



Design Document  
Overview  
Alexander Ragland  
CSE 13

Universe (struct)

- Number of rows = x
- Number of columns = y
- Toroidal = true or false
- Grid

[  
[ T , F , T ]  
,  
[ F , F , T ]  
]

# universe.c

```
struct Universe;
{...};

Universe *uv_create(uint32_t rows, uint32_t cols, bool toroidal);
    Universe *uv = (Universe *) malloc(sizeof Universe)
    If (memory alloc success)
        uv-> rows = rows;
        uv-> cols = cols;
        uv-> toroidal = toroidal;
        uv-> grid = (bool **) calloc(rows, sizeof (bool *))
        For (r=0,r<rows,r+=1)
            uv->grid[r] = (uint32_t *) calloc(cols, sizeof uint32t)
    Return u;

void uv_delete(Universe *u);
    For loop (free each row)
    Free grid itself

uint32_t uv_rows(Universe *u);
    Return u->rows;

uint32_t uv_cols(Universe *u);
    Return u->cols;

void uv_live_cell(Universe *u, uint32_t r, uint32_t c);
    Check if u->grid[r][c] is in bounds;
    If yes, u->grid[r][c] = true;

void uv_dead_cell(Universe *u, uint32_t r, uint32_t c);
    Check if u->grid[r][c] is in bounds;
    If yes, u->grid[r][c] = false;

bool uv_get_cell(Universe *u, uint32_t r, uint32_t c);
    Check if u->grid[r][c] is in bounds; return false if not
    return (bool) u->grid[r][c];

bool uv_populate(Universe *u, FILE *infile);
    While loop (until EOF)
        fscan()
        L = {data}
        R = {data}
        If linenummer == 1:
            u->rows = L
            u->cols = R
        If (L > u->rows) or (R > u->cols)
            print(failed to populate universe: cell not within
                bounds of universe)
        Return false
    uv_live_cell(u, L, R)
    Return true;

uint32_t uv_census(Universe *u, uint32_t r, uint32_t c);
    If not toroidal
        Neighbor Counter = 0
        For loop (r-1, r+1)
            For loop (c-1, c+1)
                If u[r][c] is not the cell itself and it is true (alive)
                    Neighbor counter ++
    If toroidal
        Cry in modulus

void uv_print(Universe *u, FILE *outfile);
    For rows in u
        For cols in u
            If u[r][c] is true
                Print o
            If u[r][c] is false
                Print .
```

# life.c

```
#define OPTIONS “”
```

Set defaults

```
main():
```

```
    getopt()...
```

```
    Declare universe
```

```
    Declare universeb
```

```
    Populate universea()...
```

```
        Ncurses example...
```

```
            Do generation stuff
```

```
        End ncurses
```

```
    uv_print(universea)
```

# Makefile

CC = clang-15

CFLAGS = -Wall -Wextra -Werror -Wpedantic

OBJECTS = life.o universe.o

LINKS =

All: life

life: \$(OBJECTS)

\$(CC) -o sorting sorting.o \$(LINKS)

life.o: life.c

\$(CC) \$(CFLAGS) -c life.c

universe.o: universe.c

\$(CC) \$(CFLAGS) -c universe.c

format:

clang-format-15 -i -style=file \*.ch

clean:

rm -f life \*.o