: 111

This same process is repeated for finding columns, but instead we use the following loop:

PHP Code

```
<?php
```

```
// print rows with same values, such as: 000, 111
```

```
for (c : columns)
```

```
    // column-specific variables
```

```
    for(r : rows)
```

```
        // check if grid[r][c] == grid[r-1][c]
```

```
        // note: there are many ways to check if the current value is = to the last. I leave this up to you.
```

Note that the only change is that we are iterating over columns in the outer loop instead of rows. The logic of the inner loop is mostly the same. You should be able to piece it together.

5: Printing diagonals with matching numbers

I would be pushing it if I put 3 parts into one section, but this is also like the previous sections (with one small change). We only one loop since since we are going diagonally across the grid. For example:

PHP Code

```
<?php
```

```
// print rows with same values, such as: 000, 111
```

```
for (d : grid-size)
```

```
    grid[d][d]; // follows the grid like \
```

```
    grid[d][(size-1) - d]; // follows the grid like /
```

The code for checking consecutive value is just like before. Before where we had row/column specific variables we have diagonal-specific values which are placed outside of the loop (Since we only have one loop versus embedded loops).

The important part here is knowing how to reference grid values. If you want to access the top-left of the grid [0][0]. And for the top-right: [0][size-1]. The key for the loop is that if you want to follow the grid in a diagonal shaped like / then you have to do (size-1) - d.

To show why we do (size-1) remember that if our grid is new Type[size][size] referencing Type[size][size] will result in an index-out-of-bounds error if d=0 and we use [d][size-d] == [0][3-0].

So to summarize, this is mostly the same code as before, just with some indexing logic.

Full code: pastebin.com/TGemwA5S

Computronium - April 11, 2018, 9:48 am

Awesome, great! Thanks, Skid. I will work on this program over the course of a week or two since I have plans currently. I will post out to here any code I have or need help with.