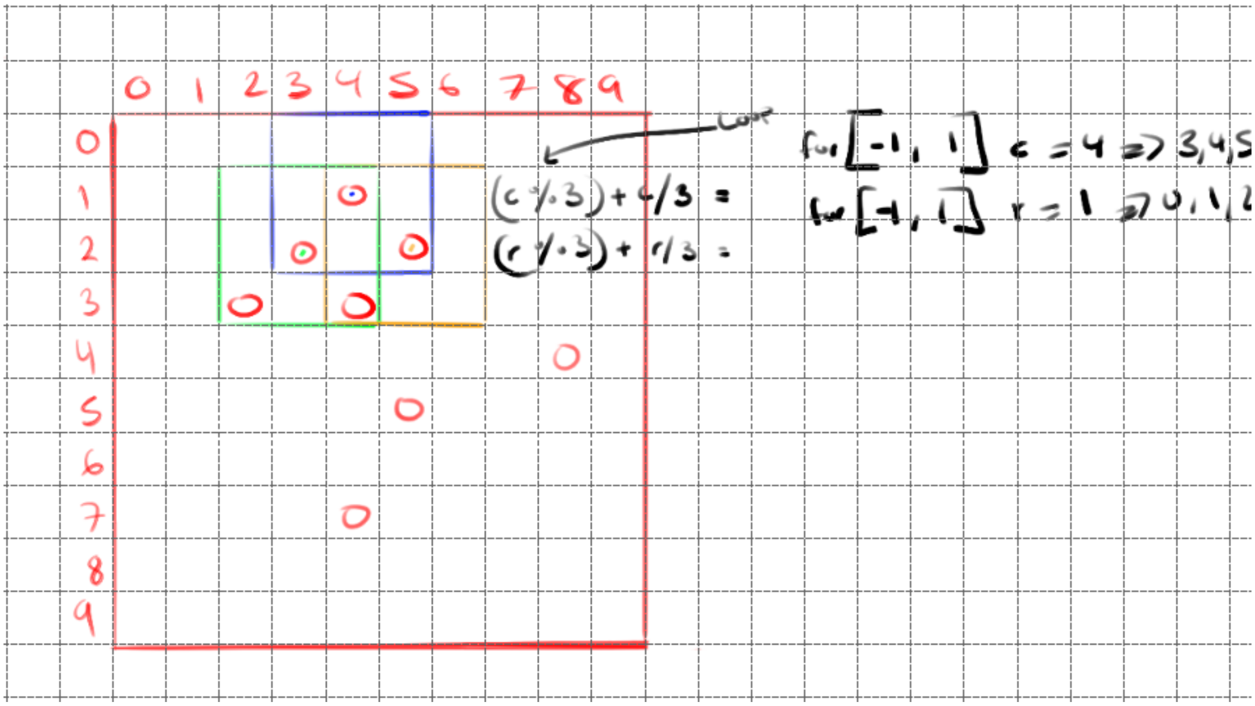
#inThe Game of Life

Rules:

1. Any live cells with 2 or 3 neighbors survive
2. Any dead cell with exactly 3 neighbors becomes a live cell
3. All other cells die

2-D array of True/False values



```
Function Surivival of 1 Cell(Universe *u)
for R += [-1, 1] rows:
        for C += [-1, 1] cols:
                If (R in range[0, Rows) and C in range [0, Cols)):
                        If (alive) neighbors++:
                Else If (Toroidal):
                        If R, C < 0: Rows - R, Cols - C
                        If R, C >= Rows, Cols
```

If neighbors == 3, LIVE cell

If neighbors == 2, if RC is alive LIVE cell

Else DEAD

```
Next Generation
For r in Rows:
    For c in Cols:
        Surivival of 1 Cell(Universe *u)

Main Function:
    Universe u      //even generations
    Universe uo     //odd generations
    // Alternate universes to not override.
```

```
<stdlib.h>      //calloc
<stdbool.h>     //boolean

struct Universe:
    int rows
    int cols
    bool **grid
    bool toroidal

Universe *uv_create(int rows, int cols, bool toroidal):
    bool **matrix = (bool **) calloc(rows, sizeof(bool *))
    for (int r = 0; r < rows; r+1):
        matrix[r] = (bool *) calloc(cols, sizeof(bool))
    return matrix

void uv_delete(Universe *u):
    int rows = uv_rows(u)
    for (int r = 0; r < rows; r+1):     //for the number of rows
        free([r])                       //free the column

void int uv_rows(Universe *u):
    return u -> rows

void int uv_cols(Universe *u):
    return u -> cols

void int uv_live_cell(Universe *u, int r, int c):
    u->grid[r][c] = true

void int uv_dead_cell(Universe *u, int r, int c):
    u->grid[r][c] = false
```

```
bool uv_get_cell(Universe *u, int r, int c):
    return u->grid[r][c]

int uv_census(Universe *u, int r, int c):
    rows = uv_rows(Universe *u)
    cols =  uv_cols(Universe *u)
    int population = 0;
    for (int r = 0; r < rows; r+1):
        for (int c = 0; c < cols; r+1):
            if(u->grid[r][c]): population++
    return population

void uv_print(Universe *u, FILE *outfile):
    for (int r = 0; r < rows; r+1):
        for (int c = 0; c < cols; r+1):
            if (uv_get_cell(u, r, u)):
                fprintf(outfile, “o”)
            else:
                fprintf(outfile, “ ”)
        fprintf(outfile, “\n”)
```