2 Videos to Start With

https://www.youtube.com/watch?v=FdMzngWchDk

https://www.youtube.com/watch?v=k2IZ1qsx4CM
1-Dimensional Game of Life
Rules of 1D-Life

**Rule 1**: 0 neighbors
Rule 1: 0 neighbors

Underpopulation
Rules of 1D-Life

Rule 2: 1 neighbor
Rules of 1D-Life

Rule 2: 1 neighbor

Right Environment
Rules of 1D-Life

Rule 3: 2 neighbors
Rules of 1D-Life

Rule 3: 2 neighbors

Overpopulation
Example
A 1-D Version
A 1-D Version
A 1-D Version
Problem of the Day(s):
Implement 1D Game of Life in Assembly!
How to Approach This?

https://img.clipartfest.com/db77689f2cfc577629ec3ff678465323 Managed IT Services NJ IT Person with Question Mark Clipart 4100-6000.jpeg
#Step 1: Write Algorithm in an more "comfortable" language...
# 1-Dimensional Game of Life

```
# initial dish
dish = " # # # # # ###### # # # # # # # # # "
N = len(dish)

# new dish used to compute next generation
newDish = N * " "

# iterate over all generations
for i in range(numGen):
    # print current/new dish
    print(dish)

    # get ready to compute next generation
    newDish = " "  # first and last cells always dead

    for j in range(1, N-2):  # skip 1st and last
        count = 0
        if dish[j-1] == '#': count += 1
        if dish[j+1] == '#': count += 1
        if count == 1:
            newDish = newDish + '#'
        else:
            newDish = newDish + ' '  # add Nth cell as dead to newDish

    newDish = newDish + " "

    # next generation becomes current
    dish = newDish
```

getcopy GameOfLife1D.py
#! /usr/bin/env python3
# gameOfLife.py
# D. Thiebaut
# 1-Dimensional Game of Life
from __future__ import print_function
from __future__ import division

numGen = 20  # number of generations

# initial dish
dish = [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0]
N = len(dish)

# new dish used to compute next generation
newDish = N * [0]

# iterate over all generations
for i in range(numGen):

    # print current/new dish
    print("".join([chr(ord(' ') + c) for c in dish]))

    # get ready to compute next generation
    newDish[0] = newDish[N - 1] = 0

    for j in range(1, N - 1):
        count = dish[j - 1] ^ dish[j + 1]
        newDish[j] = count

    # next generation becomes current
    dish = newDish

def GameOfLife1D_V2():
    # same version but without tests
    getcopy GameOfLife1D_V2.py
Sierpinski Gasket

http://www.lutanho.net/fractal/sierpa.html
Back to Game of Life
Python to Assembly

• Develop Python solution

• Copy Python solution in assembly program and comment out all the lines

• Take groups of Python statements and translate into assembly

• Python program becomes natural comments for the assembly
Develop Assembly Program as a Class Exercise
If-statements in Assembly
Outline

• Jmp: the jump instruction
• flags register
• conditional jumps (jne, je, jgt, jge, jlt, jle, ja, jb…)

D. Thiebaut, Computer Science, Smith College
Jumping around...

Start:

```
mov    ebx, Table          ;
jmp    there               ;
here:  mov     al, 1               ;
       mov     ecx, N              ;
there: mov     byte[ebx+esi], al   ;
       inc     esi                 ;
       add     al, al               ;
jmp    here                ;
```
Jumping around...

_Start:

    mov     ebx, Table                           ;
    jmp     there                               ;

_here:  mov     al, 1                           ;
         mov     ecx, N                         ;

_there: mov     byte[ebx+esi], al            ;
        inc     esi                         ;
        add     al, al                      ;
        jmp     here                       ;
Jumping around…

_Start:

mov    ebx, Table ;
jmp    there ;

here:  mov    al, 1 ;
mov    ecx, N ;

there: mov    byte[ebx+esi], al ;
inc    esi ;
add    al, al ;
jmp    here ;
Jumping around…

_Start:
  mov     ebx, Table          ;
  jmp _there

_here:
  mov     al, 1               ;
  mov     ecx, N              ;

_there:
  mov     byte[ebx+esi], al   ;
  inc     esi                 ;
  add     al, al              ;
  jmp _here

Jumping around...

_Start:

    mov     ebx, Table          ;
    jmp     there             ;

here:   mov     al, 1                   ;
        mov     ecx, N            ;

there:  mov     byte[ebx+esi], al    ;
        inc     esi             ;
        add     al, al          ;
        jmp     here           ;
Jumping around…

_Start:
    mov     ebx, Table         ;
    jmp     there              ;

here:   mov     al, 1             ;
        mov     ecx, N          ;

there:  mov     byte[ebx+esi], al ;
        inc     esi           ;
        add     al, al        ;
        jmp     here          ;
Jumping around…

_Start:

```
mov     ebx, Table          ;
jmp     there

here:   mov     al, 1               ;
        mov     ecx, N              ;

there:  mov     byte[ebx+esi], al  ;
        inc     esi                 ;
        add     al, al              ;
jmp     here
```
Jumping around...

```
_Start:
    mov   ebx, Table          ;
    jmp   there              ;

here:  mov   al, 1            ;
       mov   ecx, N          ;

there: mov   byte[ebx+esi], al ;
       inc   esi            ;
       add   al, al         ;
    jmp   here              ;
```
jmp there ; "mov eip,there"
Flags Register
Examples

```assembly
_start:  nop
     nop

; immediate  Flag values
; value       AFTER the instruction

mov  al, 0x43  ; 67
sub  al, 0x43  ; PF ZF IF ID

mov  al, 0x43  ; 67
sub  al, 0x42  ; 66  IF ID

mov  al, 0x43  ; 67
sub  al, 0x44  ; 68  CF PF AF SF IF ID

mov  al, 0x43  ; 67
sub  al, 0xff  ; 255 or -1  CF PF AF IF ID

mov  al, 0x43  ; 67
sub  al, 0x81  ; 129 or -127  CF SF IF OF ID
```
We need an instruction that will jump to some place other than the next instruction if one or more of the flags bits are set a particular way.

For example, it would be great to see what flag bits are set when the result of a subtraction is positive or zero, and create a special jump instruction that jumps to some label only if these bits are set.
Meet the Conditional Jumps!

- je, jz
- jne, jnz
- jl
- jle
- jg
- jge
Meet the CMP instruction!

- `cmp op1, op2` ; `op1 - op2, set the flag bits`
We stopped here last time...
Examples

; if eax == ebx:
;    count += 1
;

; if dish[j] == 1:
;    dishString[j] = '#'
; else
;    dishString[j] = ' '
; # c is a char
; if c >= 'a' and c <= 'z':
;   c = chr( ord(c) - 32 )
; ...

Examples
Examples

; if x > 0xFFFFFFFF:
;    z = x+2
; else:
;    z = x-1
Meet the Conditional Jumps!

- je, jz
- jne, jnz
- jl
- jle
- jg
- jge
- je, jz
- jne, jnz
- jb
- jbe
- ja
- jae