quine.sl: A self-shading shader

Dan Piponi
Industrial Light and Magic

6th Century BC

All Cretans are liars

This sentence is false

Quine's Paradox

- "Yields falsehood when preceded by its quotation" yields falsehood when preceded by its quotation

A quine is...

- ...a program that prints out its own source code.
- It shouldn't 'cheat' by reading its own source code directly from memory or from a file.

Not a quine

```
10 LIST
```

Trying to write quines

```
print “print \“print ...\””
```

A ‘C’ Solution

```
char*f="char*f=%c%s%c;main(){printf(f,34,f,34,10);}%c"
;main(){printf(f,34,f,34,10);}
```


```
char*f="char*f=%c%s%c;main(){printf(f,34,f,34,10);}%c"
;main(){printf(f,34,f,34,10);}
```

APL

O'Caml

BASIC

C++

Python

Javascript

Lua

C

Intercal

Tcl

Brainf***

Rexx

8086

Java

Renderman Shader Language?

Forth

Pascal

Scheme

Prolog

Perl

Lisp

Postscript

Fortran

Haskell

Cobol

What Should it Do?

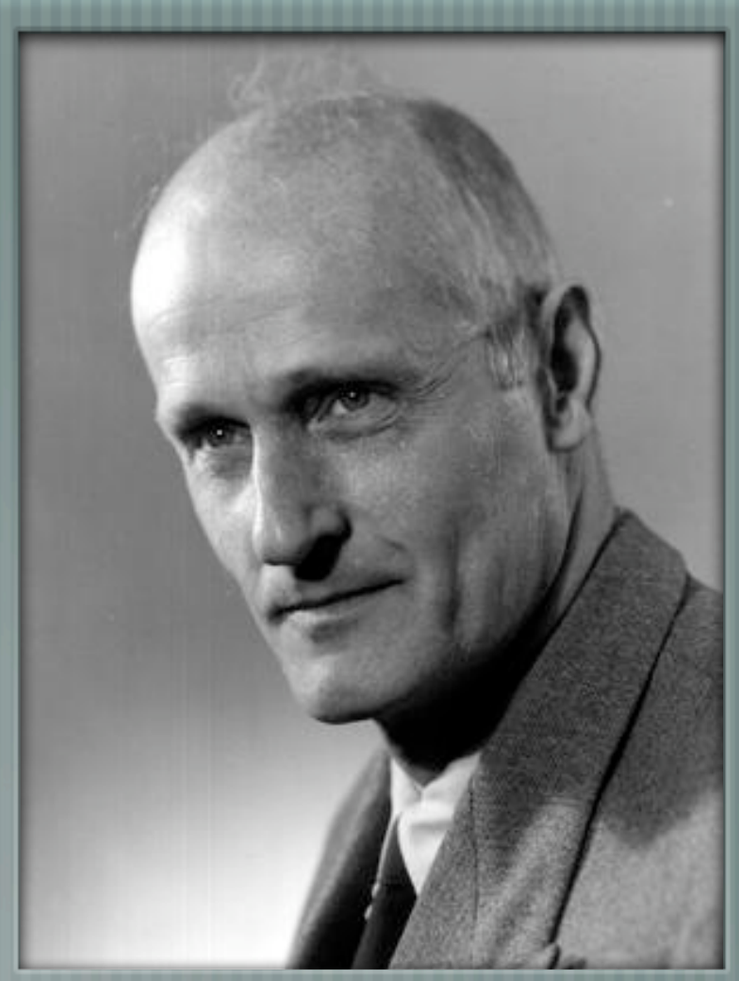
- Should be idiomatic. Renderman shaders can print, but really it should shade.
- Shouldn’t use DSOs, store source code in texture or point clouds.

What are the problems?

- No font
- No typesetting
- Limited string handling

S. C. Kleene

Corollary of Second Recursion Theorem



Font

- Find a bit mapped font
- Store it as an array of floats
- Write binary bit-twiddling rotuines for floating point numbers

String Handling

- Forget strings
- Use arrays of floats

Typesetting

- Lay out code in precisely 80 columns so we get a simple geometric mapping from ‘code space’ to texture space.

Results

Further Research

- This quine is long. How short can we get?
- Can we do something similar with RIB files? Not really a programming language but there may be some possibilities.

References and Credits

- *Gödel, Escher, Bach* Douglas R Hofstadter
- *Vicious Circles* John Barwise and Lawrence Moss
- *The Stupidest Shader I Ever Written* Mach Kobayashi *SRTX 2007*
- Font thanks to John Hall (RIP) http://overcode.yak.net/12


// the rest of the code is what turns the representation above into readable
float digit(float pi float x)
float charAt(uniform float data[];float x;float y)
float c;
Font thanks to John Hall
http://overcode.yak.net/12
uniform float font[] = {