SCC.111 Software Development – Lecture Lecture 20: Fun by the C

Adrian Friday, Nigel Davies, Hansi Hettiarachchi, Saad Ezzini

SCC.111 Software Development – Lecture Lecture 20: The Christmas Lecture

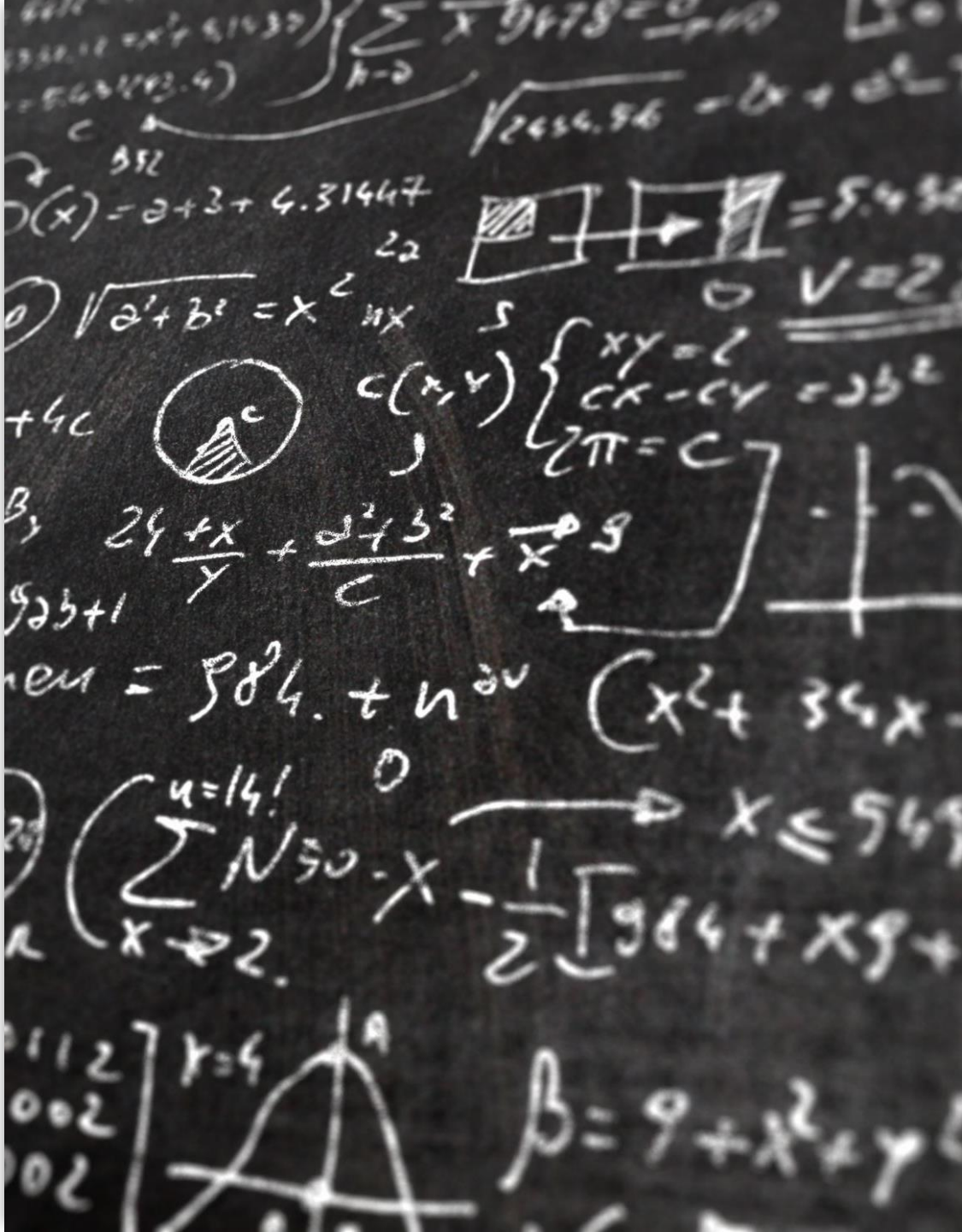
Adrian Friday, Nigel Davies, Hansi Hettiarachchi, Saad Ezzini

The best way to prepare [to be a programmer] is to write programs, and to study great programs that other people have written.

Bill Gates

This lecture

- Enjoy looking at some really challenging C code - and see the power of the language.



C’s hidden pipeline

When we build in C

- gcc -o main main.c

The pre-processor is *really* useful!

- Define constants
  - Unchangable values (TRUE, FALSE, MAXINT, MAX_HIGHSCORES, ROCK, M_PI)
- Define macros
  - Expansions and definitions of useful things
  - #define MIN(X,Y) (X) < (Y) ? (X) : (Y)
- Include other files
  - #include – of course!
- Conditional compilation

Defining constant things

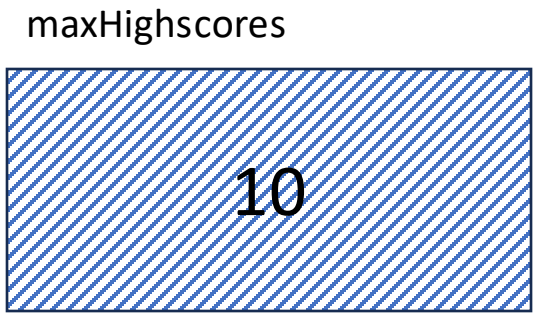
- Define constants
  - #define FALSE 0
  - #define TRUE !FALSE

  - Textually replaces ‘FALSE’ with 0
  - ‘Scope free’ (i.e. does not understand C at all!)
  - Does not affect compile time performance

Defining constant things

- Alternative,
  - #define MAX_HIGHSCORES 10
  - const int maxHighscores = 10;

  - The const int is a ‘non-variable variable’
  - Compile time
  - So, type is known and *checked* by the compiler



Macros

- Expansions and definitions of useful things
  - #define MIN(X,Y) (X) < (Y) ? (X) : (Y)

  - NOT a function!
  - A textual expansion of MIN(X,Y) – in this case to *an inline conditional expression*
  - Note how we've done X as (X) – *why?*

Conditional compilation

- Conditional compilation
  - Typically used to exclude code

```
#if 0

// ignore all this

#endif
```

Typically used to protect header files from multiple inclusion

```
#ifndef __MINISET_H

#define __MINISET_H

// conditionally do me

#endif
```

How small can you go ?

%d, %s, %c – nice and easy…

%n – with great power …

```c
int main()
{
  int x;
  printf("I just printed %n characters.\n", &x);
}
```

The International Obfuscated C Code Contest

The 27th IOCCC

2020 marked the “The Twenty Seventh International Obfuscated C Code Contest”

Standard IOCCC stuff

The primary IOCCC web site can be found at,

> http://www.ioccc.org/

Use make to compile entries. It is possible that on non-Un*x / non-Linux systems the makefile needs to be changed. See the Makefile for details.

Look at the source and try to figure out what the programs do, and run them with various inputs. If you want to, look at the hints files for spoilers - this year we included most of the information included by the submitter.

Read over the makefile for compile/build issues. Your system may require certain changes (add or remove a library, add or remove a #define).

Some C compilers are not quite as good as they should be. If yours is lacking, you may need to compile using clang or gcc instead of your local compiler.

```
char d[538] = {1,0,10,0,10};

int main()
{
  while(*d)
    printf(fmt, arg);
}
```

```
#define N(a) "%"#a"$hhn"
#define O(a,b) "%10$"#a"d"N(b)
#define U "%10$.*37$d"
#define G(a) "%"#a"$s"
#define H(a,b) G(a)G(b)
#define T(a) a a
#define s(a) T(a)T(a)
#define A(a) s(a)T(a)a
#define n(a) A(a)a
#define D(a) n(a)A(a)
#define C(a) D(a)a
#define R C(C(N(12)G(12)))
#define o(a,b,c) C(H(a,a))D(G(a))C(H(b,b)G(b))n(G(b))O(32,c)R
#define SS O(78,55)R "\n\033[2J\n%26$s";
#define E(a,b,c,d) H(a,b)G(c)O(253,11)R G(11)O(255,11)R H(11,d)N(d)O(253,35)R
#define S(a,b) O(254,11)H(a,b)N(68)R G(68)O(255,68)N(12)H(12,68)G(67)N(67)
```

```
char* fmt = O(10,39)N(40)N(41)N(42)N(43)N(66)N(69)N(24)O(22,65)O(5,70)O(8,44)N( 45)N(46)N
(47)N(48)N( 49)N( 50)N( 51)N(52)N(53 )O( 28, 54)O(5, 55) O(2, 56)O(3,57)O( 4,58 )O(13, 73)O(4, 71 )N(
72)O (20,59 )N(60)N(61)N( 62)N (63)N (64)R R E(1,2, 3,13 )E(4, 5,6,13)E(7,8,9 ,13)E(1,4 ,7,13)E (2,5,8,
13)E( 3,6,9,13)E(1,5, 9,13)E(3 ,5,7,13 )E(14,15, 16,23) E(17,18,19,23)E( 20, 21, 22,23)E
(14,17,20,23)E(15, 18,21,23)E(16,19, 22 ,23)E( 14, 18, 22,23)E(16,18,20, 23)R U O(255 ,38)R G ( 38)O(
255,36) R H(13,23)O(255, 11)R H(11,36) O(254 ,36) R G( 36 ) O( 255,36)R S(1,14 )S(2,15)S(3, 16)S(4, 17
)S (5, 18)S(6, 19)S(7,20)S(8, 21)S(9 ,22)H(13,23 )H(36, 67 )N(11)R G(11)""O(255, 25 )R s(C(G(11) ))n (G(
11) )G( 11)N(54)R C( "aa") s(A( G(25)))T (G(25))N (69)R o (14,1,26)o( 15, 2, 27)o (16,3,28 )o( 17,4,
29)o(18 ,5,30)o(19 ,6,31)o( 20,7,32)o (21,8,33)o (22 ,9, 34)n(C(U) )N( 68)R H( 36,13)G(23) N(11)R C(D(
G(11))) D(G(11))G(68)N(68)R G(68)O(49,35)R H(13,23)G(67)N(11)R C(H(11,11)G(
11))A(G(11))C(H(36,36)G(36))s(G(36))O(32,58)R C(D(G(36)))A(G(36))SS

#define arg d+6,d+8,d+10,d+12,d+14,d+16,d+18,d+20,d+22,0,d+46,d+52,d+48,d+24,d\
+26,d+28,d+30,d+32,d+34,d+36,d+38,d+40,d+50,(scanf(d+126,d+4),d+(6\ -2)+18*(1-
d[2]%2)+d[4]*2),d,d+66,d+68,d+70, d+78,d+80,d+82,d+90,d+\
92,d+94,d+97,d+54,d[2],d+2,d+71,d+77,d+83,d+89,d+95,d+72,d+73,d+74\
,d+75,d+76,d+84,d+85,d+86,d+87,d+88,d+100,d+101,d+96,d+102,d+99,d+\
67,d+69,d+79,d+81,d+91,d+93,d+98,d+103,d+58,d+60,d+98,d+126,d+127,\ d+128,d+129
```

I am a research scientist at Google DeepMind (formerly at Google Brain) working at the intersection of machine learning and computer security. My most recent line of work studies properties of neural networks from an adversarial perspective. I received my Ph.D. from UC Berkeley in 2018, and my B.A. in computer science and mathematics (also from UC Berkeley) in 2013.

Generally, I am interested in developing attacks on machine learning systems; most of my work develops attacks demonstrating security and privacy risks of these systems. I have received best paper awards at USENIX Security, IEEE S&P, and ICML, and my work has been featured in the New York Times, the BBC, Nature Magazine, Science Magazine, Wired, and Popular Science.

When not otherwise busy with research, I write lots of useless code ranging from an obfuscated Tic-Tac-Toe Game written in a single call to `printf` (which won the IOCCC 2020 Best of Show), to a Doom clone in 13k of WebGL + JavaScript, to a fully functional CPU built on top of Conway's Game of Life.

Nicholas Carlini
Research Scientist, Google DeepMind
nicholas [at] carlini [dot] com
GitHub | Google Scholar

A complete list of my publications are online, along with some of my code, and some extra writings.

What are you pointing at ?

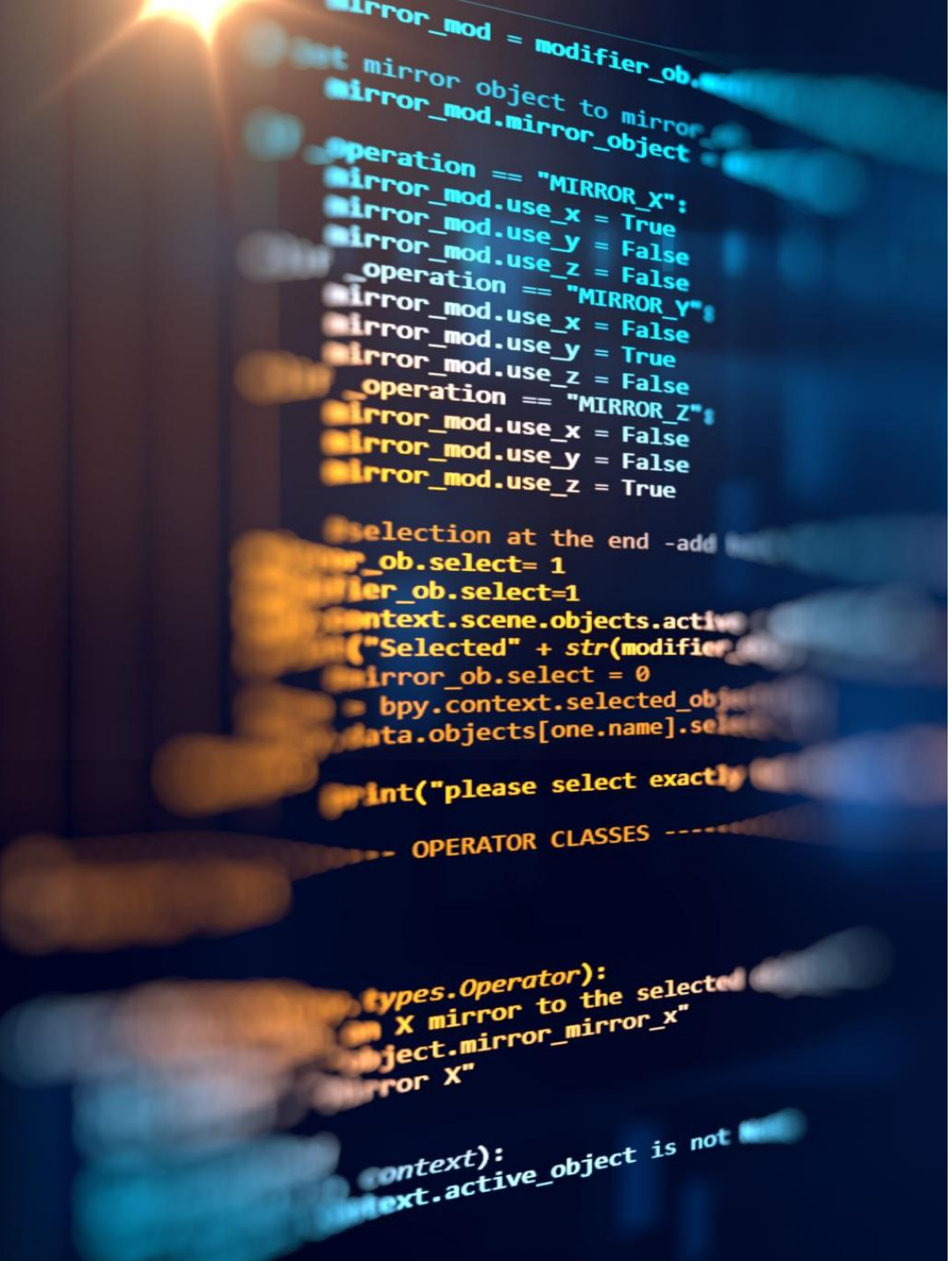
What does this do ?

```c
#include<stdio.h>

char *c[] = { "ENTER", "NEW", "POINT", "FIRST" };
char **cp[] = { c+3, c+2, c+1, c };
char ***cpp = cp;

main()
{
  printf("%s", **++cpp);
  printf("%s ", *--*++cpp+3);
  printf("%s", *cpp[-2]+3);
  printf("%s\n", cpp[-1][-1]+1);

  return 0;
}
```



Summary

- We hope you've enjoyed learning software development and C with us so far!
  - Next term you can look forward to more with Joe and Saad
  - Focusing on introducing Object Oriented programming and higher level language programming

Have a great Christmas !