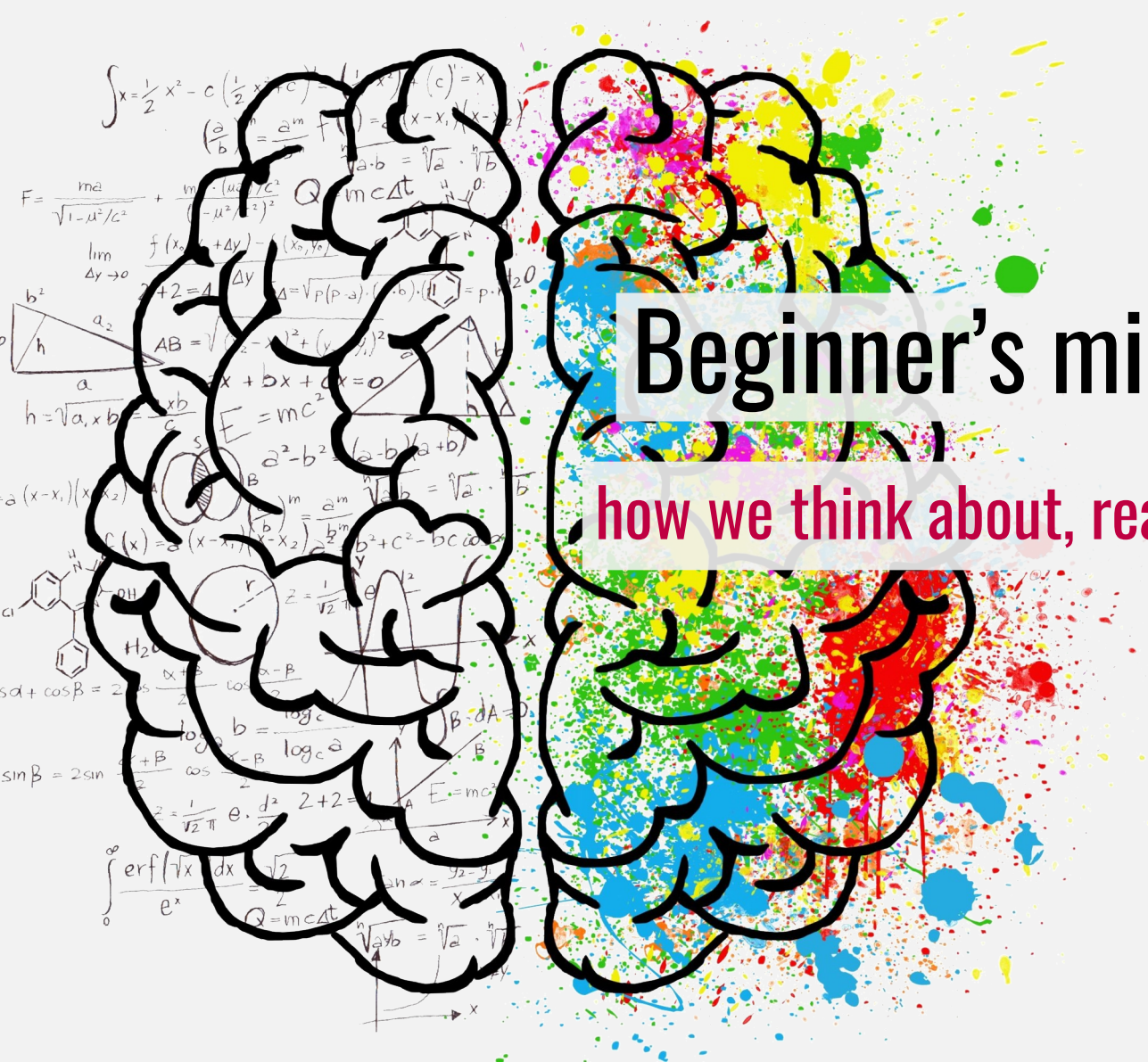
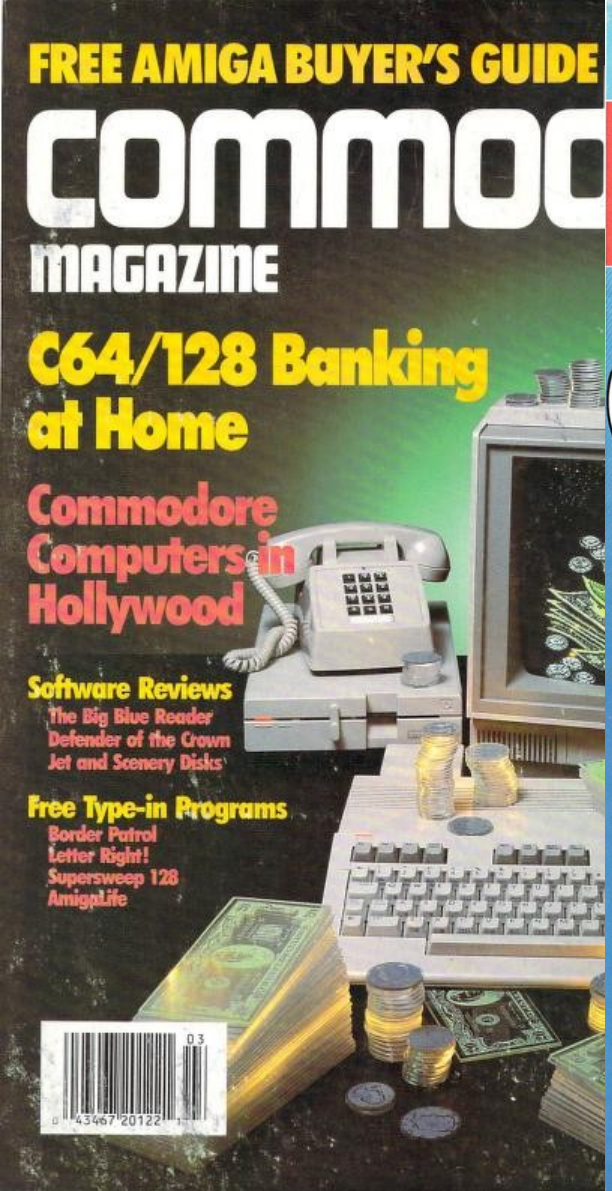


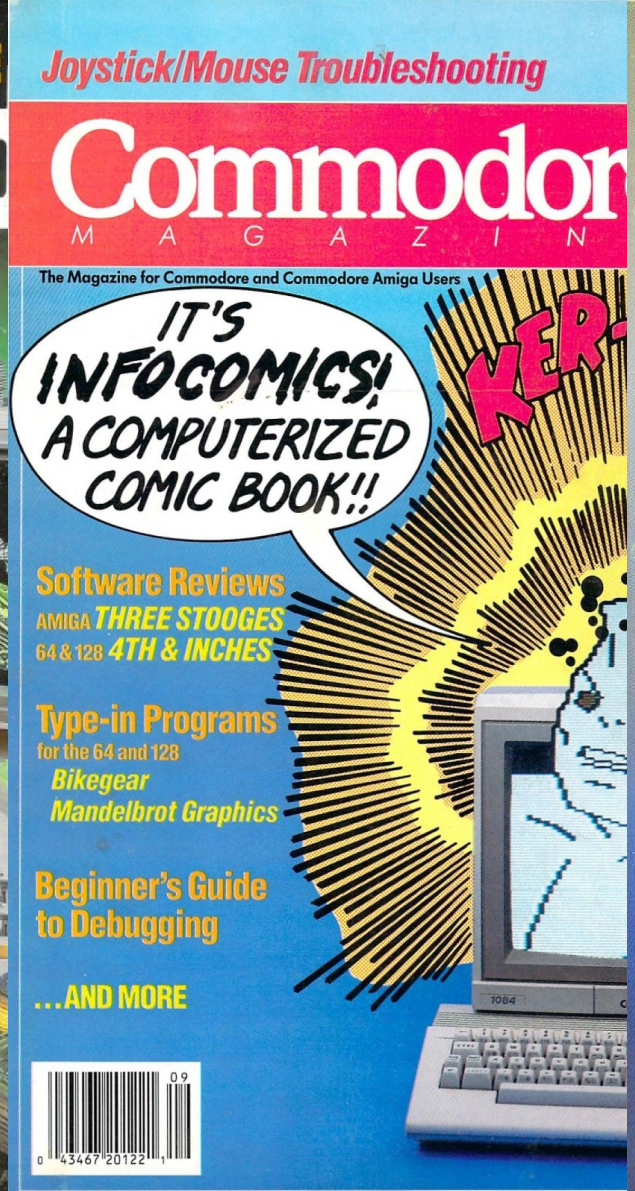
Beginner's mind, expert's mind

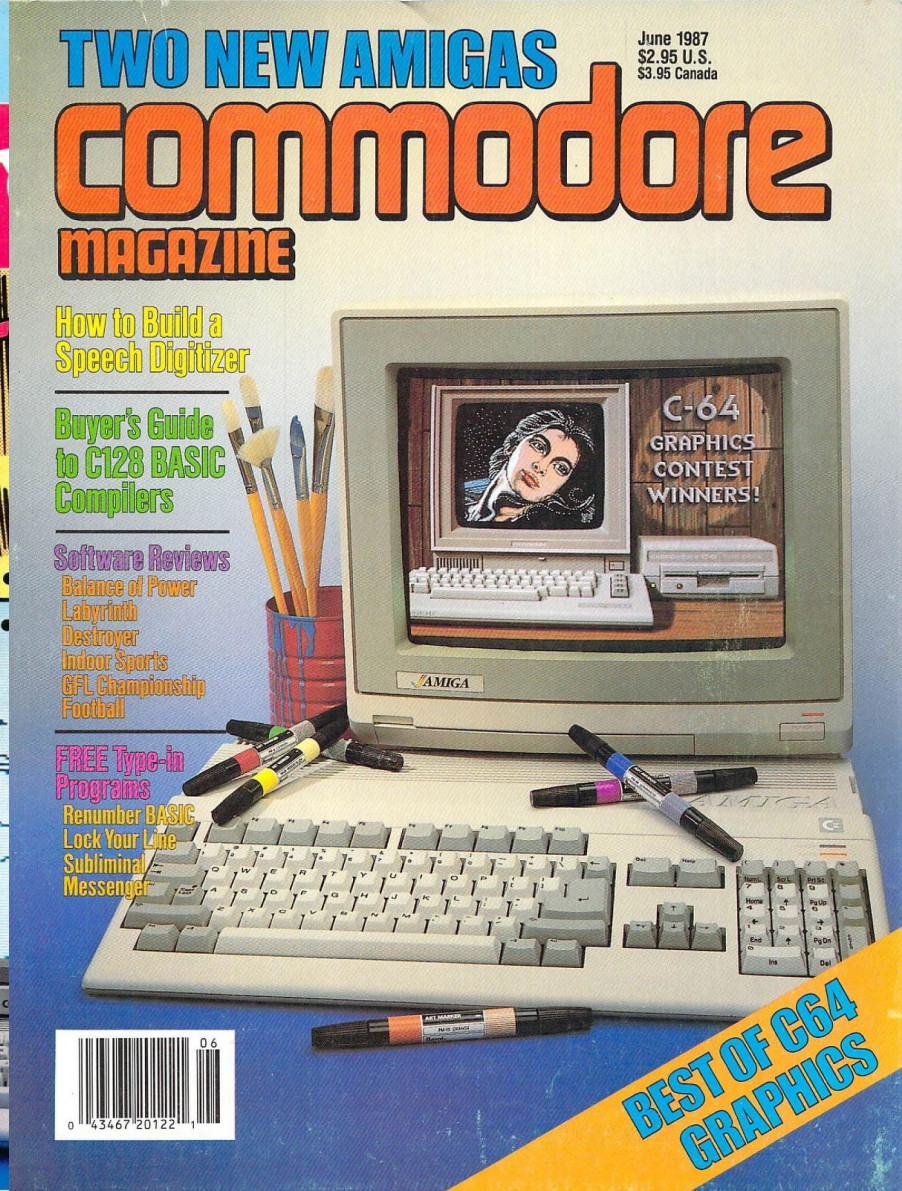
how we think about, read, write, and learn to code

Dawid Zalewski


ME BRIDGE
1
E 1/2 F
P1
HI
0 PLANES
0 4


FREE AMIGA BUYER'S GUIDE
COMMOD
MAGAZINE
C64/128 Banking at Home
Commodore Computers in Hollywood
Software Reviews
The Big Blue Reader
Defender of the Crown
Jet and Scenery Disks
Free Type-in Programs
Border Patrol
Letter Right!
Supersweep 128
AmigaLife
03
0 43467 20122 1


Joystick/Mouse Troubleshooting
Commodor
MAGAZIN
The Magazine for Commodore and Commodore Amiga Users
IT'S INFOCOMICS! A COMPUTERIZED COMIC BOOK!!
KER-
Software Reviews
AMIGA THREE STOOGES
64 & 128 4TH & INCHES
Type-in Programs
for the 64 and 128
Bikegear
Mandelbrot Graphics
Beginner's Guide to Debugging
...AND MORE
09
0 43467 20122 1


TWO NEW AMIGAS
June 1987
$2.95 U.S.
$3.95 Canada
COMMODORE
MAGAZINE
How to Build a Speech Digitizer
Buyer's Guide to C128 BASIC Compilers
Software Reviews
Balance of Power
Labyrinth
Destroyer
Indoor Sports
GFL Championship Football
FREE Type-in Programs
Renumber BASIC
Lock Your Line
Subliminal Messenger
C-64 GRAPHICS CONTEST WINNERS!
AMIGA
BEST OF C64 GRAPHICS
06
0 43467 20122 1

by Fred Karg

The Challenge

for the Commodore 64

ART BAXTER

The Challenge is a one-player strategy game designed for the Commodore 64. The game is reminiscent of Rubik's Cube, although it's much simpler to solve. The Challenge offers two options of play with three difficulty levels per option.

In the Knight's Challenge Mode, the player is shown chess pieces arranged in columns and rows. The pieces are then shuffled, and the player must put them back in the original order with the fewest number of moves. Pressing F7 during play will show what the completed puzzle is supposed to look like.

Color Challenge Mode uses colored squares instead of chess pieces. A joystick plugged into port 2 is required.

Use the joystick to move the arrow on the screen to point to any row or column of the game grid. If the arrow points to a row, then pressing the fire button will move each piece in that row one position to the left. The piece on the far left will wrap around to the right. Columns can be manipulated in a similar fashion. Each press of the fire button counts as one move. When the puzzle is solved, the computer will play the choral movement from Beethoven's Ninth Symphony.

You may change the screen and background border colors at any time by pressing F1 and F3, respectively. Chess piece colors can also be changed by pressing F5 and F7. This option is only available before the first game is played.

Like the game of chess, the game becomes most fascinating when the player tries to anticipate patterns several moves in the future and is thus able to combine many pieces in just one move. C

Before typing this program, read "How to Enter Programs" and "How to Use the Magazine Entry Program." The BASIC programs in this magazine are available on disk from Loadstar, P.O. Box 30008, Shreveport, LA 71130-0007, 1-800-831-2694.

The Challenge

```
20 POKE 53281,0:POKE 53280,12
   :PL=0'DUQE
30 PRINT"[CLEAR,DOWN2,RIGHT12,RVS,
   YELLOW] THE CHALLENGE "'BASG
40 PRINT"[DOWN4] LOADING MACHINE
   LANGUAGE..."'BAXI
50 FOR N=1 TO 30:READ A:NEXT'EHSE
120 POKE 646,PEEK(53281)-1
    :POKE 49383,0'EUPD
130 POKE 49374,1:POKE 53269,0'CPCC
140 PRINT"[CLEAR,DOWN3,RIGHT2,RVS] 1
    [RVOFF] KNIGHT'S CHALLENGE"
    :PRINT"[DOWN2,RIGHT2,RVS] 2
    [RVOFF] COLOR CHALLENGE"'CBPN
150 PRINT"[DOWN4,RIGHT2,RVS]
     PLEASE CHOOSE 1 OR 2 ";
    :POKE 198,0'CHEI
160 GET KS:IF KS=""THEN 160'EIBF
```

number of moves. Pressing F7 during play will show what the completed puzzle is supposed to look like.

Color Challenge Mode uses colored squares instead of chess pieces. A joystick plugged into port 2 is required.

Use the joystick to move the arrow on the screen to point to any row or column of the game grid. If the arrow points to a row, then pressing the fire button will move each piece in that row one position to the left. The piece on the far left will wrap around to the right. Columns can be manipulated in a similar fashion. Each press of the fire button counts as one move. When the puzzle is solved, the computer will play the choral movement from Beethoven's Ninth Symphony.

You may change the screen and background border colors at any time by pressing F1 and F3, respectively. Chess piece colors can also be changed by pressing F5 and F7. This option is only available before the first game is played.

Like the game of chess, the game becomes most fascinating when the player tries to anticipate patterns several moves in the future and is thus able to combine many pieces in just one move.

Before typing this program, read "How to Enter Programs" and "How to Use the Magazine Entry Program." The BASIC programs in this magazine are available on disk from Loadstar, P.O. Box 30008, Shreveport, LA 71130-0007, 1-800-831-2694.

The Challenge

```
20 POKE 53281,0:POKE 53280,12
   :PL=0'DUQE
30 PRINT"[CLEAR,DOWN2,RIGHT12,RVS,
   YELLOW] THE CHALLENGE "'BASG
40 PRINT"[DOWN4] LOADING MACHINE
   LANGUAGE..."'BAXI
50 FOR N=1 TO 30:READ A:NEXT'FHSF
60 FOR N=16128 TO 17412:READ A
   :POKE N,A:NEXT'GSLJ
70 S=54272:V=53248:POKE 53276,255
   :POKE V+27,255:POKE V+16,0
   :POKE V+28,0'JSES
80 FOR N=S TO S+24:POKE N,0:NEXT
   :POKE S+5,144'IRRM
85 POKE S+6,251:POKE 49378,2
   :POKE 49380,1'EWCQ
90 FOR N=0 TO 6:X=212-28*N
   :POKE 49153+N,X'IUWO
95 POKE 49161+N,X-21:POKE 53248+N*2,
   30+N*30'IYEU
100 POKE 53249+N*2,230:NEXT
    :POKE 53262,240:POKE 49152,240
    :POKE 49160,219'HRLH
110 POKE 53263,230:FOR N=0 TO 3
    :POKE 2040+N,255'GXPE
115 POKE V+39+N,1:NEXT:FOR N=4 TO 7
    :POKE 2040+N,254:POKE V+39+N,2
    :NEXT'NEFP
120 POKE 646,PEEK(53281)-1
    :POKE 49383,0'EUPD
130 POKE 49374,1:POKE 53269,0'CPCC
140 PRINT"[CLEAR,DOWN3,RIGHT2,RVS] 1
    [RVOFF] KNIGHT'S CHALLENGE"
    :PRINT"[DOWN2,RIGHT2,RVS] 2
    [RVOFF] COLOR CHALLENGE"'CBPN
150 PRINT"[DOWN4,RIGHT2,RVS]
     PLEASE CHOOSE 1 OR 2 ";
    :POKE 198,0'CHEI
160 GET K$:IF K$=""THEN 160'EIBF
170 IF K$<>"1"AND K$<>"2"THEN 120'HHBI
180 PRINT"[CLEAR,DOWN2,RIGHT2]
    PLEASE CHOOSE DIFFICULTY
    LEVEL"'BAIM
190 PRINT"[DOWN2,RIGHT2]
    PICK A NUMBER FROM [RVS] 1 [RVOFF]
     (EASY)":PRINT'CBXN
195 PRINT SPC(18)"TO [RVS] 3 [RVOFF]
     (HARD)"'CDSO
200 GET K1$:IF K1$=""THEN 200'EKTA
210 K=ASC(K1$)-48:IF K<1 OR K>3 THEN
    200'IQOF
220 IF K$="1"THEN GOSUB 400
    :POKE 829,PEEK(53281)'GRQF
230 IF K$="2"THEN GOSUB 600'EFSC
240 IF PL=0 THEN:SYS 16385:PL=1'FNVG
250 POKE 49377,1:IF PL=1 THEN POKE
    174,71:POKE 175,4'GXEJ
260 POKE 49396,3:PRINT"[CLEAR,DOWN2]
    "SPC(7)"LET THE CHALLENGE BEGIN!
    [DOWN]"'DKGM
```

```
270 PRINT SPC(8)"(USE JOYSTICK
    PORT2)"'CCXK
275 PRINT"[DOWN2,RIGHT6]
    YOUR CHALLENGE IS TO
    DUPLICATE"'BAYR
280 PRINT SPC(12)"THIS PATTERN"'CDAJ
285 PRINT"[DOWN2,RIGHT6]PRESS [RVS]
     F7 [RVOFF] DURING GAME TO
    SEE"'BAMS
290 PRINT SPC(12)"[DOWN]
    THIS SOLVED PUZZLE"'CDBM
295 PRINT"[DOWN2,RIGHT6,RVS]
    PRESS SPACE BAR TO BEGIN":C=0'CDRT
300 C=(1-C):POKE 53269,C*255
    :FOR N=1 TO 100'HWXG
305 IF PEEK(197)<>60 THEN NEXT N
    :GOTO 300'HMNI
310 N=100:NEXT'CFBA
320 IF PEEK(197)<>64 THEN 320'FKWE
330 PRINT"[CLEAR]SHUFFLING PATTERN"
    :POKE 53269,0:POKE 49383,1'DQSK
335 FOR N=1 TO 200:NEXT N'EHQI
340 POKE 53281,PEEK(53281)+1
    :PRINT"[CLEAR]":POKE 53281,
    PEEK(53281)-1:POKE 1095,31'INNO
350 FOR N=1499 TO 1503:POKE N,48
    :NEXT'FPFI
360 POKE 646,PEEK(53281)+1'DMIH
370 POKE 49374,0:PRINT"[HOME,DOWN9]
    "SPC(34)"[RVS] MOVES"'DLPK
375 POKE 53269,255:POKE 49377,0
    :POKE 49379,0'DAXP
380 IF PEEK(49379)=1 THEN POKE 49377,1
    :GOTO 700'GTWM
390 GOTO 380'BDKH
400 IF PL=1 THEN POKE 53281,PEEK(829)
    :GOTO 550'GSMF
410 POKE 53269,255'BJDB
420 POKE 646,PEEK(53281)+1'DMIE
430 PRINT"[CLEAR]"SPC(10)"[DOWN2,RVS]
     TO CHANGE COLORS "'CDHI
435 PRINT SPC(8)"[DOWN,RVS]
     PRESS FUNCTION KEYS"'CCXN
440 PRINT"[DOWN2,RIGHT2,RVS] F1
    [RVOFF] CHANGES SCREEN COLOR"'BACJ
445 PRINT"[DOWN,RIGHT2,RVS] F3 [RVOFF]
     CHANGES BACKGROUND COLOR"'BASQ
450 PRINT"[DOWN,RIGHT2,RVS] F5 [RVOFF]
     CHANGES CHESSPIECE 1 COLOR"'BAWM
460 PRINT"[DOWN,RIGHT2,RVS] F7 [RVOFF]
     CHANGES CHESSPIECE 2 COLOR"'BAAN
470 PRINT"[DOWN,RIGHT5,RVS]
     PRESS SPACE BAR TO START"'BAAN
480 GET K2$:IF K2$=""THEN 480'EKGK
490 IF K2$=" "THEN PRINT"[CLEAR]"
    :GOTO 550'FHVM
500 IF K2$="[F1]"THEN X=PEEK(53281)+1
    :GOSUB 680:POKE 53281,X
    :GOTO 420'JDCL
510 IF K2$="[F3]"THEN X=PEEK(53280)+1
    :GOSUB 680:POKE 53280,X'IYSK
520 IF K2$="[F5]"THEN GOSUB 640'EGBF
530 IF K2$="[F7]"THEN GOSUB 660'EGEG
540 GOTO 480'BDLE
550 PRINT"[DOWN]  PLEASE WAIT..."
    :FOR R=0 TO 6:CO=PEEK(49378):SH=0
    :FOR C=0 TO 7'KWYT
560 IF C>3 THEN CO=PEEK(49380)
    :IF K=1 THEN SH=1'JRHO
570 IF K=2 THEN SH=INT(C/2)+2*(C>3)
    'JNNO
580 IF K=3 THEN SH=C+4*(C>3)'HKSN
590 POKE 49170+C+R*8,CO
    :POKE 49226+C+R*8,255-SH:NEXT:NEXT
    :RETURN'MEAW
600 PRINT"[DOWN]  PLEASE WAIT..."
    :FOR R=0 TO 6:CO=0:FOR C=0 TO
    7'IMLL
605 IF K=1 THEN IF C>3 THEN CO=1'HHUL
610 IF K=2 THEN CO=INT(C/2)'GIEG
620 IF K=3 THEN CO=C'EFHF
630 POKE 49170+C+R*8,CO+1
    :POKE 49226+C+R*8,251:NEXT
    :NEXT'LCBP
635 FOR N=16064 TO 16127:POKE N,255
    :NEXT:POKE 53281,0:RETURN'HCSR
640 FOR N=0 TO 3:X=PEEK(V+39+N)+1
    :IF X>255 THEN X=0'MTEQ
650 POKE V+39+N,X:NEXT:POKE 49380,X
    :RETURN'GQUM
660 FOR N=4 TO 7:X=PEEK(V+39+N)+1
    :IF X>255 THEN X=0'MTMS
670 POKE V+39+N,X:NEXT:POKE 49378,X
    :RETURN'GQCO
680 IF X>253 THEN X=0'EGPM
690 RETURN'BAQJ
700 POKE S+24,15:FOR J=0 TO 1:RESTORE
    :FOR J1=1 TO 15:READ HF,LF
    :POKE 53269,LF'LHNO
710 POKE S+1,HF:POKE S,LF
    :POKE S+4,33-J*16:FOR N=1 TO 80
    :NEXT:POKE S+4,32-J*16'PJAT
720 FOR N=1 TO 100:NEXT:NEXT
    :FOR N=1 TO 100:NEXT:NEXT
    :POKE S+24,0:GOTO 120'NAVP
730 DATA 21,31,21,31,22,96,25,30,25,
    30,22,96,21,31,18,209,16,195,16,
    195'BMBQ
740 DATA 18,209,21,31,21,31,18,209,18,
    209'BHSL
750 DATA 0,24,0,0,60,0,0,122,0,0,118,
    0,0,60,0,0'BNKN
760 DATA 24,0,0,60,0,0,126,0,0,0,0,0,
    60,0,0,126'BNNO
770 DATA 0,0,0,0,0,60,0,0,60,0,0,60,0,
    0,60,0'BKNO
780 DATA 0,60,0,0,126,0,0,255,0,0,0,0,
    1,255,128,0'BPYQ
790 DATA 0,0,0,1,126,128,1,126,128,1,
    255,128,1,255,128,0'BWMT
800 DATA 0,0,0,126,0,0,126,0,0,126,0,
    0,126,0,0,126'BQKK
810 DATA 0,0,126,0,0,126,0,1,255,128,
    0,0,0,1,255,128'BSCL
820 DATA 3,255,192,7,255,224,0,0,0,7,
    255,224,7,255,224,167'BYPN
830 DATA 0,3,0,0,15,0,0,127,0,1,254,
```

Continued on page 86

Continued from page 84

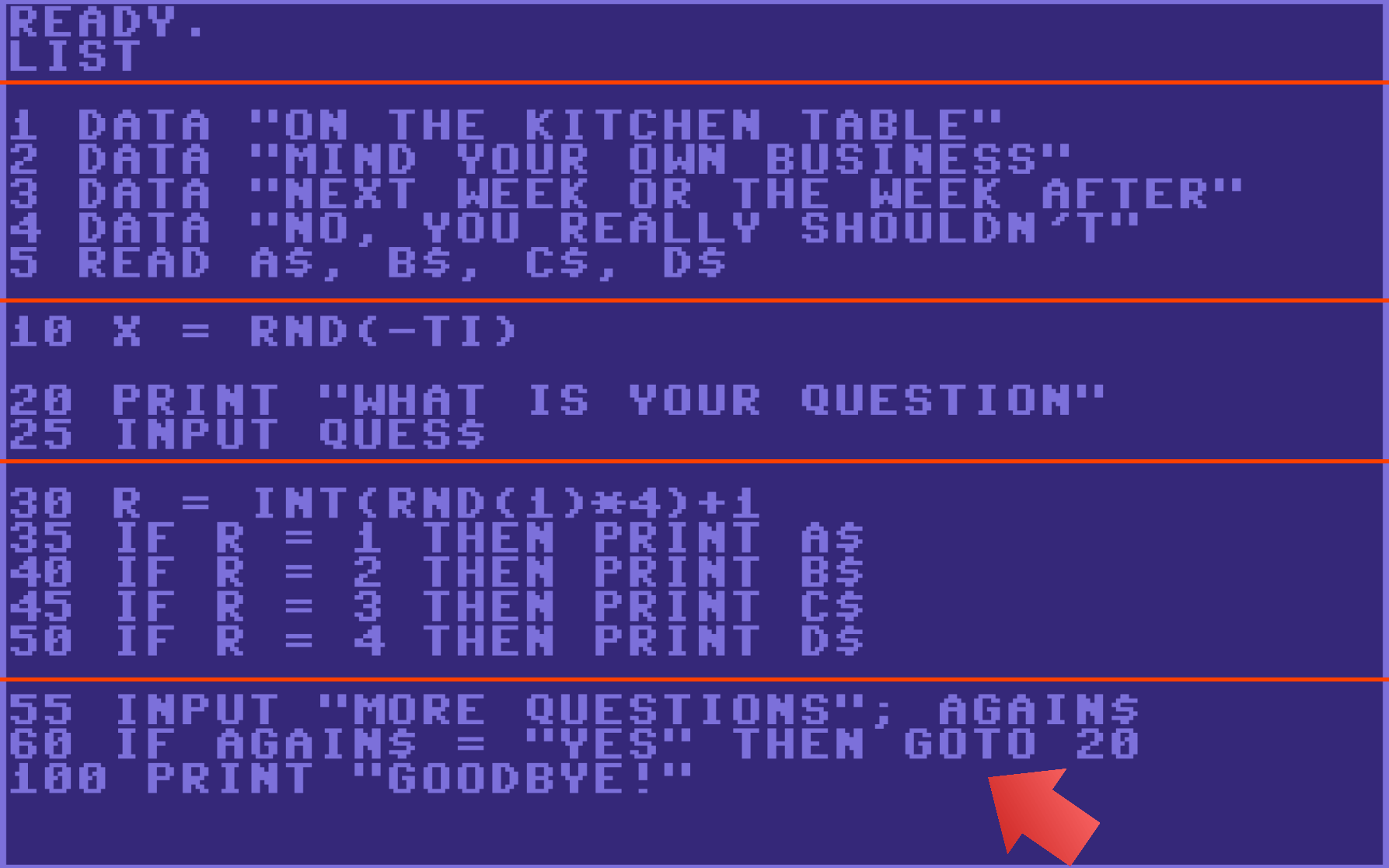
```
    128,3,157,192,15'BSMN
840 DATA 255,96,25,254,224,6,15,176,0,
    31,112,0,62,224,0,255'BAAQ
850 DATA 160,1,255,64,1,254,192,1,255,
    128,0,255,128,0,127,128'BCDR
860 DATA 0,0,0,7,255,224,0,0,0,1,255,
    128,7,255,224,0'BSOQ
870 DATA 0,60,0,0,24,0,0,126,0,0,24,0,
    0,60,0,1'BMNQ
880 DATA 255,128,0,255,0,0,0,0,0,60,0,
    0,126,0,0,36'BQYS
890 DATA 0,0,60,0,0,60,0,0,126,0,0,
    126,0,0,126,0'BOOS
900 DATA 0,255,0,0,189,0,0,102,0,1,
    255,128,7,255,224,0'BULM
910 DATA 0,169,255,141,14,212,141,15,
    212,169,129,141,18,212,141,
    141'BIJP
920 DATA 2,120,169,127,141,13,220,169,
    1,141,26,208,169,8,141,247'BFYQ
930 DATA 192,173,0,192,141,18,208,169,
    27,141,17,208,169,124,141,20'BHFR
940 DATA 3,169,64,141,21,3,169,147,32,
    210,255,169,0,141,227,192'BELR
950 DATA 141,225,192,168,169,48,141,
    218,5,141,219,5,141,220,5,141'BGDT
960 DATA 221,5,141,222,5,141,223,5,
    169,3,141,244,192,169,71,133'BEAT
970 DATA 174,169,4,133,175,169,31,141,
    243,192,145,174,169,18,133,
    253'BJYV
980 DATA 169,74,133,251,169,192,133,
    254,133,252,88,96,173,25,208,
    141'BJAX
990 DATA 25,208,201,7,208,3,76,55,65,
    206,247,192,16,100,169,6'BCSW
1000 DATA 141,247,192,165,162,208,91,
    169,245,133,162,173,225,192,208,
    3'BKNF
1010 DATA 32,64,65,165,197,201,64,240,
    57,201,4,208,28,238,33,208'BETE
1020 DATA 173,228,192,205,33,208,208,
    3,238,33,208,173,226,192,205,
    33'BIJG
1030 DATA 208,208,47,238,33,208,76,
    242,64,201,5,208,6,238,32,
    208'BEUG
1040 DATA 76,242,64,201,3,208,27,173,
    222,192,208,6,32,164,67,32'BDOH
1050 DATA 120,67,173,231,192,240,11,
    32,216,66,32,102,66,169,0,
    141'BFQJ
1060 DATA 231,192,174,247,192,160,0,
    189,8,192,153,1,208,200,200,
    192'BHBK
1070 DATA 16,208,247,160,0,177,253,
    153,39,208,177,251,153,248,7,
    200'BHNL
1080 DATA 192,8,208,241,165,251,24,
    105,8,133,251,165,253,24,105,
    8'BFDM
1090 DATA 133,253,201,74,208,8,169,18,
    133,253,169,74,133,251,189,0'BGSN
1100 DATA 192,141,18,208,138,240,6,
    104,168,104,170,104,64,76,49,
    234'BHJF
1110 DATA 173,0,220,160,0,162,0,74,
    176,1,136,74,176,1,200,74'BARF
1120 DATA 176,1,202,74,176,1,232,74,
    142,242,192,140,241,192,144,
    125'BHVH
1130 DATA 173,242,192,240,22,201,1,
    240,9,238,244,192,238,244,192,
    76'BHII
1140 DATA 148,65,206,244,192,206,244,
    192,76,148,65,173,241,192,208,
    1'BIAJ
1150 DATA 96,201,1,240,9,206,244,192,
    206,244,192,76,148,65,238,
    244'BGVK
1160 DATA 192,238,244,192,173,244,192,
    201,33,240,13,201,1,208,15,
    238'BIUL
1170 DATA 244,192,238,244,192,76,174,
    65,206,244,192,206,244,192,173,
    244'BLAN
1180 DATA 192,201,17,144,8,169,30,141,
    243,192,76,194,65,169,31,141'BGRN
1190 DATA 243,192,160,0,169,32,145,
    174,172,244,192,185,193,67,133,
    174'BJXO
1200 DATA 185,194,67,133,175,160,0,
    173,243,192,145,174,96,32,86,
    67'BGLG
1210 DATA 172,244,192,185,226,67,133,
    176,105,56,133,178,185,227,67,
    133'BKDI
1220 DATA 177,133,179,173,244,192,201,
    16,176,43,160,0,177,176,141,
    239'BJUI
1230 DATA 192,177,178,141,237,192,200,
    177,176,136,145,176,200,177,178,
    136'BNDK
1240 DATA 145,178,200,192,7,208,239,
    173,239,192,145,176,173,237,192,
    145'BLDL
1250 DATA 178,32,249,66,96,160,0,177,
    176,141,239,192,177,178,141,
    237'BISL
1260 DATA 192,152,24,105,8,168,177,
    176,141,238,192,177,178,141,236,
    192'BKDN
1270 DATA 152,56,233,8,168,173,238,
    192,145,176,173,236,192,145,178,
    152'BKGO
1280 DATA 24,105,8,168,192,48,208,217,
    173,239,192,145,176,173,237,
    192'BJGP
1290 DATA 145,178,32,249,66,96,165,
    174,141,230,192,165,175,141,229,
    192'BKEQ
1300 DATA 169,100,141,222,192,160,0,
    169,18,133,174,169,74,133,176,
    169'BJRH
1310 DATA 192,133,175,133,177,177,174,
    141,239,192,177,176,141,237,192,
    140'BNYJ
1320 DATA 232,192,32,206,66,177,174,
    141,238,192,177,176,141,236,192,
    173'BLWK
1330 DATA 239,192,145,174,173,237,192,
    145,176,172,232,192,173,238,192,
    145'BNEL
1340 DATA 174,173,236,192,145,176,200,
    192,56,208,202,238,222,192,173,
    222'BMGM
1350 DATA 192,208,178,173,229,192,133,
    175,173,230,192,133,174,96,173,
    27'BLCN
1360 DATA 212,74,168,201,55,176,247,
    96,32,65,67,169,18,133,251,
    169'BGNN
1370 DATA 192,133,252,169,1,133,253,
    169,194,133,254,160,0,177,251,
    145'BJLO
1380 DATA 253,200,192,112,208,247,32,
    44,67,173,225,192,208,66,32,
    65'BHMP
1390 DATA 67,169,18,133,251,169,192,
    133,252,169,1,133,253,169,194,
    133'BJEQ
1400 DATA 254,160,0,177,251,141,223,
    192,177,253,205,223,192,208,13,
    200'BKMJ
1410 DATA 192,112,208,239,169,1,141,
    227,192,141,32,208,173,239,192,
    133'BKEK
1420 DATA 251,173,238,192,133,252,173,
    237,192,133,253,173,236,192,133,
    254'BNLL
1430 DATA 96,165,251,141,239,192,165,
    252,141,238,192,165,253,141,237,
    192'BMUM
1440 DATA 165,254,141,236,192,96,32,
    65,67,160,5,169,218,133,251,
    169'BHGM
1450 DATA 5,133,252,177,251,24,105,1,
    145,251,201,58,208,190,169,
    48'BGGN
1460 DATA 145,251,136,208,238,32,44,
    67,32,65,67,169,18,133,251,
    169'BGEO
1470 DATA 192,133,252,169,1,133,253,
    169,194,133,254,160,0,177,253,
    141'BJJP
1480 DATA 229,192,177,251,145,253,173,
    229,192,145,251,200,192,112,208,
    237'BNJR
1490 DATA 32,44,67,96,173,225,192,208,
    12,169,1,141,225,192,173,32'BFPR
1500 DATA 208,141,224,192,96,169,0,
    141,225,192,173,224,192,141,32,
    208'BJHJ
1510 DATA 96,0,0,0,0,71,4,231,4,95,5,
    255,5,119,6,23,7'BQRH
1520 DATA 143,7,220,7,216,7,212,7,209,
    7,205,7,201,7,197,7'BWRJ
1530 DATA 194,7,0,0,0,0,18,192,26,192,
    34,192,42,192,50,192,58'BYWK
1540 DATA 192,66,192,25,192,24,192,23,
    192,22,192,21,192,20,192,19'BFON
1550 DATA 192,18,192,255,0'BQJG
```

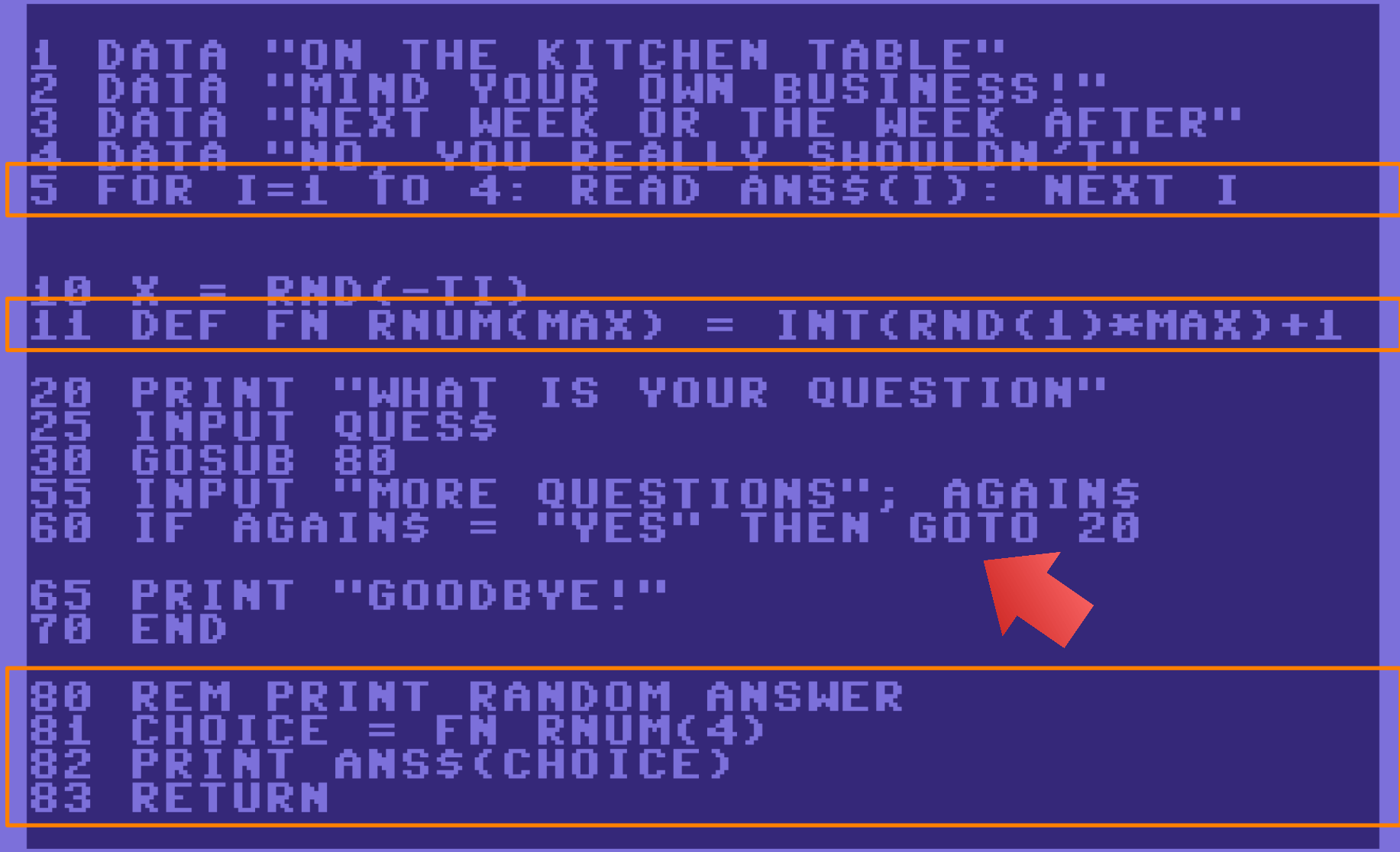
END

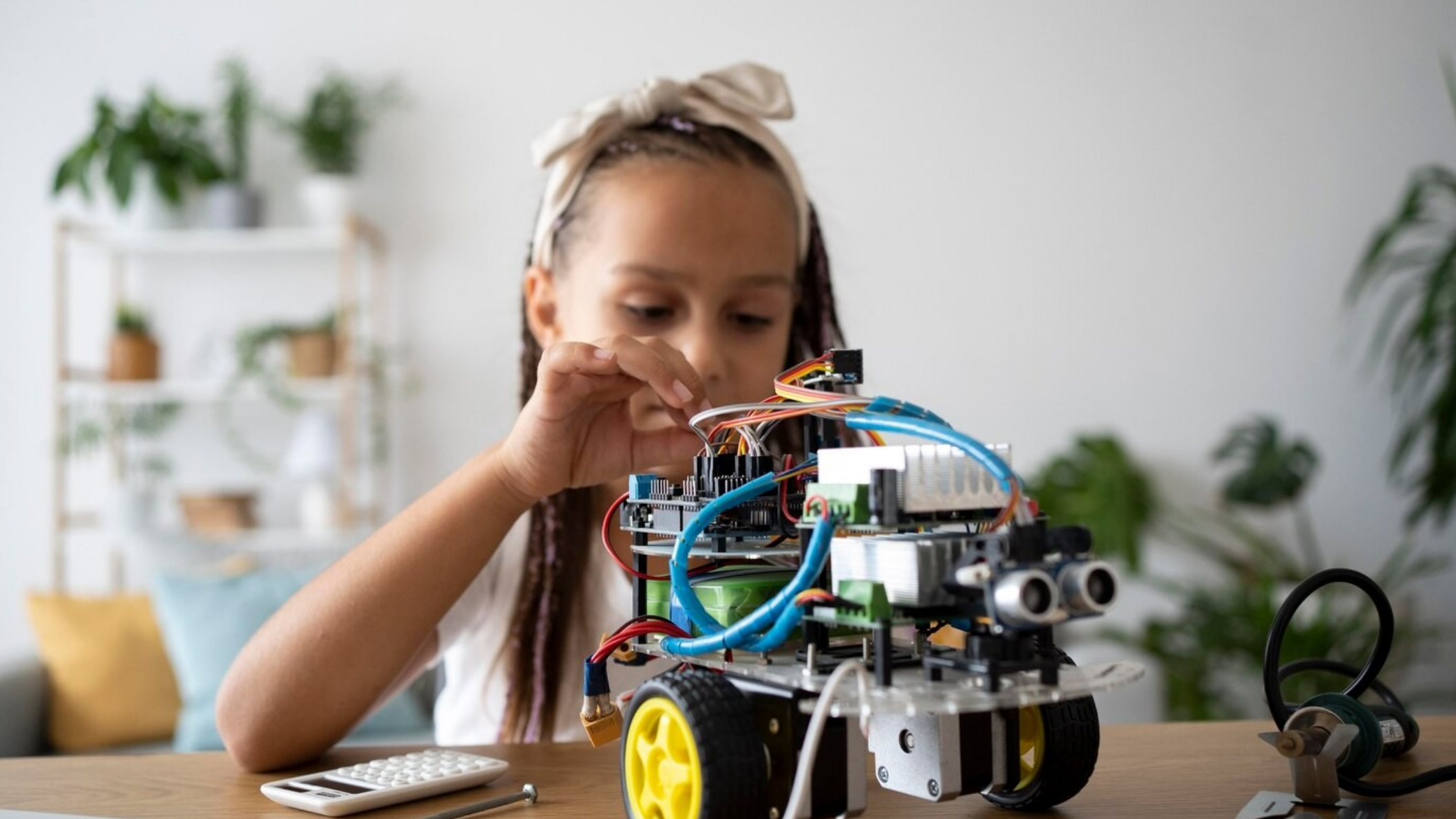
```
    **** COMMODORE 64 BASIC V2 ****

 64K RAM SYSTEM  38911 BASIC BYTES FREE

READY.
RUN

WHAT IS YOUR QUESTION
?
```

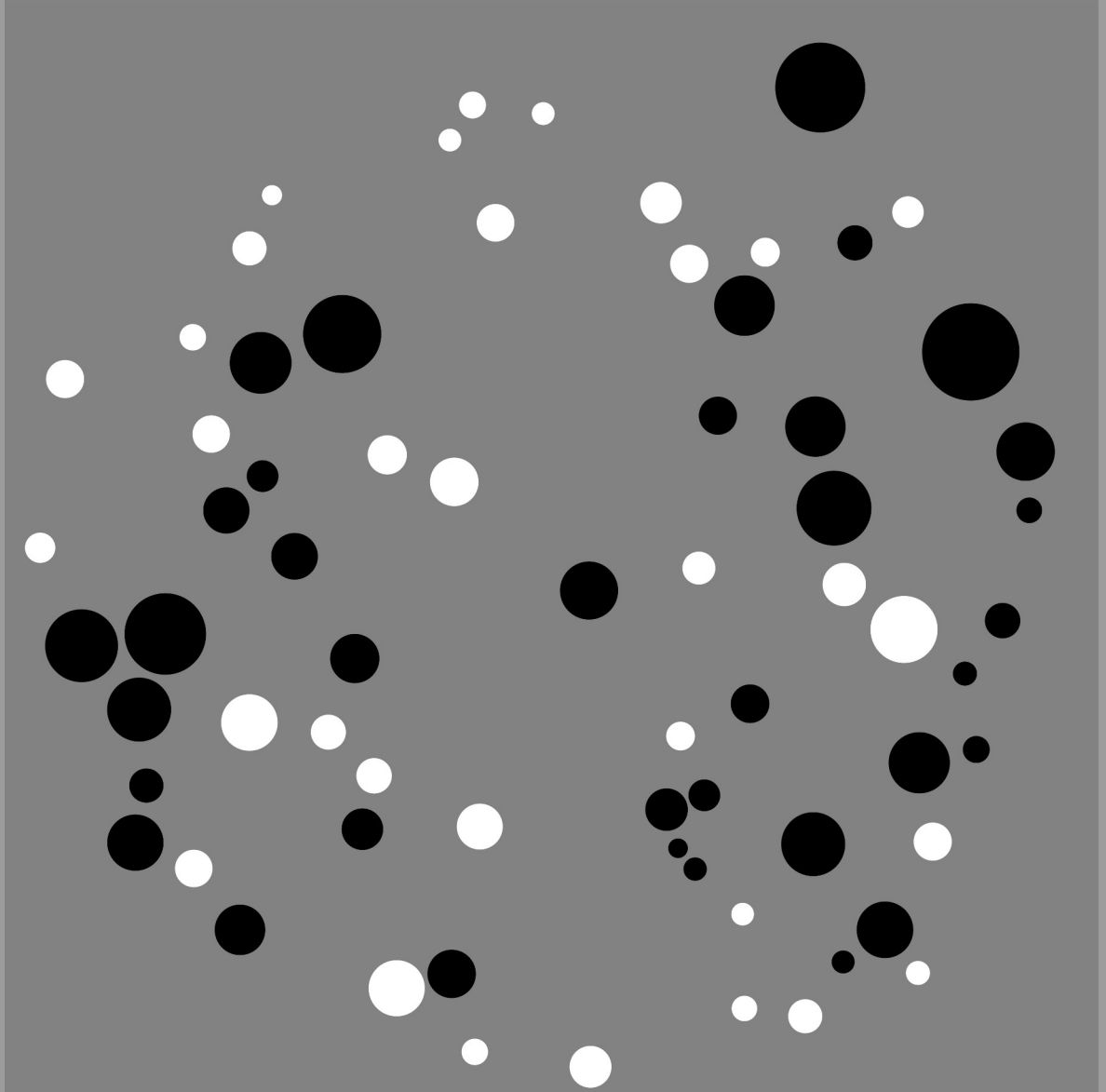
```
    **** COMMODORE 64 BASIC V2 ****

 64K RAM SYSTEM  38911 BASIC BYTES FREE

READY.
RUN

WHAT IS YOUR QUESTION
? WHEN WILL YOU GO TO BED
```

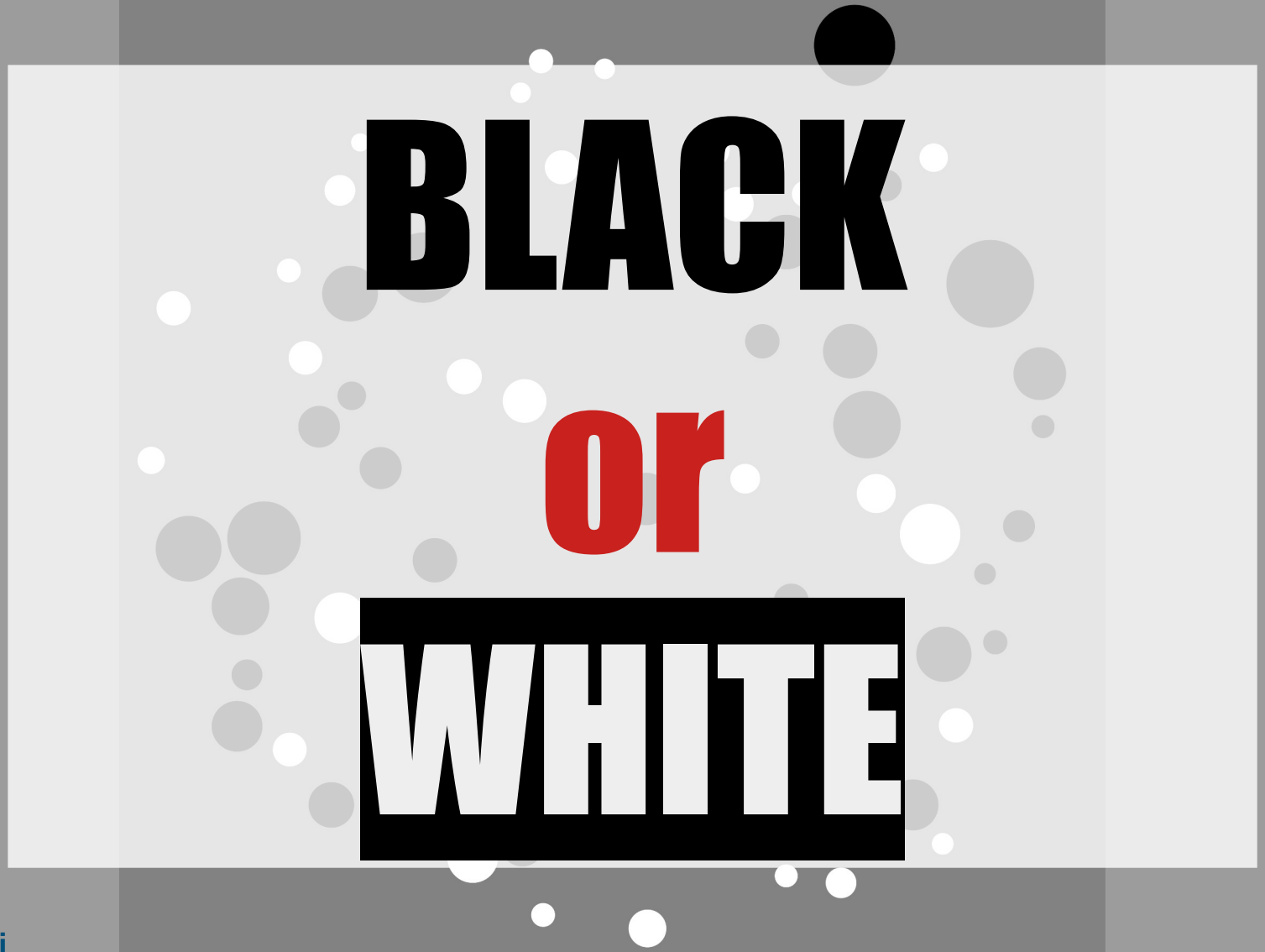
```
    **** COMMODORE 64 BASIC V2 ****

 64K RAM SYSTEM  38911 BASIC BYTES FREE

READY.
RUN

WHAT IS YOUR QUESTION
? WHEN WILL YOU GO TO BED

MIND YOUR OWN BUSINESS

MORE QUESTIONS?
```


READY.
LIST
1 DATA "ON THE KITCHEN TABLE"
2 DATA "MIND YOUR OWN BUSINESS"
3 DATA "NEXT WEEK OR THE WEEK AFTER"
4 DATA "NO, YOU REALLY SHOULDN'T"
5 READ A$, B$, C$, D$
10 X = RND(-TI)
20 PRINT "WHAT IS YOUR QUESTION"
25 INPUT QUES$
30 R = INT(RND(1)*4)+1
35 IF R = 1 THEN PRINT A$
40 IF R = 2 THEN PRINT B$
45 IF R = 3 THEN PRINT C$
50 IF R = 4 THEN PRINT D$
55 INPUT "MORE QUESTIONS"; AGAIN$
60 IF AGAIN$ = "YES" THEN GOTO 20
100 PRINT "GOODBYE!"

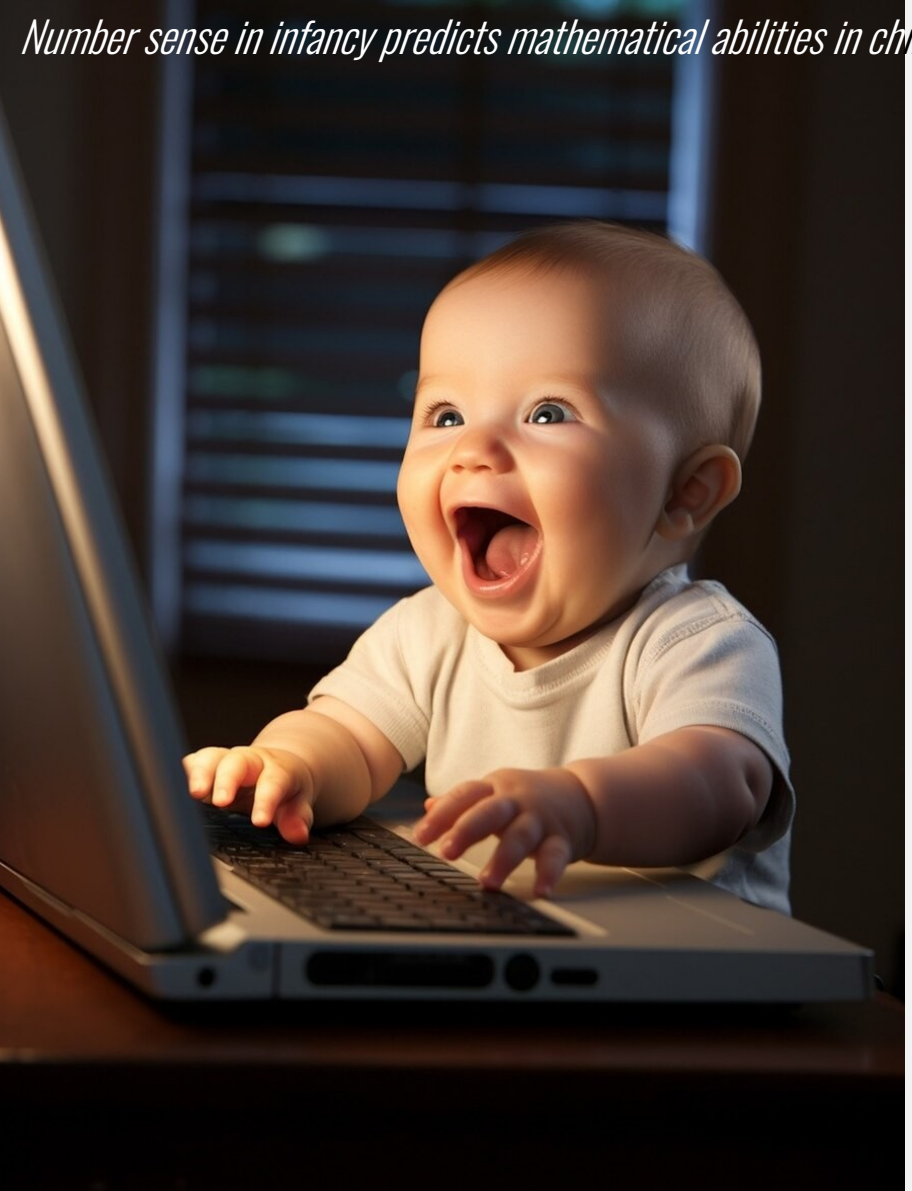

1 DATA "ON THE KITCHEN TABLE"
2 DATA "MIND YOUR OWN BUSINESS!"
3 DATA "NEXT WEEK OR THE WEEK AFTER"
4 DATA "NO, YOU REALLY SHOULDN'T"
5 FOR I=1 TO 4: READ ANS$(I): NEXT I
10 X = RND(-TI)
11 DEF FN RNUM(MAX) = INT(RND(1)*MAX)+1
20 PRINT "WHAT IS YOUR QUESTION"
25 INPUT QUES$
30 GOSUB 80
55 INPUT "MORE QUESTIONS"; AGAIN$
60 IF AGAIN$ = "YES" THEN GOTO 20
65 PRINT "GOODBYE!"
70 END
80 REM PRINT RANDOM ANSWER
81 CHOICE = FN RNUM(4)
82 PRINT ANS$(CHOICE)
83 RETURN

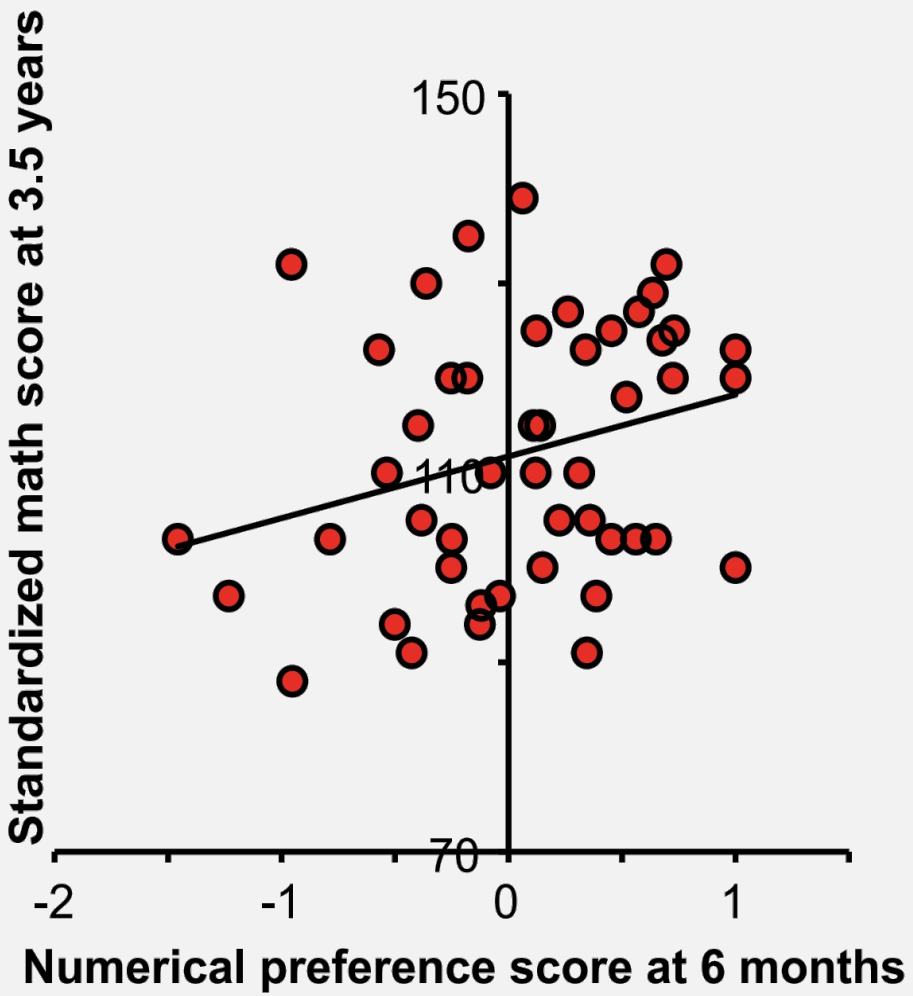




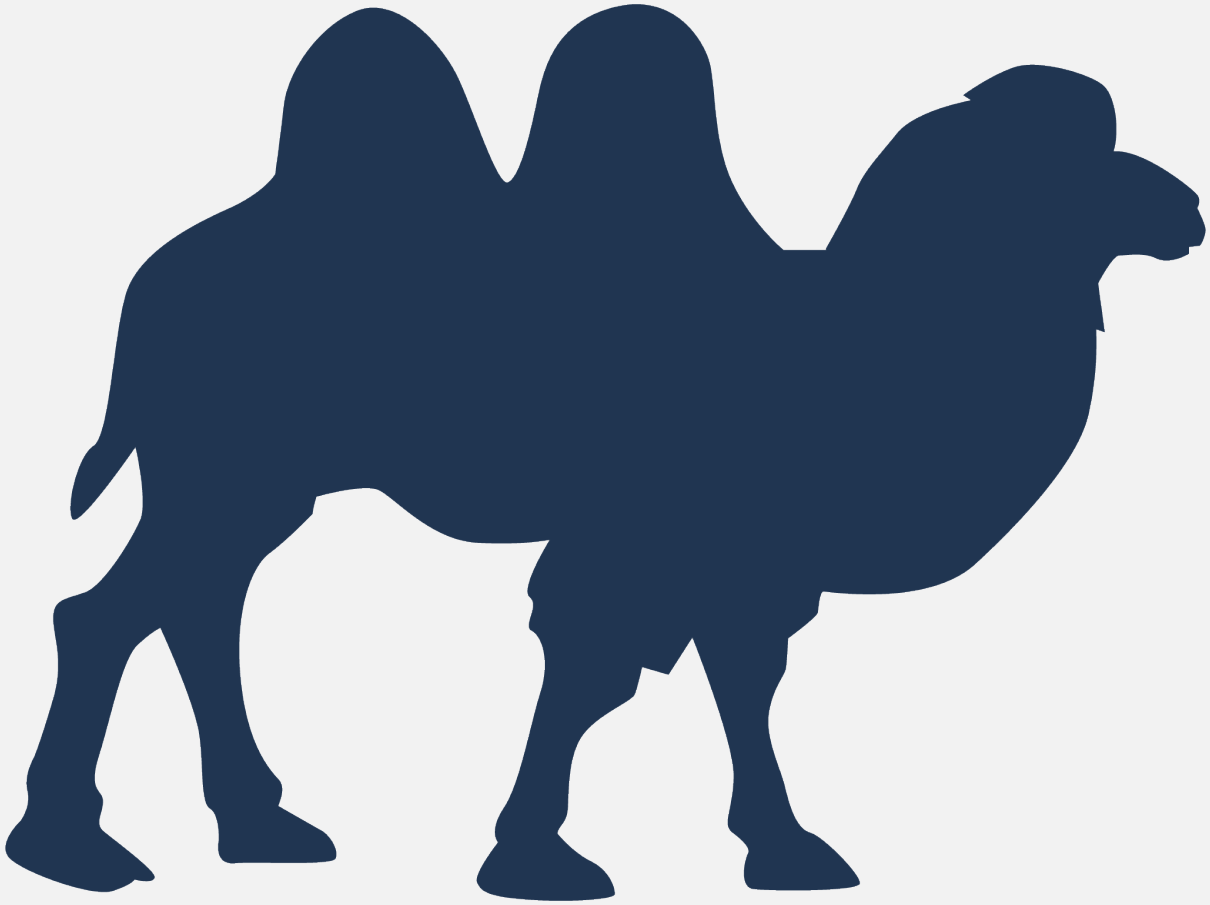

BLACK
or
WHITE

Number sense in infancy predicts mathematical abilities in childhood, Ariel Starr et al. 2023

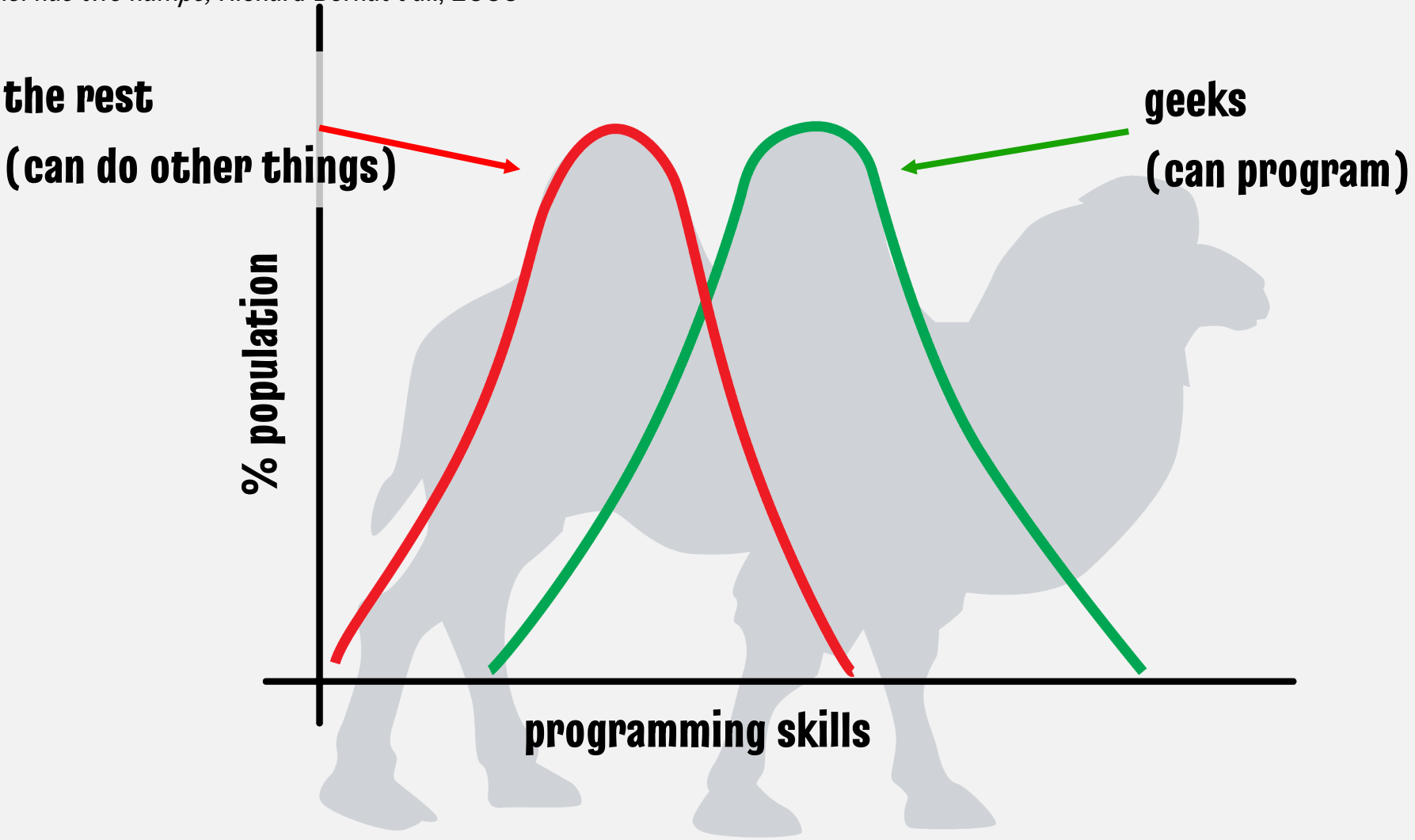




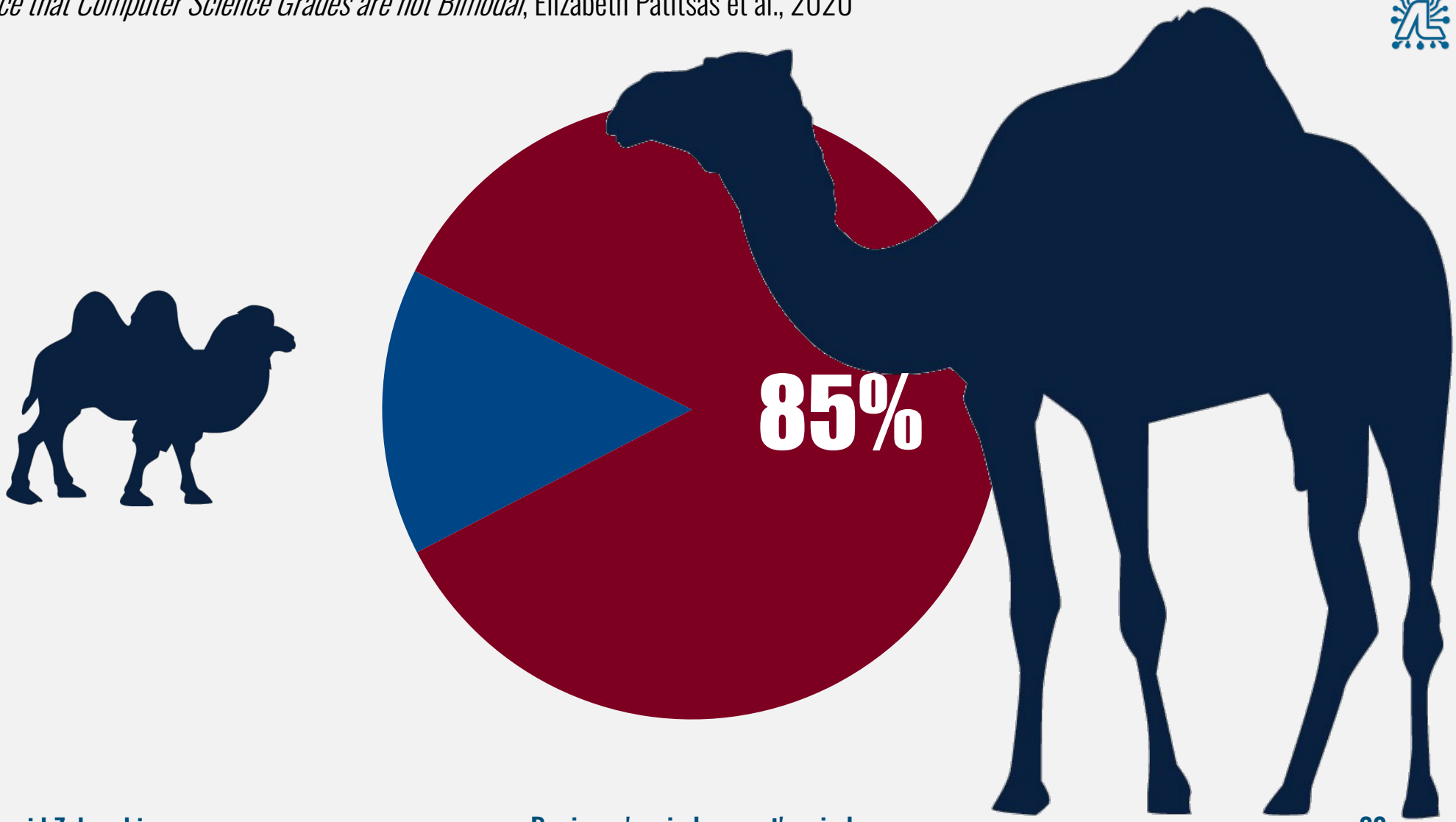
The camel has two humps, Richard Bornat t al., 2008



The camel has two humps, Richard Bornat t al., 2008



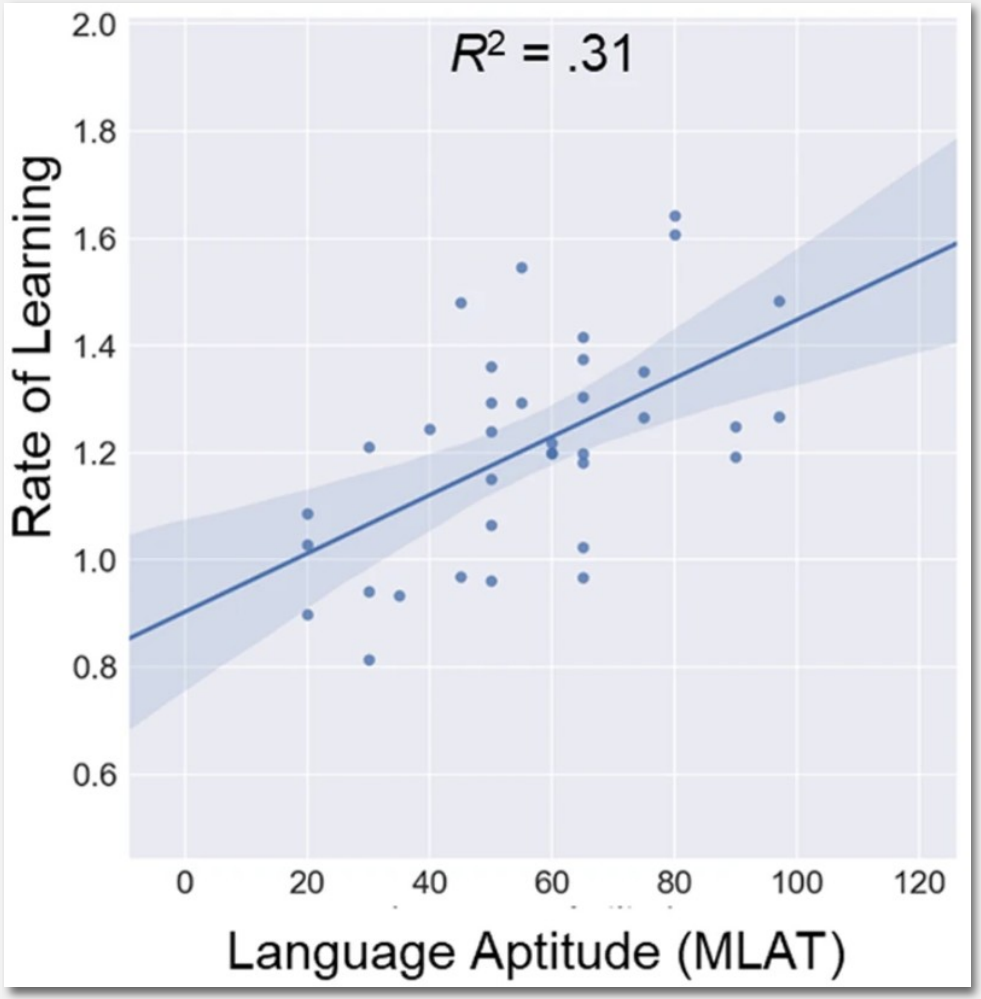
Evidence that Computer Science Grades are not Bimodal, Elizabeth Patitsas et al., 2020

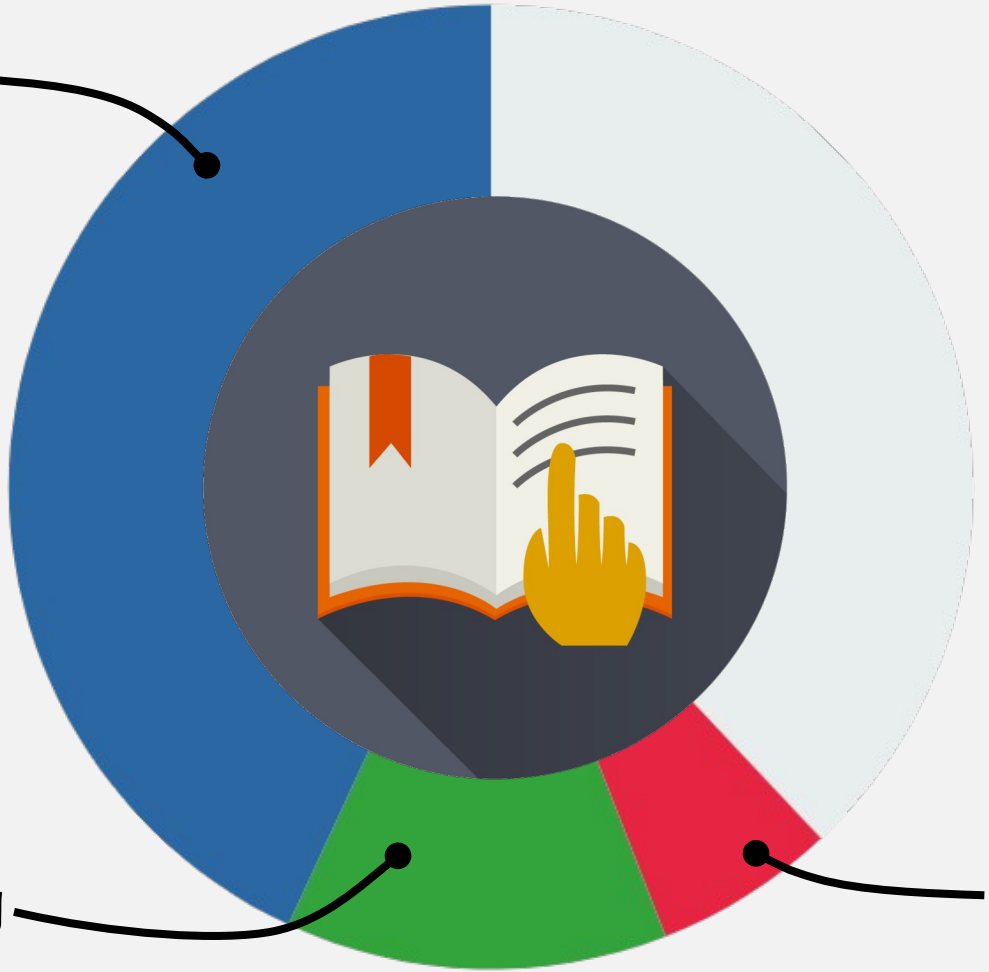


THERE IS NO GEEK GENE. Everybody can learn programming.

```
if ([[person age] isBetween:12 and:18]) {
    [ticket applyDiscount:20 in: percent];
}
```

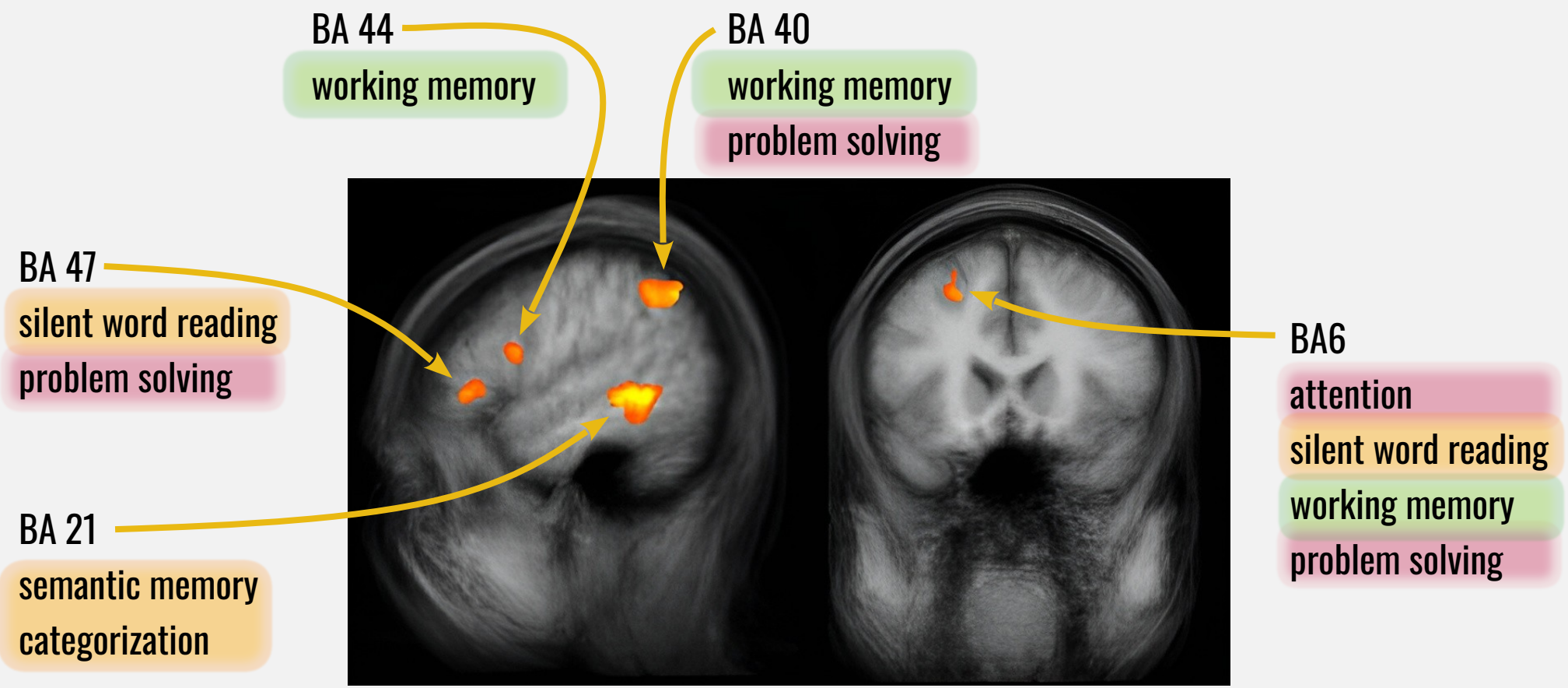
Relating Natural Language Aptitude to Individual Differences in Learning Programming Languages, Chantel S. Prat et al. 2020



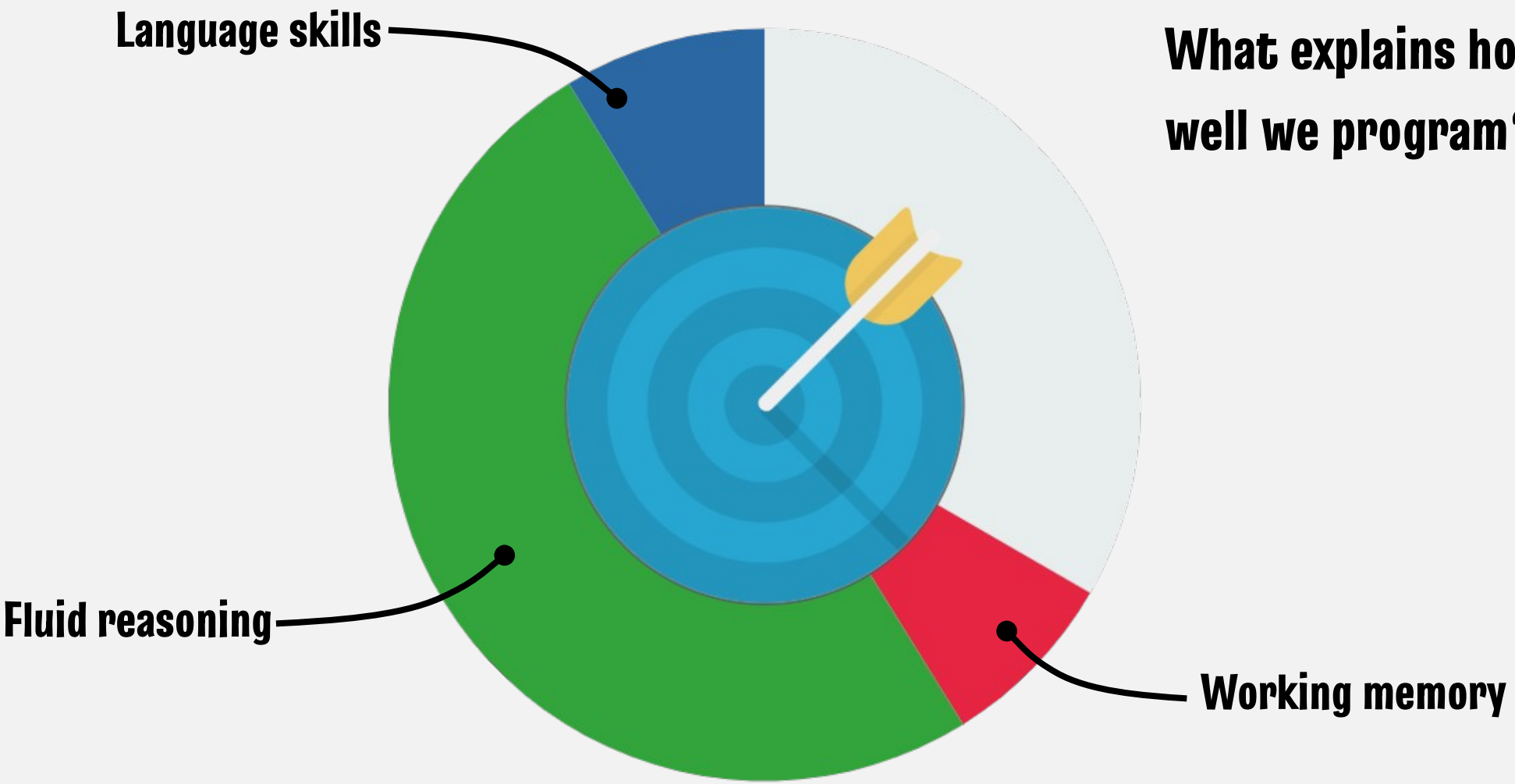


What explains how fast we learn to program?

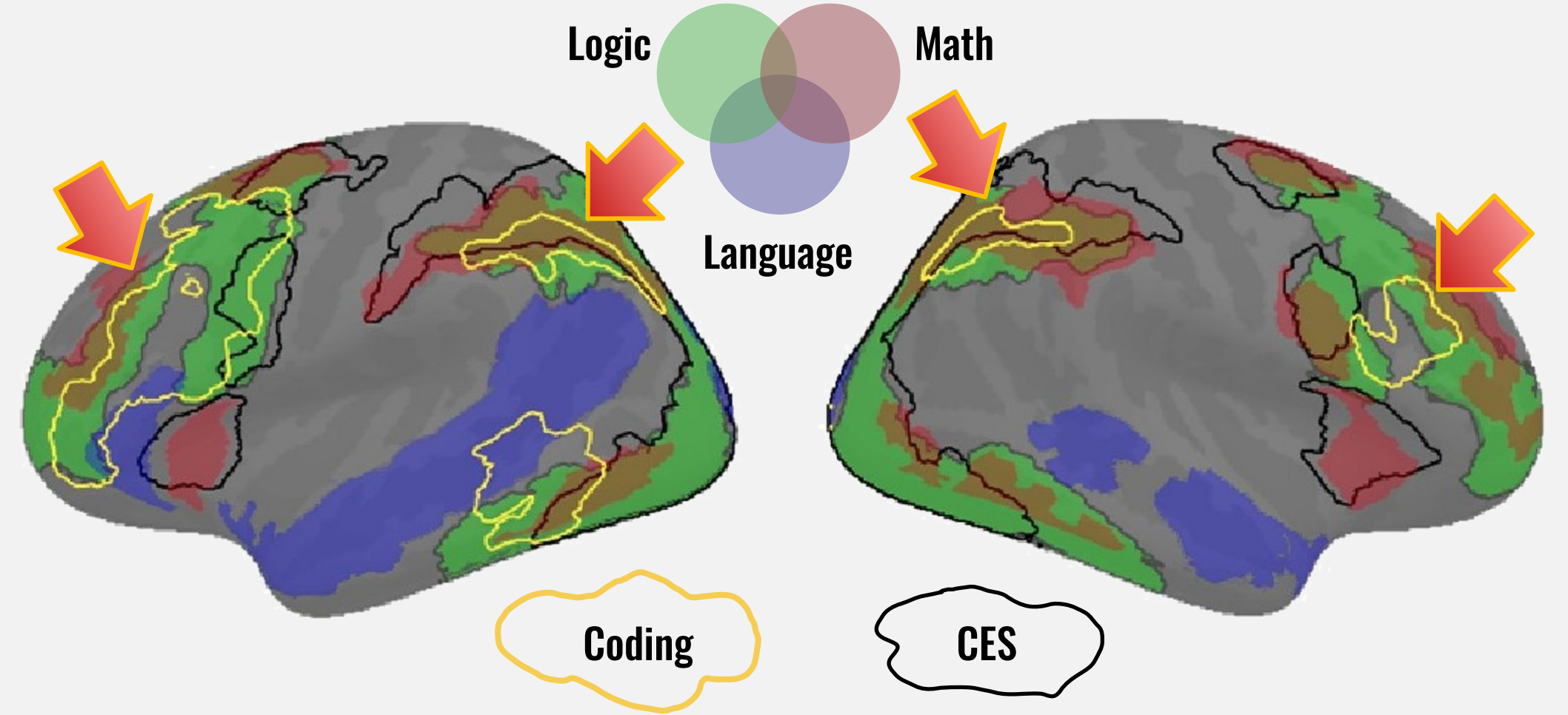
Understanding understanding source code with functional magnetic resonance imaging, Janet Siegmund et al. 2014



Relating Natural Language Aptitude to Individual Differences in Learning Programming Languages, Chantel S. Prat et al. 2020

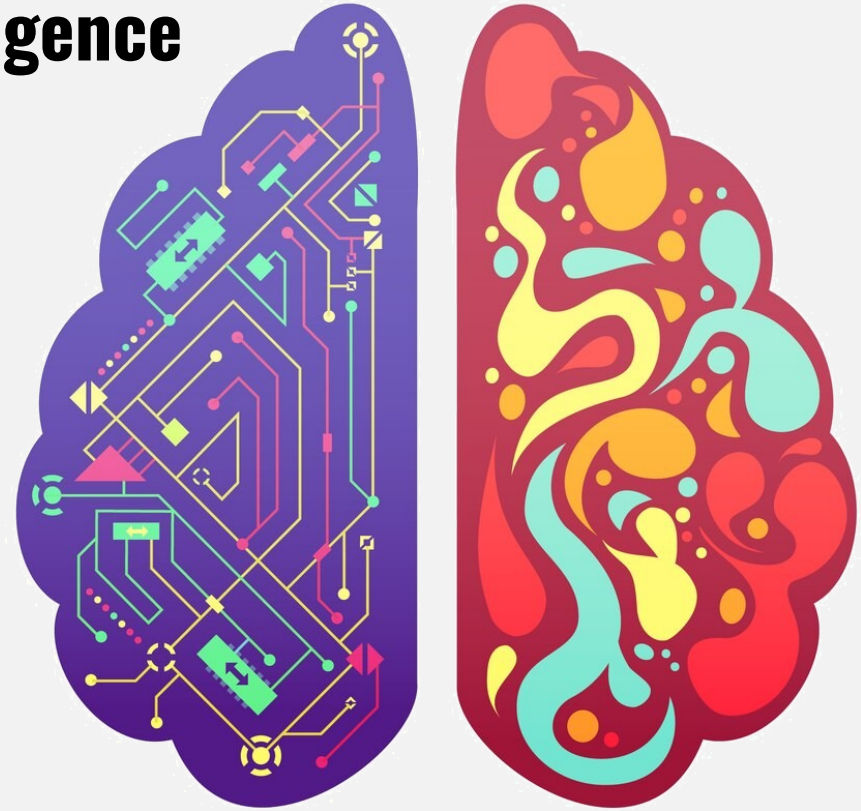


Computer code comprehension shares neural resources with formal logical inference in the fronto-parietal network,
Yun-Fei Liu et al. 2020



Crystallized Intelligence

- Procedures
- Schemes
- Knowledge



Fluid Intelligence

- Abstract reasoning
- Analogical ability
- Spatial reasoning
- Problem-solving

Language

+

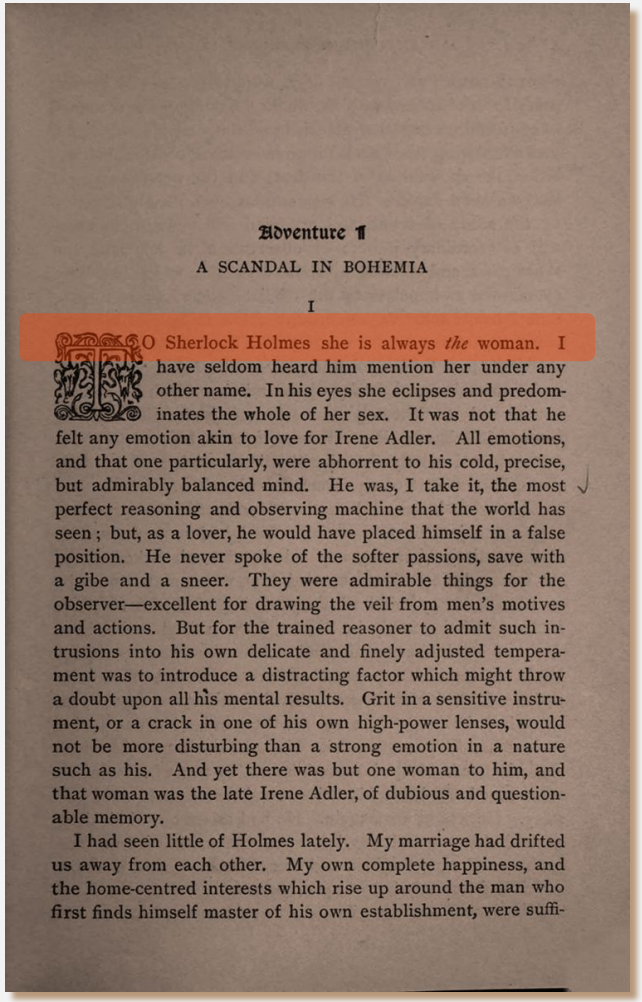
Working memory

+

Fluid Intelligence

=

Programming



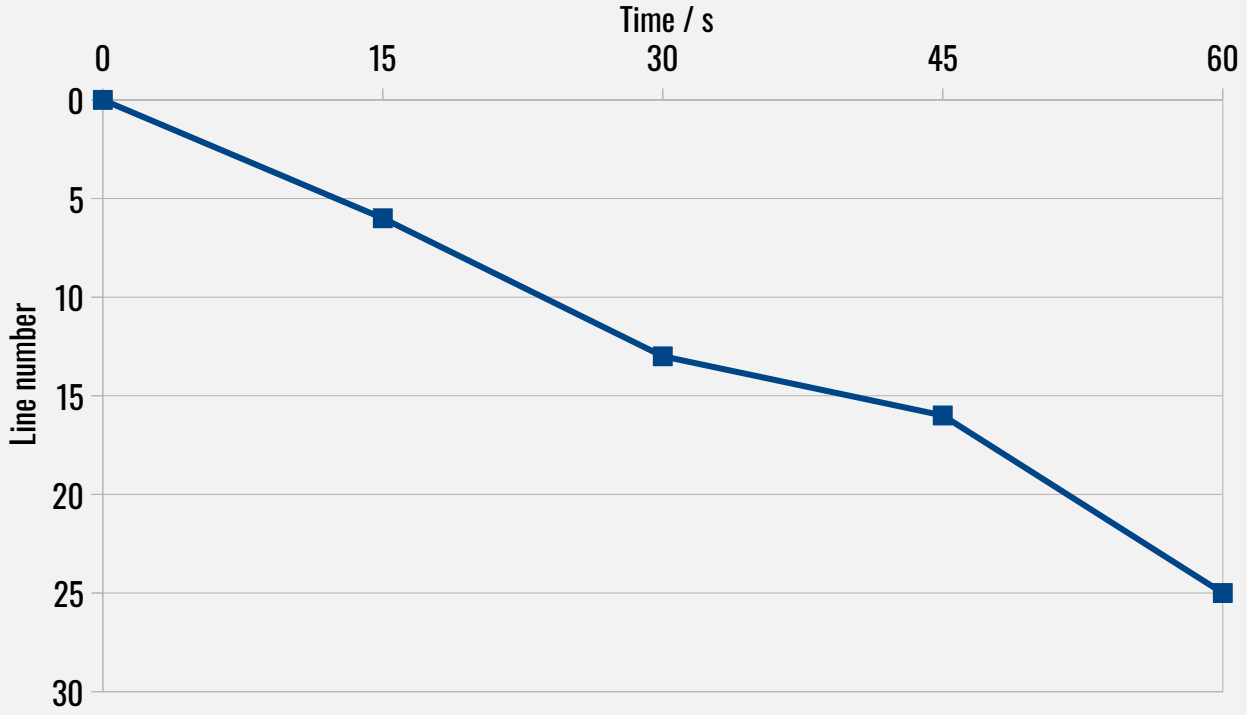
Adventure I

A SCANDAL IN BOHEMIA

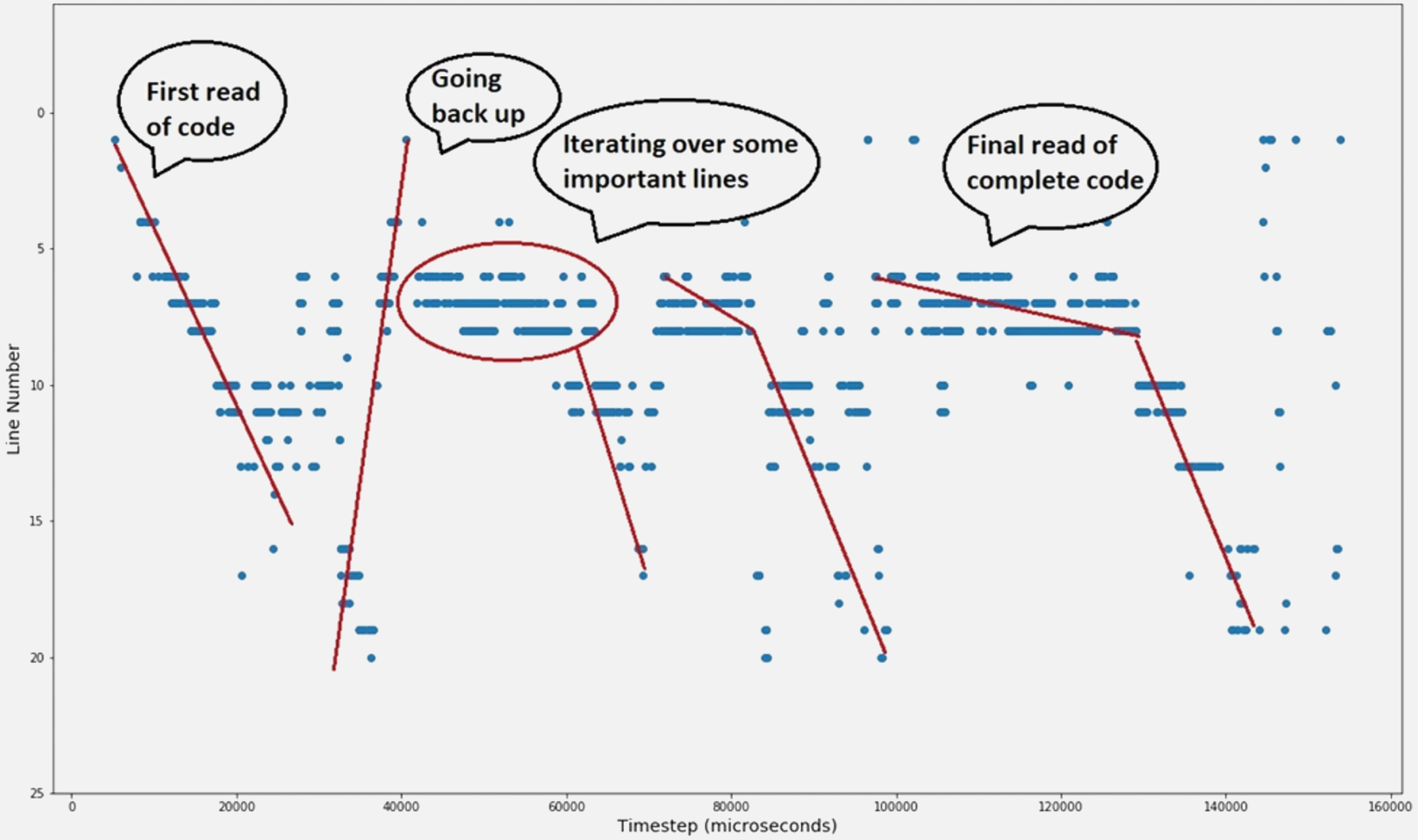
I

TO Sherlock Holmes she is always *the* woman. I have seldom heard him mention her under any other name. In his eyes she eclipses and predominates the whole of her sex. It was not that he felt any emotion akin to love for Irene Adler. All emotions, and that one particularly, were abhorrent to his cold, precise, but admirably balanced mind. He was, I take it, the most perfect reasoning and observing machine that the world has seen; but, as a lover, he would have placed himself in a false position. He never spoke of the softer passions, save with a gibe and a sneer. They were admirable things for the observer—excellent for drawing the veil from men's motives and actions. But for the trained reasoner to admit such intrusions into his own delicate and finely adjusted temperament was to introduce a distracting factor which might throw a doubt upon all his mental results. Grit in a sensitive instrument, or a crack in one of his own high-power lenses, would not be more disturbing than a strong emotion in a nature such as his. And yet there was but one woman to him, and that woman was the late Irene Adler, of dubious and questionable memory.

I had seen little of Holmes lately. My marriage had drifted us away from each other. My own complete happiness, and the home-centred interests which rise up around the man who first finds himself master of his own establishment, were suffi-



Measuring code comprehension effort using code reading pattern, Sayani Mondal et al., 2022

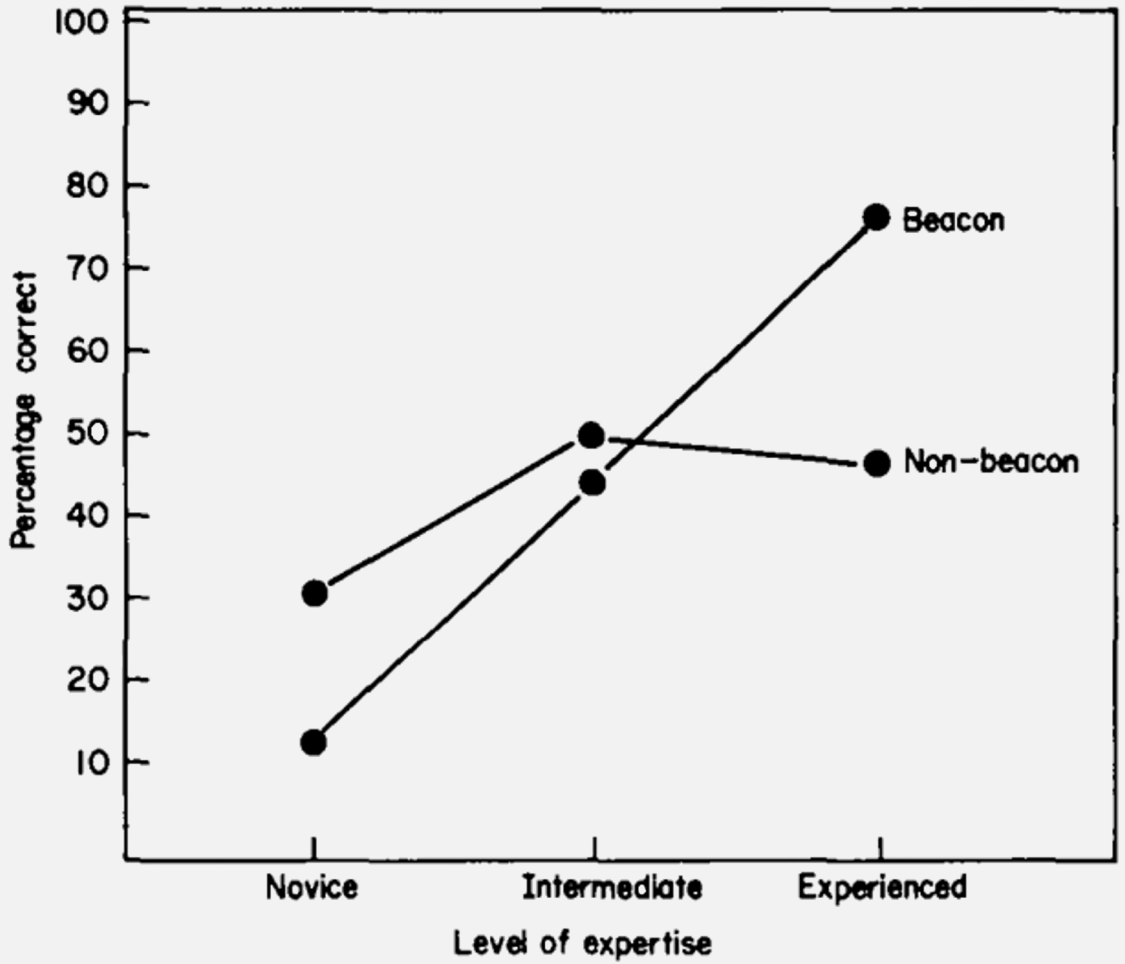


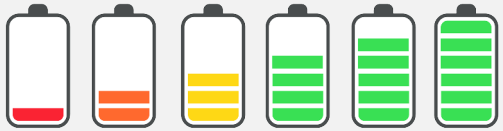
Towards a theory of the comprehension of computer programs, Ruven Brooks, 1981

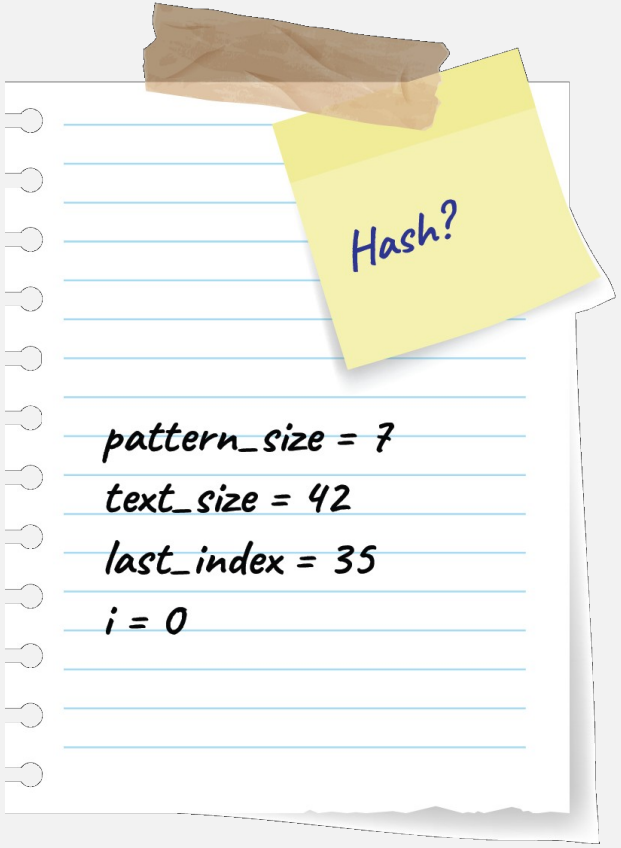
```
auto binary_search(auto const& range, auto const& value){
    auto high = range.size();
    decltype(high) low = 0;

    while (low < high) {
        auto middle = low + (high - low) / 2;
        if (range[middle] < value) {
            low = middle + 1;
        } else {
            high = middle;
        }
    }
    return low;
}
```

```
auto binary_search(auto const& range, auto const& value){
    auto high = range.size();
    decltype(high) low = 0;

    while (low < high) {
        auto middle = low + (high - low) / 2;
        if (range[middle] < value) {
            low = middle + 1;
        } else {
            high = middle;
        }
    }
    return low;
}
```

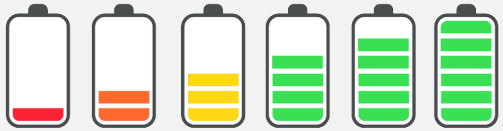
```
(1)   auto first = span.front();
(2)   auto high = range.size();
(3)   auto middle = low + (high - low) / 2;
(4)   auto min = iter;
(5)   decltype(high) low = 0;
(6)   high = middle;
(7)   if (*next < *min)
(8)   if (i != last_index)
(9)   if (range[middle] < value) {
(10)  low = middle + 1;
(11)  return low;
(12)  while (low < high) {
```

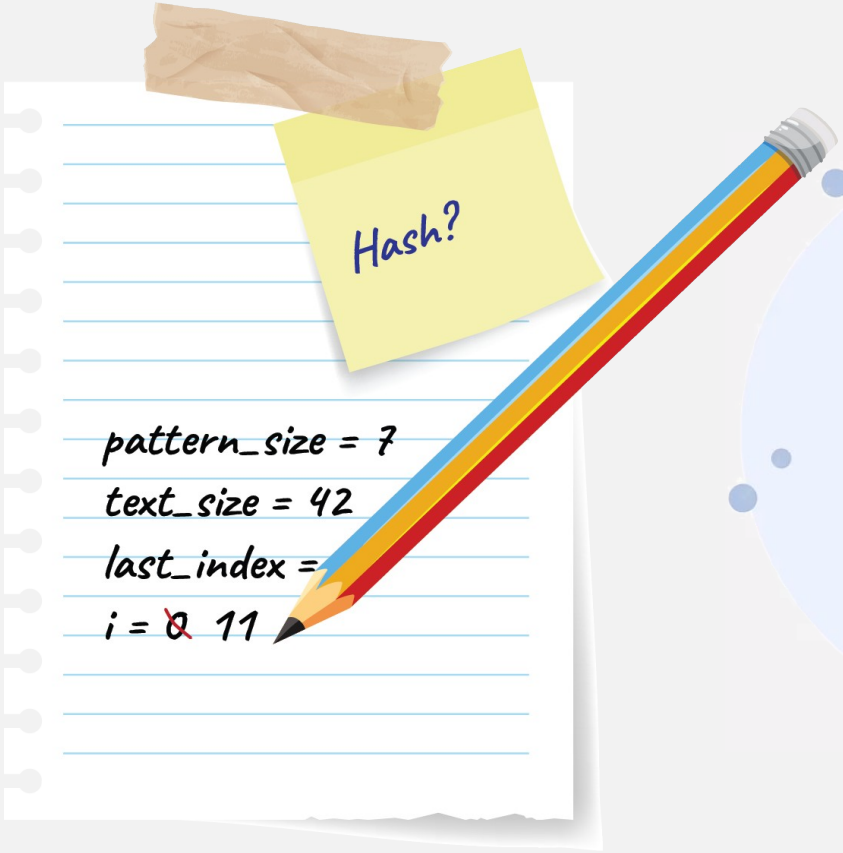

100
90
80
70
60
50
40
30
20
10
Percentage correct
Beacon
Non-beacon
Novice
Intermediate
Experienced
Level of expertise





Short-term memory

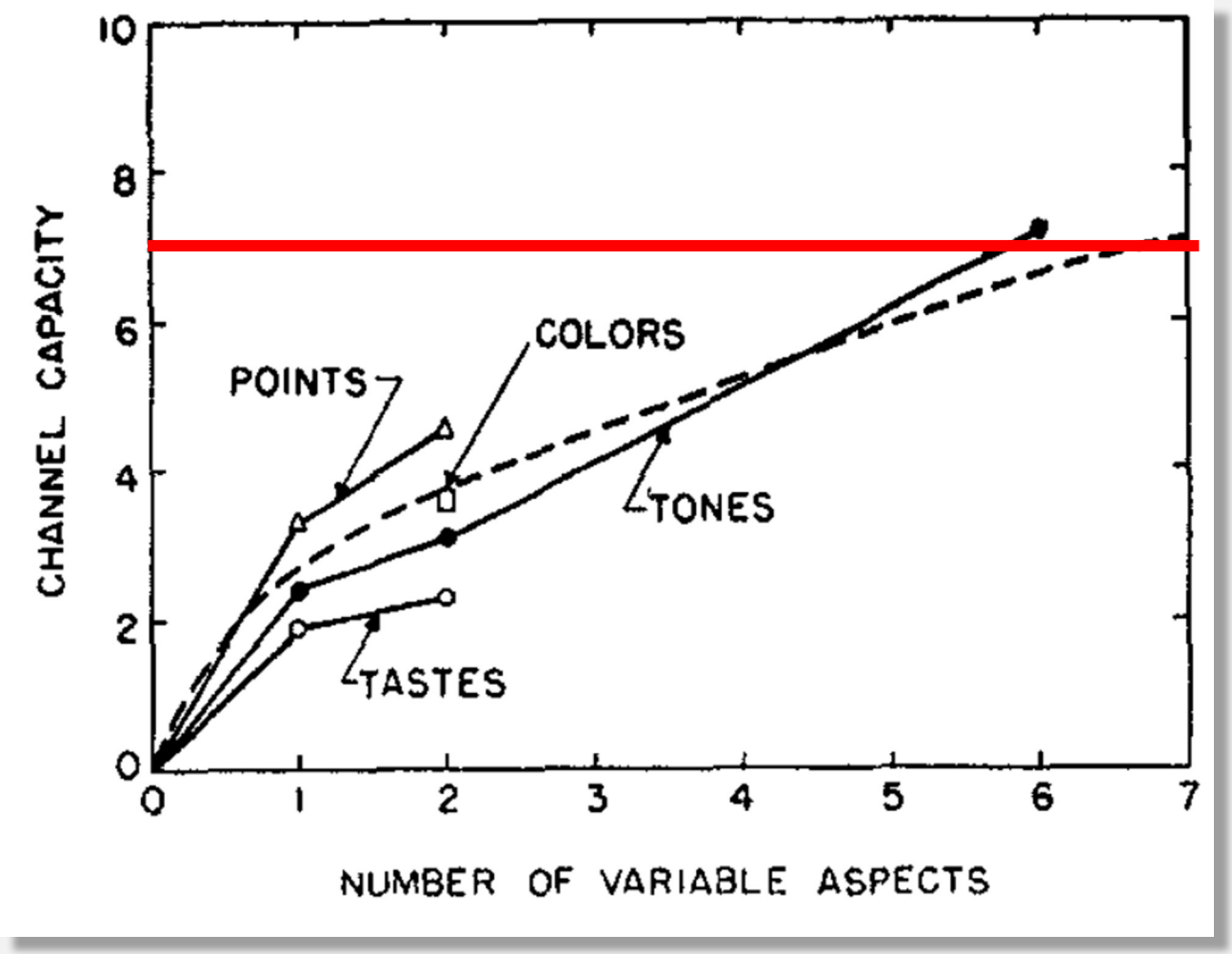




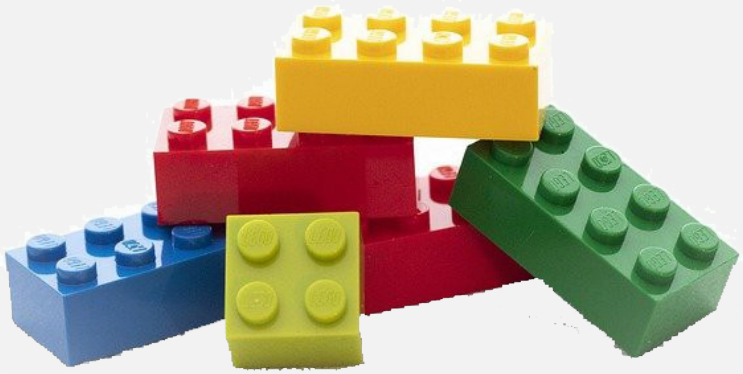
+ Short-term memory Processing = Working memory

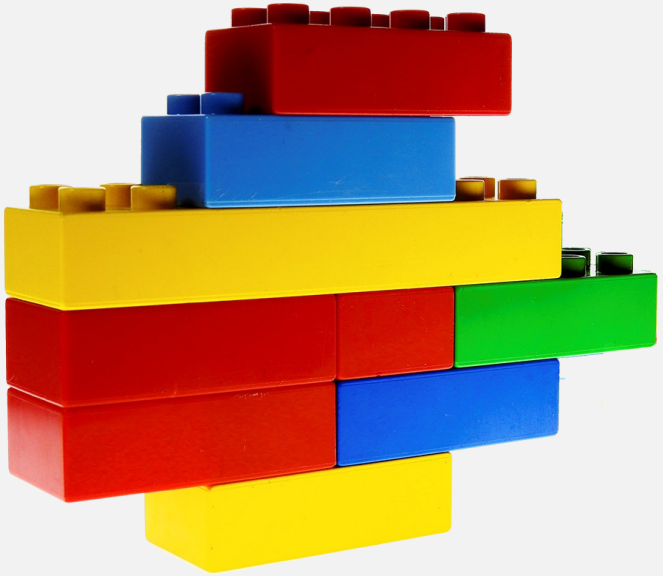
```
auto process(auto text, auto pattern, auto callback) {
  auto pattern_size = pattern.size();
  auto pattern_hash = Hash{pattern};
  auto last_index = text.size() - pattern_size;

  auto text_hash = Hash{text.substr(0, pattern_size)};

  for (auto i = 0zu; i <= last_index; ++i) {
    if (pattern_hash == text_hash)
      if (pattern == text.substr(i, pattern_size))
        callback(i);

    if (i != last_index)
      text_hash.update(text[i + pattern_size]);
  }
}
```

The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information, George A. Miller, 1956



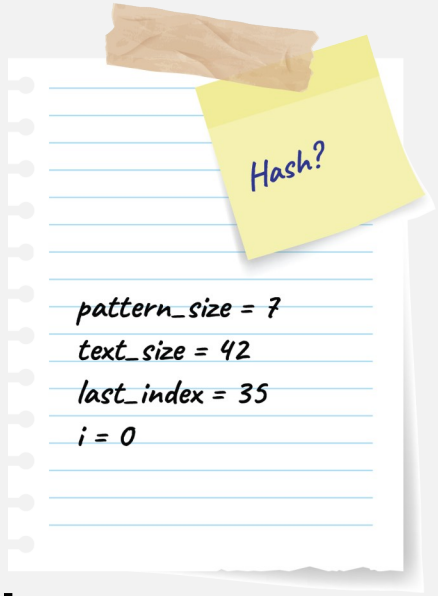
Chunking





```
if (pattern_hash == text_hash)
  if (pattern == text.substr(i, pattern_size))
    callback(i);
```

Callback if match



Short-term memory

Long-term memory

Cat’s lives	Six pack
Soccer team	Octopus
Tricycle	Hand
Angry men	Bowling pins
Unicorn	Couple
Lucky number	Four-leaf clover

Cat’s lives	Six pack
Soccer team	Octopus
Tricycle	Hand
Angry men	Bowling pins
Unicorn	Couple
Lucky number	Four-leaf clover

(9) Cat's lives

(11) Soccer team

(3) Tricycle

(12) Angry men

(1) Unicorn

(7) Lucky number

(6) Six pack

(8) Octopus

(5) Hand

(10) Bowling pins

(2) Couple

(4) Four-leaf clover

(1) Unicorn

(2) Couple

(3) Tricycle

(4) Four-leaf clover

(5) Hand

(6) Six pack

(7) Lucky number

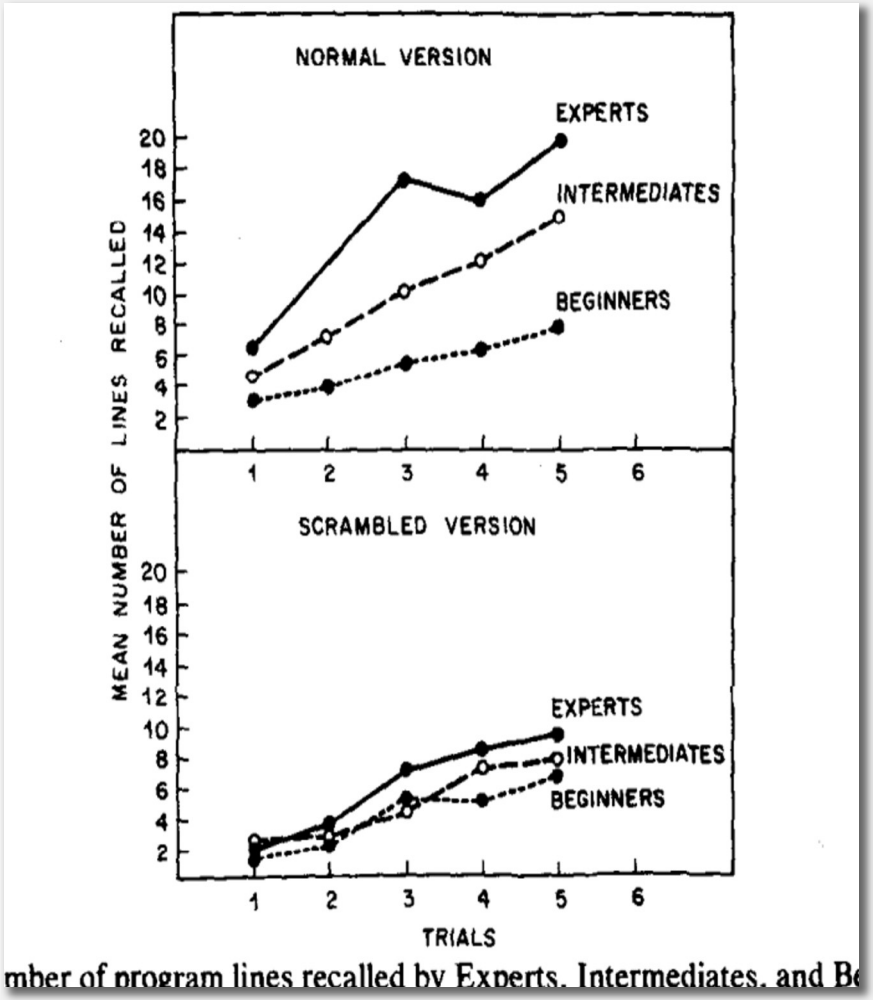
(8) Octopus

(9) Cat’s lives

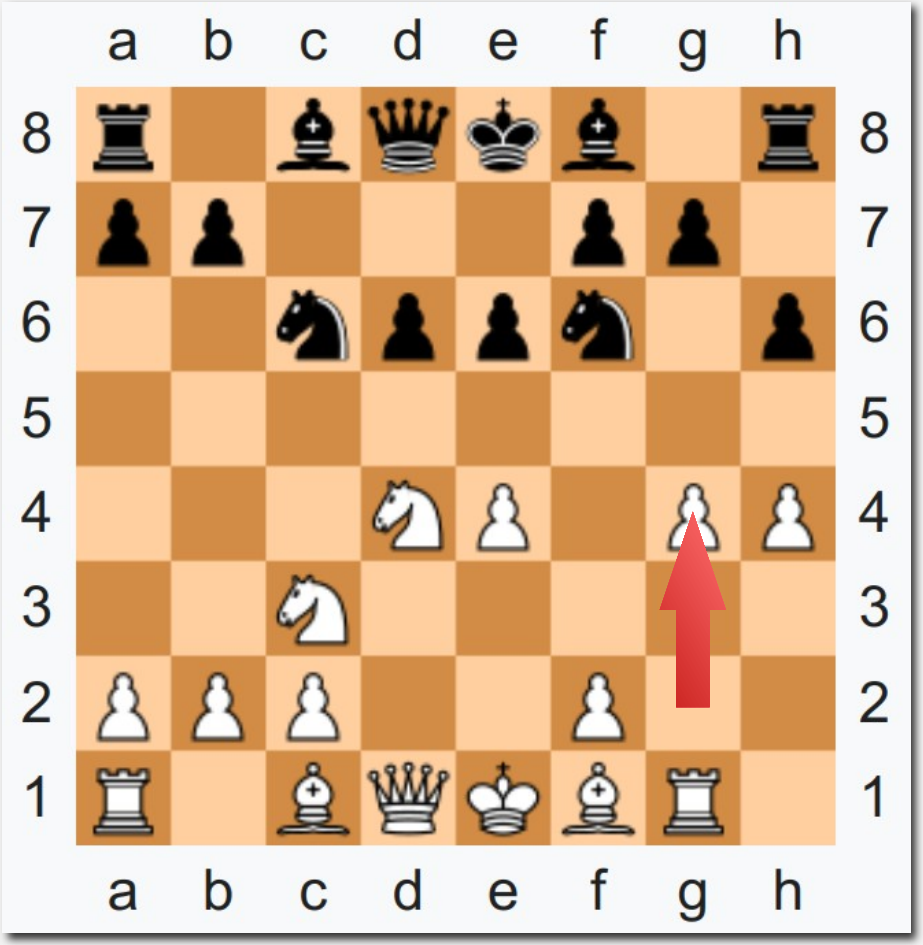
(10) Bowling pins

(11) Soccer team

(12) Angry men



mber of program lines recalled by Experts, Intermediates, and Be



The
Scheveningen Variation
of the Sicilian Defence
followed by
the Keres attack

Slot

(short-term memory mediated)

for-i Plan/ Pattern

(long-term memory)

```
for (auto i = 0; i <          ; ++i) {


}
```

```
auto process(auto text, auto pattern, auto callback) {
  auto pattern_size = pattern.size();
  auto pattern_hash = Hash{pattern};
  auto last_index = text.size() - pattern_size;

  auto text_hash = Hash{text.substr(0, pattern_size)};

  for (auto i = 0uz; i <= last_index; ++i) {
    if (pattern_hash == text_hash)
      if (pattern == text.substr(i, pattern_size))
        callback(i);

    if (i != last_index)
      text_hash.update(text[i + pattern_size]);
  }
}
```

Haystack & Needle

Callback function

Lots of local variables

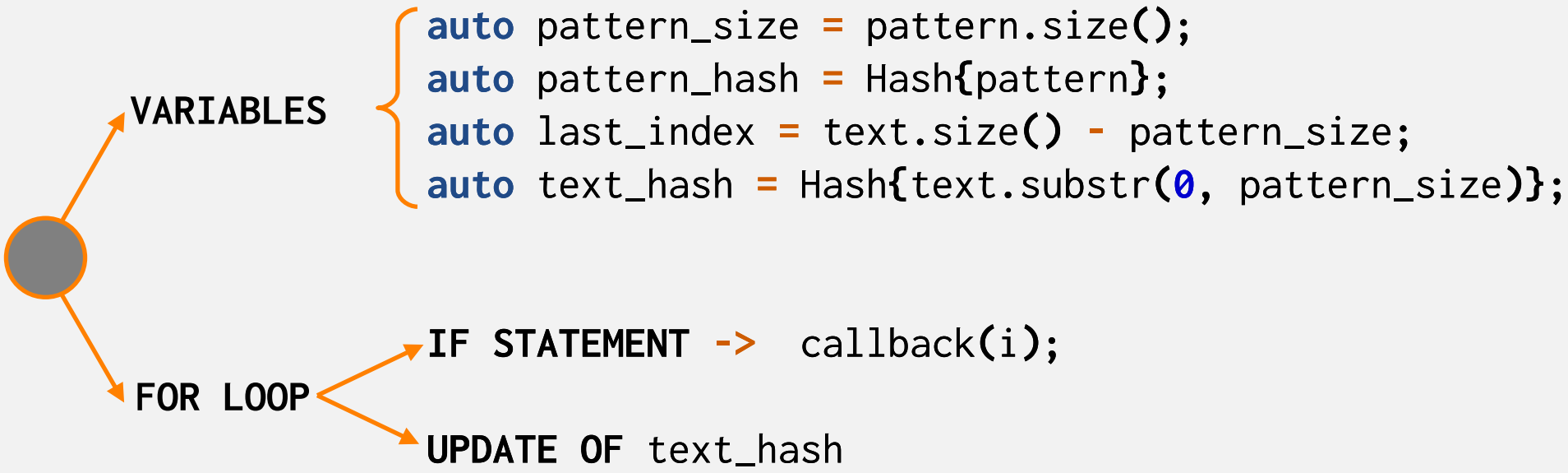
Text hash

For-i loop with a weird upper bound

If statement with hash and text comparison

If true callback invoked with index

Strange hash update


VARIABLES
auto pattern_size = pattern.size();
auto pattern_hash = Hash{pattern};
auto last_index = text.size() - pattern_size;
auto text_hash = Hash{text.substr(0, pattern_size)};
FOR LOOP
IF STATEMENT -> callback(i);
UPDATE OF text_hash

```
auto pattern_size = pattern.size();
auto pattern_hash = Hash{pattern};
auto last_index = text.size() - pattern_size;
auto text_hash = Hash{text.substr(0, pattern_size)};
for (auto i = 0uz; i <= last_index; ++i)
if (pattern_hash == text_hash)
if (pattern == text.substr(i, pattern_size))
callback(i);
if (i != last_index)
text_hash.update(text[i + pattern_size]);
```

```
constexpr auto do_a_lot = [](auto tasks, auto lock) {

  constexpr auto select = []<auto I>(auto&& tasks) {
    if constexpr (I == -1) return std::tuple{};
    else return std::make_tuple(std::get<I>(tasks));
  };

  constexpr auto completed = [](auto tasks, auto lock) {
    return [&]<std::size_t... Is>(std::index_sequence<Is...>) {
      return std::integer_sequence<int, (lock(std::get<Is>(tasks()))? int(Is): -1)...>{};
    }(std::make_index_sequence<std::tuple_size_v<decltype(tasks())>>{});
  };

  return [&]<int... Is>(std::integer_sequence<int, Is...>) {
    return std::tuple_cat(select. template operator()<Is>(tasks())...);
  }(completed(tasks, lock));
};
```

```
constexpr auto do_a_lot = [](auto tasks, auto lock) {

  constexpr auto select = []<auto I>(auto&& tasks) {
    if constexpr (I == -1) return std::tuple{};
    else return std::make_tuple(std::get<I>(tasks));
  };

  constexpr auto completed = [](auto tasks, auto lock) {
    return [&]<std::size_t... Is>(std::index_sequence<Is...>) {
      return std::integer_sequence<int, (lock(std::get<Is>(tasks()))? int(I          )...>{};
    }(std::make_index_sequence<std::tuple_size_v<decltype(input())>>{});
  };

  return [&]<int... Is>(std::integer_sequence<int, Is...>) {
    return std::tuple_cat(select. template operator()<Is>(tasks())...);
  }(completed(tasks, lock));
};
```

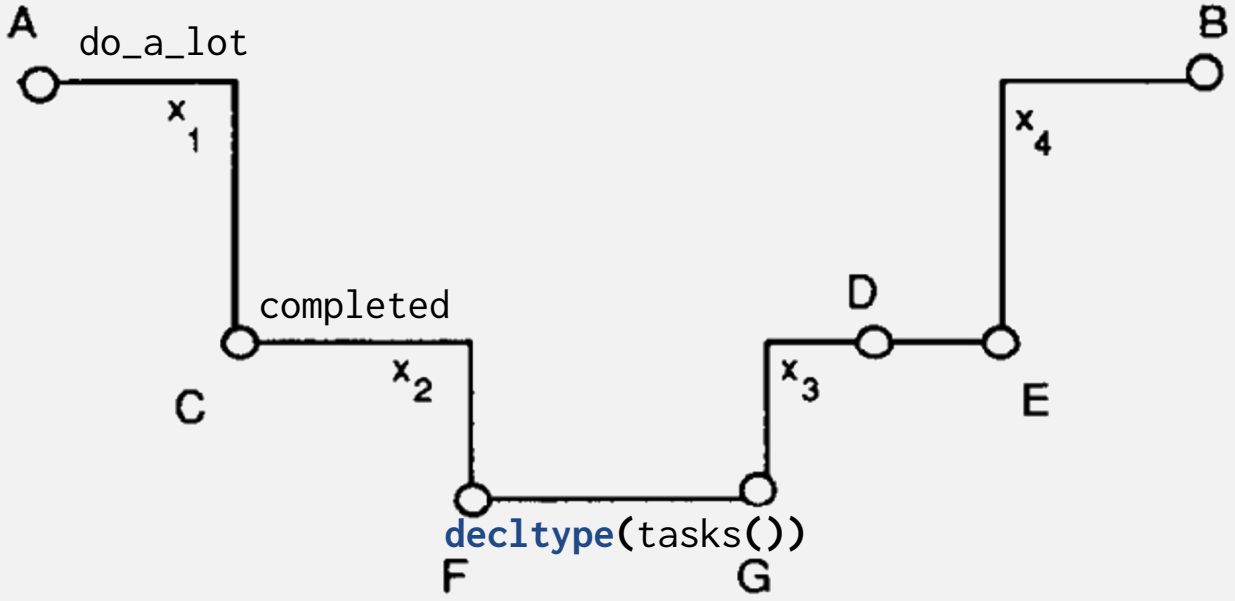


Figure 5 Landscape model of program comprehension

Programming is like reading & writing prose, but not exactly.

There's also working memory, patterns to be learned and fluid intelligence.

Code comprehension involves

Beacons—Chunking—

Plans—Abstaction lowering—

Bottom-up analysis

```
constexpr auto do_a_lot = [](auto tasks, auto lock) {

  constexpr auto select = []<auto I>(auto&& tasks) {
    if constexpr (I == -1) return std::tuple{};
    else return std::make_tuple(std::get<I>(tasks));
  };

  constexpr auto completed = [](auto tasks, auto lock) {
    return [&]<std::size_t... Is>(std::index_sequence<Is...>) {
      return std::integer_sequence<int, (lock(std::get<Is>(tasks()))? int(Is): -1)...>{};
    }(std::make_index_sequence<std::tuple_size_v<decltype(input())>>{});
  };

  return [&]<int... Is>(std::integer_sequence<int, Is...>) {
    return std::tuple_cat(select. template operator()<Is>(tasks())...);
  }(completed(tasks, lock));
};
```

```
constexpr auto filter_by = [](auto tuple, auto pred) {

  constexpr auto select_if = []<auto I>(auto&& tuple) {
    if constexpr (I == -1) return std::tuple{};
    else return std::make_tuple(std::get<I>(tuple));
  };

  constexpr auto make_mask = [](auto tuple, auto pred) {
    return [&]<std::size_t... Is>(std::index_sequence<Is...>) {
      return std::integer_sequence<int, (pred(std::get<Is>(tuple()))? int(Is): -1)...>{};
    }(std::make_index_sequence<std::tuple_size_v<decltype(tuple())>>{});
  };

  return [&]<int... Is>(std::integer_sequence<int, Is...>) {
    return std::tuple_cat(select_if. template operator()<Is>(tuple())...);
  }(make_mask(tuple, pred));
};
```

How not to

RUIN YOUR PROGRAMS

from the brain's POV

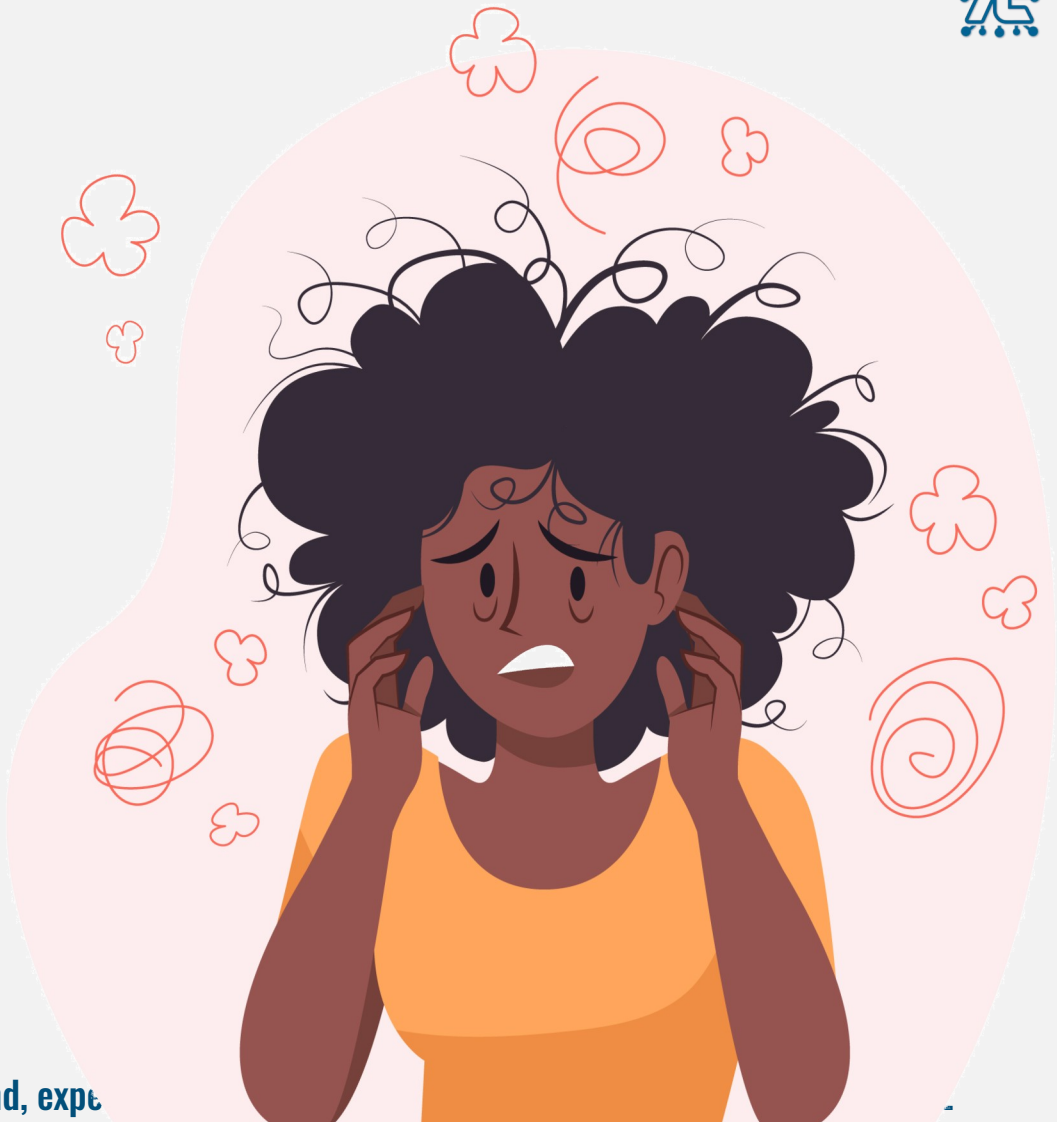
101

Cognitive (over)load

Intrinsic
Difficulty of the problem
· variable tracking
· nested loops & conditions

Extraneous
External information needs
· unfamiliar functions
· side effects

Germane
Deep processing to internalize/ learn new schemas



```cpp
template <typename...Fs>
struct overloaded : Fs... {
  using Fs::operator()...;
};
```

intrinsic

extraneous

germane

```cpp
using Variant_t = std::variant<int, double, std::string>;
std::vector<Variant_t> variants{42, 3.1415927, "There are 42 rabbits in the hat."};

for (const auto& v : variants) {
  std::print("Has answer: {}\n",
    std::visit( overloaded{
        [](int i) { return i == 42; },
        [tolerance=0.05](double d) { return std::abs(d - 42) < tolerance; },
        [](std::string_view s) { return s.contains("42") || s.contains("forty two"); }
      }, v
    )
  );
}
```

```
auto process(auto text, auto pattern, auto callback) {
  auto pattern_size = pattern.size();
  auto pattern_hash = Hash{pattern};
  auto last_index = text.size() - pattern_size;

  auto text_hash = Hash{text.substr(0, pattern_size)};

  for (auto i = 0uz; i <= last_index; ++i) {
    if (pattern_hash == text_hash)
      if (pattern == text.substr(i, pattern_size))
        callback(i);

    if (i != last_index)
      text_hash.update(text[i + pattern_size]);
  }
}
```

```
auto process(auto text, auto pattern, auto callback) {
1  std::size_t base = 257;
2  std::size_t mod = 1000000007;
3  auto text_size = text.size();
4  auto pattern_size = pattern.size();
5  auto pattern_hash = [&](auto hash){ for (auto c : pattern) hash = (hash * base + c) % mod; return hash; }(0uz);
                                                  11
                                  10
6  auto start = text.substr(0, pattern_size);
                                                                  12      13
7  auto text_hash = std::ranges::fold_left(start, 0uz, [](auto acc, auto c) { return (acc * base + c) % mod; });
                                                                        14      15
8  auto exp = std::accumulate(start.begin(), start.end(), 1uz, [](auto acc, auto c) { return (acc * base) % mod; });

9  for (auto i = 0uz; i <= (text_size - pattern_size); ++i) {
     if (pattern_hash == text_hash)
       if (pattern == text.substr(i, pattern_size))
         callback(i);

     if (i + pattern_size != text_size){
       text_hash = (text_hash * base + text[i + pattern_size]) % mod;
       text_hash = (text_hash + mod - text[i] * exp % mod) % mod;
     }
   }
}
```

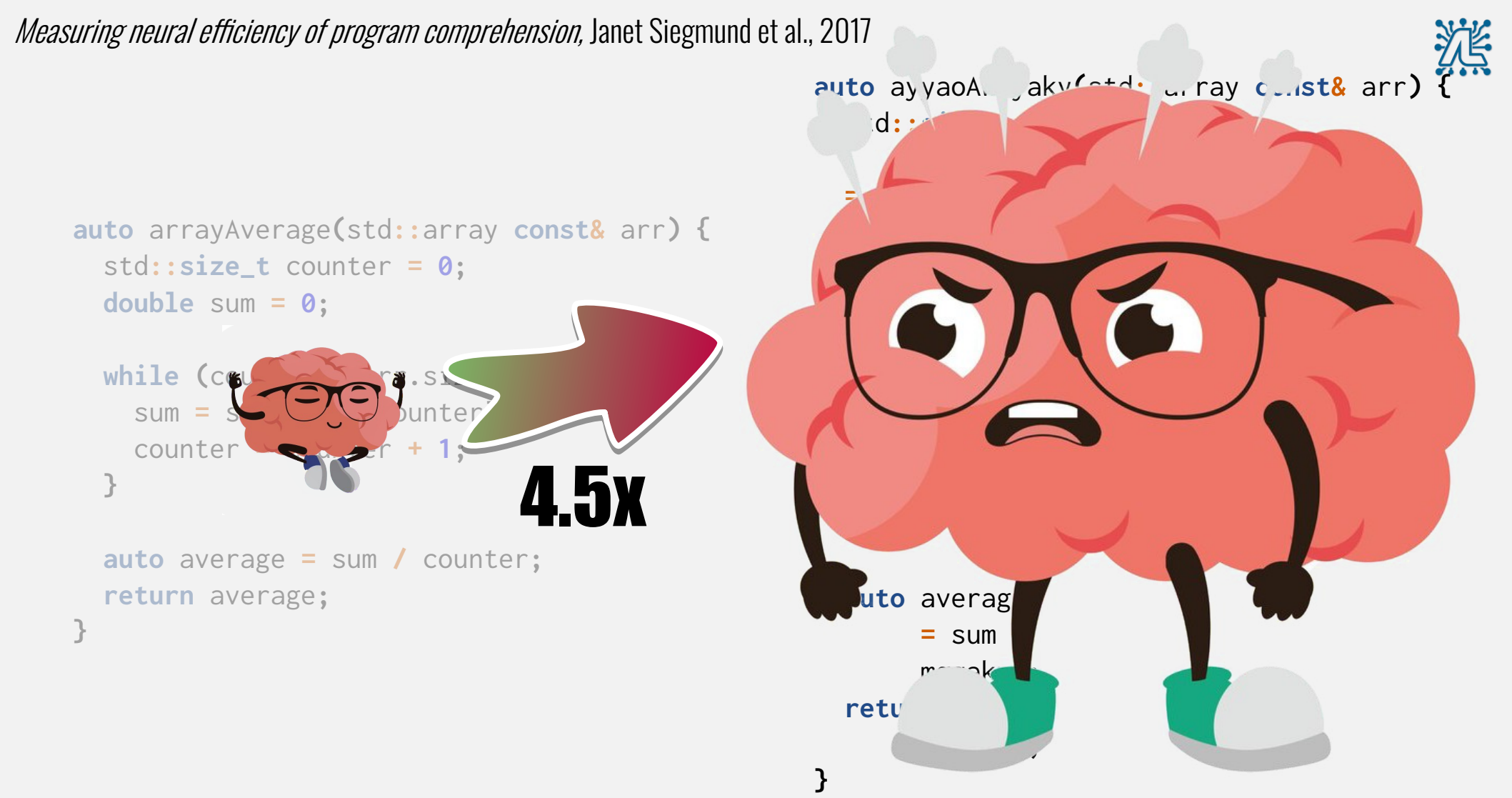
```
 auto pwucyc(auto tgherx, auto pwucyc, auto c) {
1  auto ps = pwucyc.size();
2  auto ph = H{pwucyc};
3  auto lwhcadt = tgherx.size() - ps;

4  auto hhdcit = H{tgherx.substr(0, ps)};

5  for (auto i = 0uz; i <= lwhcadt; ++i) {
     if (ph == hhdcit)
       if (pwucyc == tgherx.substr(i, ps))
         c(i);

     if (i != lwhcadt)
       hhdcit.run(r[i + ps]);
   }
 }
```

Only 5 values to track!

```
auto arrayAverage(std::array const& arr) {
  std::size_t counter = 0;
  double sum = 0;

  while (counter < arr.size()) {
    sum = sum + arr[counter];
    counter = counter + 1;
  }

  auto average = sum / counter;
  return average;
}
```

```
auto ayyaoAwyyaky(std::array const& arr) {
  std::size_t
                mgqakyy
  = 0;
  double sum = 0;

  while (mgqakyy
       < arr.size()) {
    sum =
    sum + arr[mgqakyy];
    mgqakyy
    = mgqakyy + 1;
  }

  auto average
       = sum /
       mgqakyy;
  return
       average;
}
```

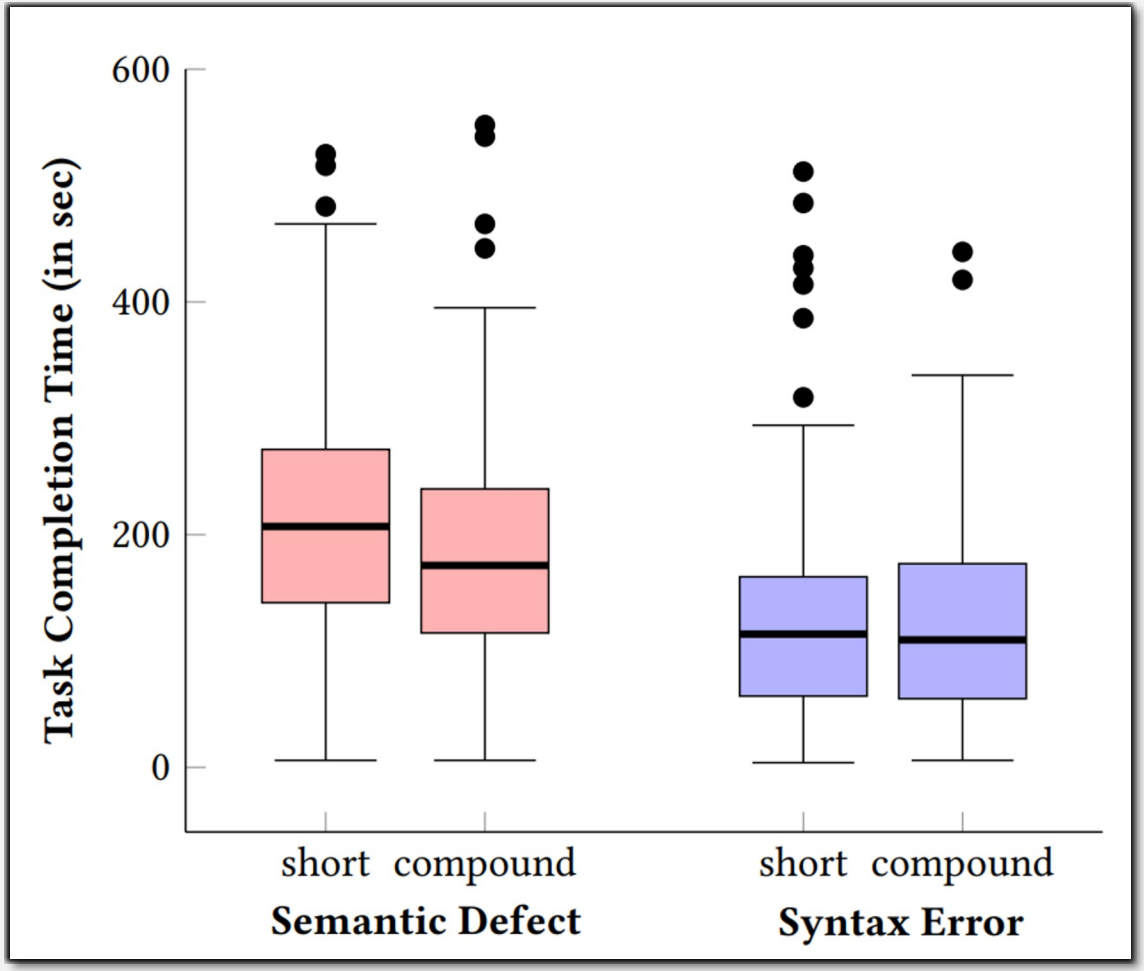
Measuring neural efficiency of program comprehension, Janet Siegmund et al., 2017



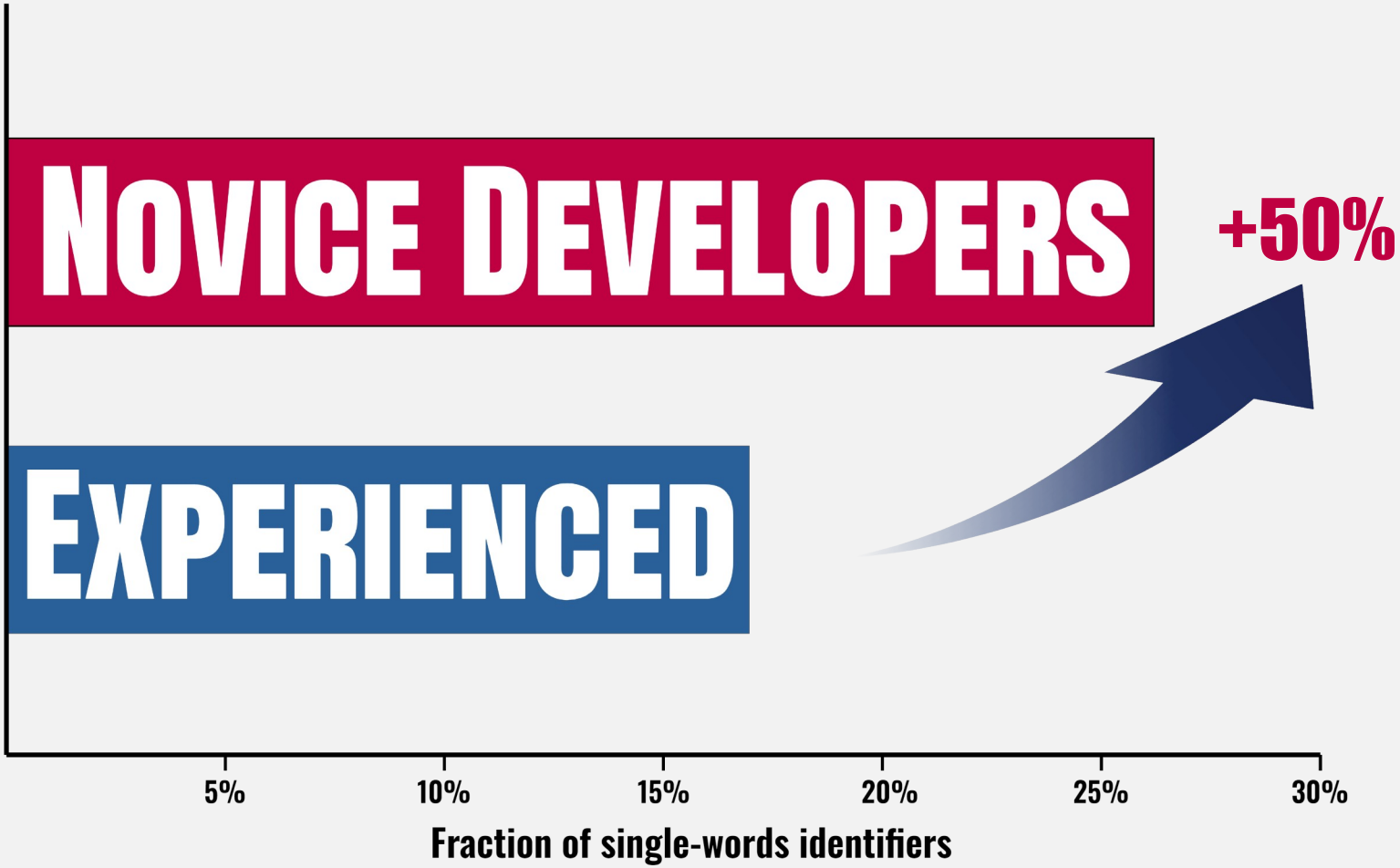
```
auto process(auto text, auto pattern, auto callback) {
  auto size = pattern.size();
  auto hash = Hash{pattern};
  auto last = text.size() - pattern_size;

  auto window = Hash{text.substr(0, pattern_size)};


  for (auto i = 0uz; i <= last; ++i) {
    if (hash == window)
      if (pattern == text.substr(i, size))
        callback(i);

    if (i != last)
      window.update(text[i + size]);
  }
}
```

```
auto process(auto text, auto pattern, auto callback) {
  auto pattern_size = pattern.size();
  auto pattern_hash = Hash{pattern};
  auto last_index = text.size() - pattern_size;

  auto text_hash = Hash{text.substr(0, pattern_size)};


  for (auto i = 0uz; i <= last_index; ++i) {
    if (pattern_hash == text_hash)
      if (pattern == text.substr(i, pattern_size))
        callback(i);

    if (i != last_index)
      text_hash.update(text[i + pattern_size]);
  }
}
```

Descriptive compound identifier names improve source code comprehension, Andrea Schankin et al., 2018



How Developers Choose Names, Dror Feitelson et al., 2021



```
auto process(auto text, auto pattern, auto callback) {
  auto pattern_size = pattern.size();
  auto pattern_hash = Hash{pattern};
  auto last_index = text.size() - pattern_size;

  auto text_hash = Hash{text.substr(0, pattern_size)};


  for (auto i = 0uz; i <= last_index; ++i) {
    if (pattern_hash == text_hash)
      if (pattern == text.substr(i, pattern_size))
        callback(i);

    if (i != last_index)
      text_hash.update(text[i + pattern_size]);
  }
}
```

```
auto find_index(auto text_to_search, auto pattern, auto callback) {
  auto pattern_size = pattern.size();
  auto pattern_hash = Hash{pattern};
  auto last_index_to_check = text_to_search.size() - pattern_size;

  auto text_window_hash =
                  RollingHash{text_to_search.substr(0, pattern_size)};

  for (auto i = 0uz; i <= last_index_to_check; ++i) {
    if (pattern_hash == text_window_hash)
      if (pattern == text_to_search.substr(i, pattern_size))
        callback(i);

    if (i != last_index_to_check)
      text_window_hash.update(text_to_search[i + pattern_size]);
  }
}
```

```
camelCaseRulesTheWorld
```

```
snake_case_is_the_king
```

To camelCase or under_score, Dave Binley et al., 2009

An Eye Tracking Study on camelCase and under_score Identifier Styles, Bonita Sharif et al., 2010

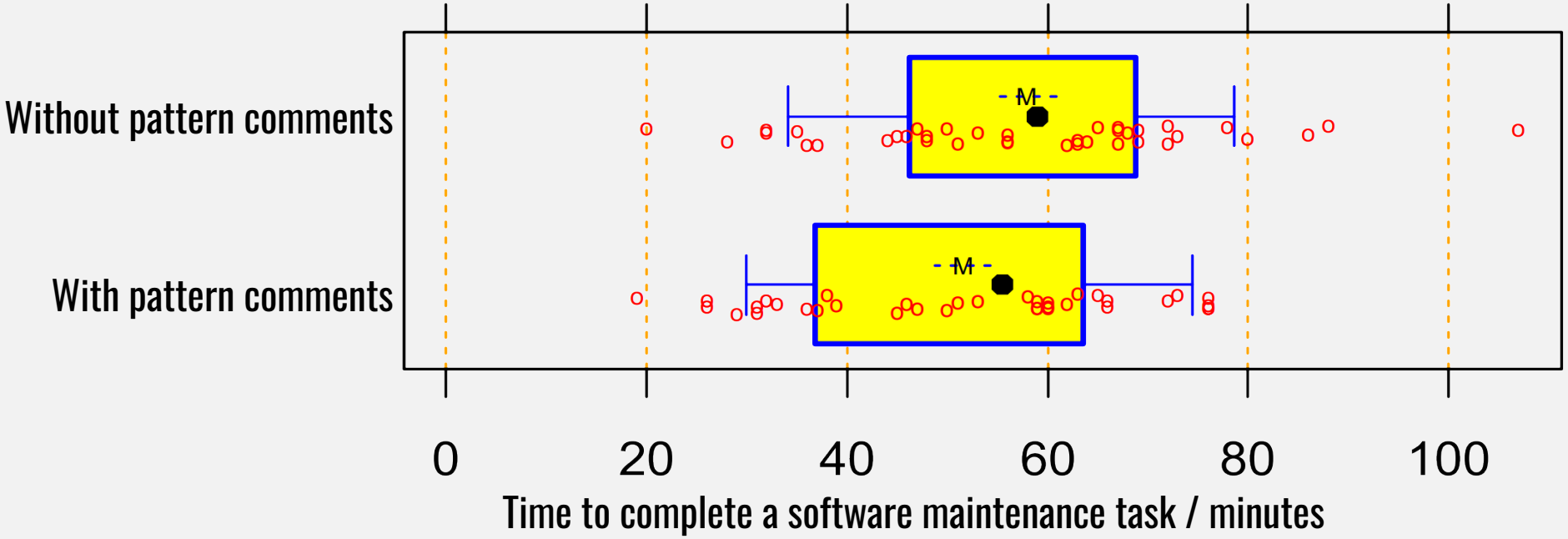
`it_doesntMatter`

`maybe_just_a_little_bit`

```
/// @brief Search for a pattern in a text
/// @param text_to_search The text to search
/// @param pattern The pattern to search for in the text
/// @param callback A callback function called with the index of a match

auto find_index(auto text_to_search, auto pattern, auto callback) {

  // ...

  for (auto i = 0uz; i <= last_index_to_check; ++i) {

    // ...

    if (i != last_index_to_check)
      // updates the rolling hash by shifting the window one char to the right
      text_window_hash.update(text_to_search[i + pattern_size]);
```

```
/// @brief Search for a pattern in a text
/// @param text_to_search The text to search
/// @param pattern The pattern to search for in the text
/// @param callback A callback function called with the index of a match
/// @note Implements the Rabin-Karp algorithm
auto find_index(auto text_to_search, auto pattern, auto callback) {

  // ...

  for (auto i = 0uz; i <= last_index_to_check; ++i) {

    // ...

    if (i != last_index_to_check)
      // updates the rolling hash by shifting the window one char to the right
      text_window_hash.update(text_to_search[i + pattern_size]);
```

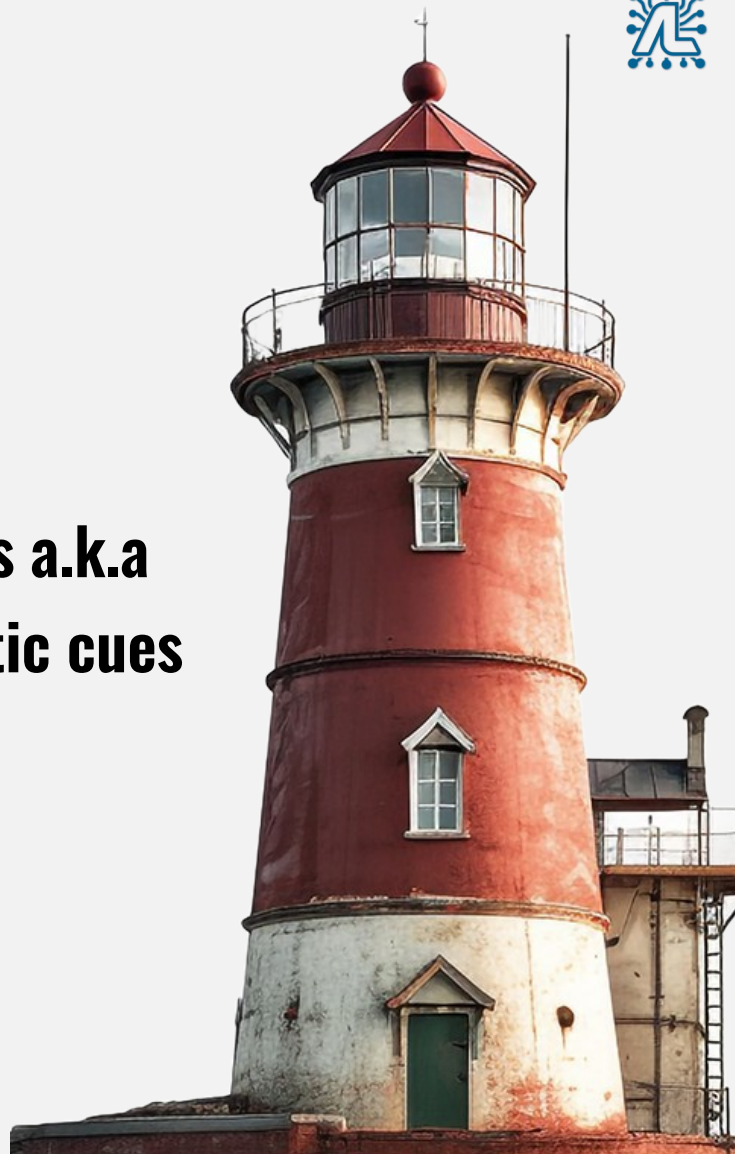
Two controlled experiments assessing the usefulness of design pattern documentation in program maintenance,
Lutz Prechelt et al., 2011



```cpp
struct RollingHash {

  RollingHash(std::string_view sv);

  std::size_t update(char c);

  std::size_t hash() const;

  std::string_view str() const;

};
```

```
struct RollingHash {
  RollingHash(std::string_view sv);
  std::size_t update(char c);
  std::size_t hash() const;
  std::string_view str() const;
};
```

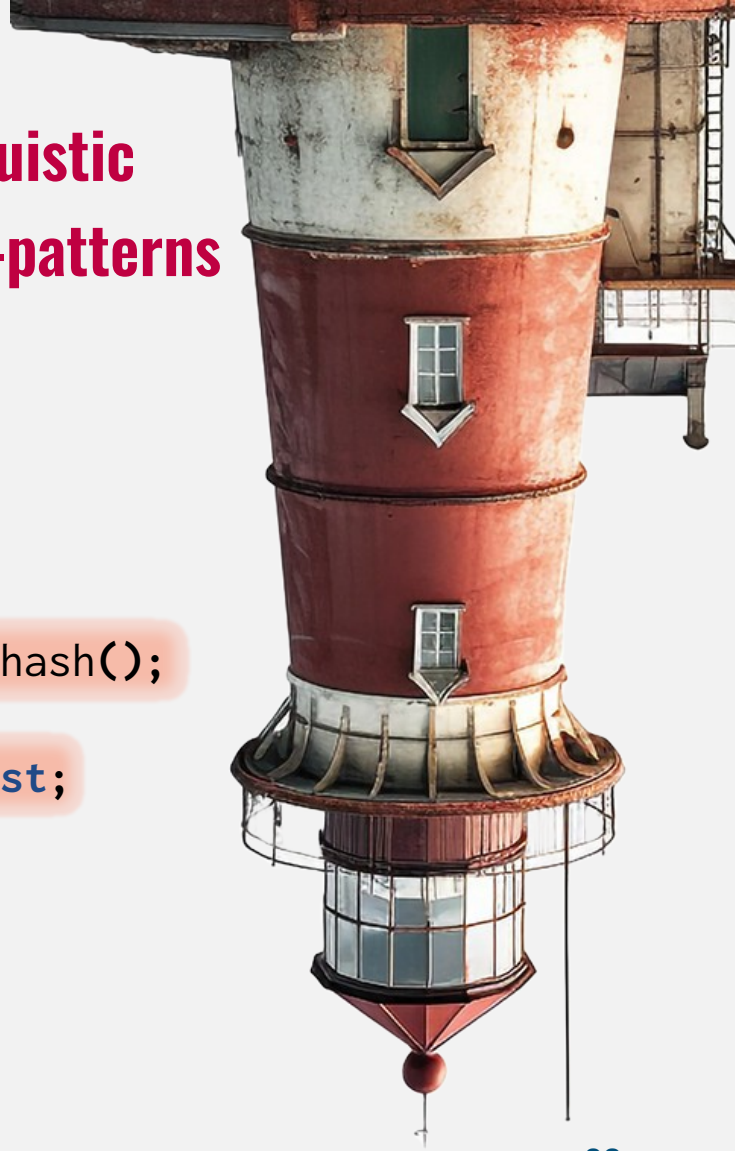
Beacons a.k.a
Linguistic cues



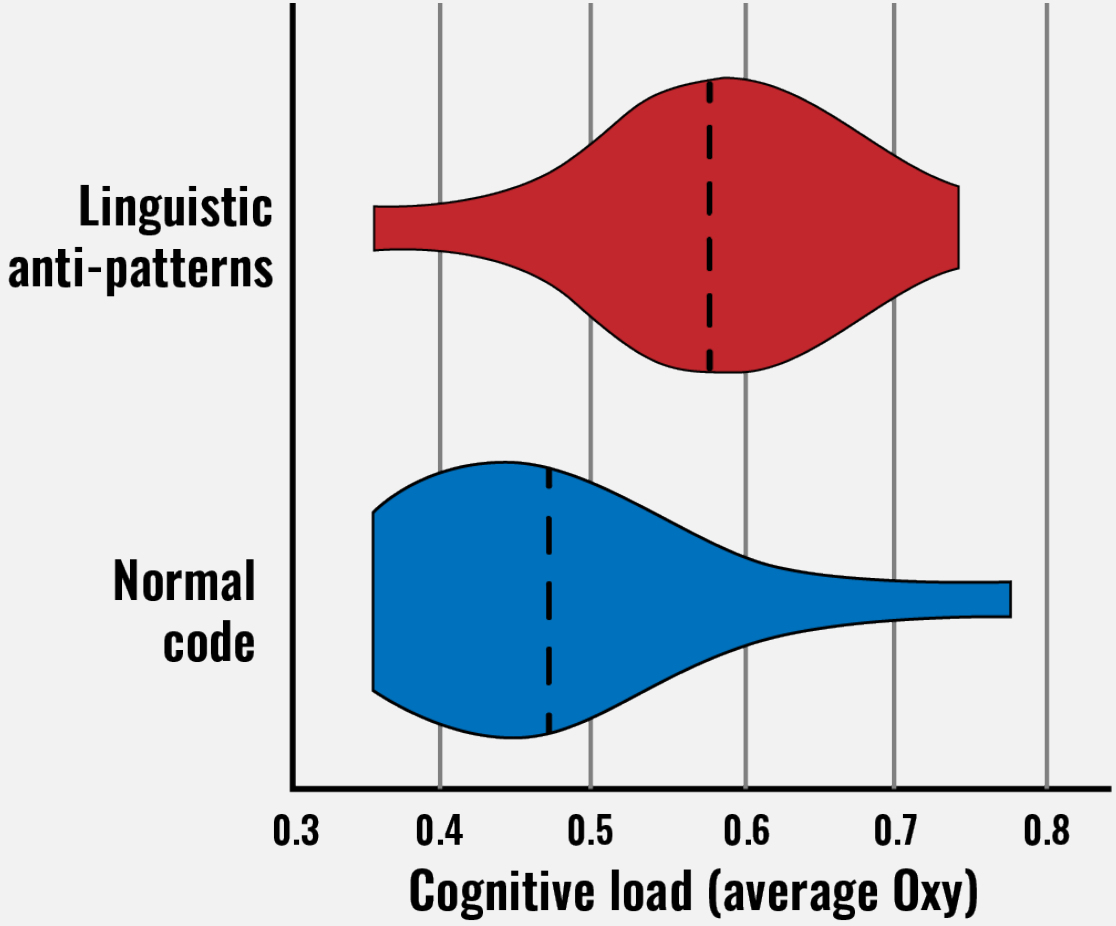
Measuring the impact of lexical and structural inconsistencies on developers' cognitive load during bug localization, Sarah Fakhoury et al., 2020

Linguistic anti-patterns

```
struct RollingHash {

  RollingHash(std::string_view sv);

  void shift_by(char c);

  std::pair<std::size_t, std::string_view> get_hash();

  std::size_t is_equal(std::string_view sv) const;

};
```



Measuring the impact of lexical and structural inconsistencies on developers' cognitive load during bug localization, Sarah Fakhoury et al., 2020



GOOD PROGRAMS == HAPPY BRAIN

—use familiar, expressive names (beacons)—

—keep the scope small (stm)—

—have regular structure (ltm)—

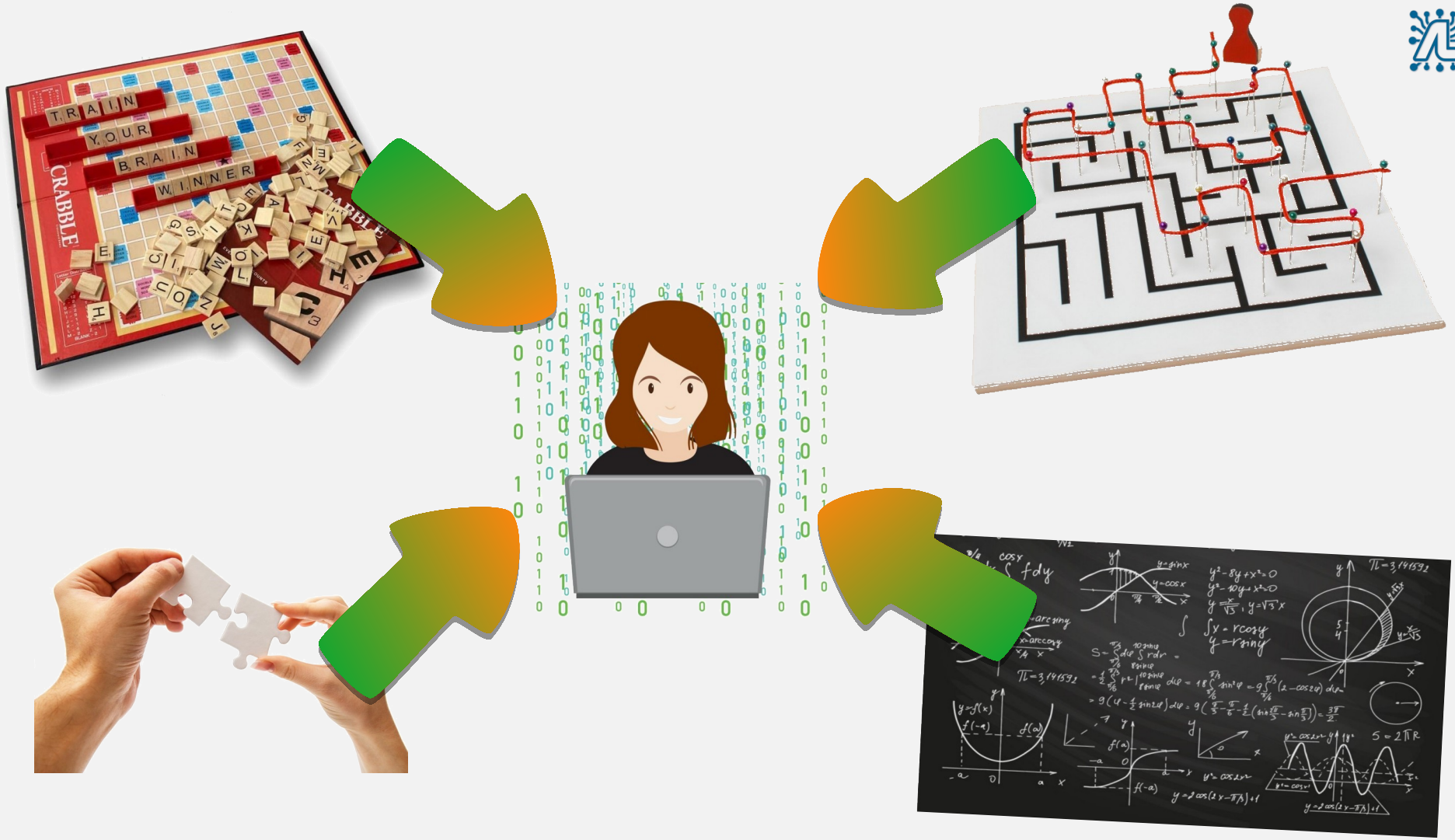
—actively use patterns (stm & ltm)—

—keep API promisses (linguistic cues & ap)

—

—useConsistentNamingConventions,

doesnt_matter_which—

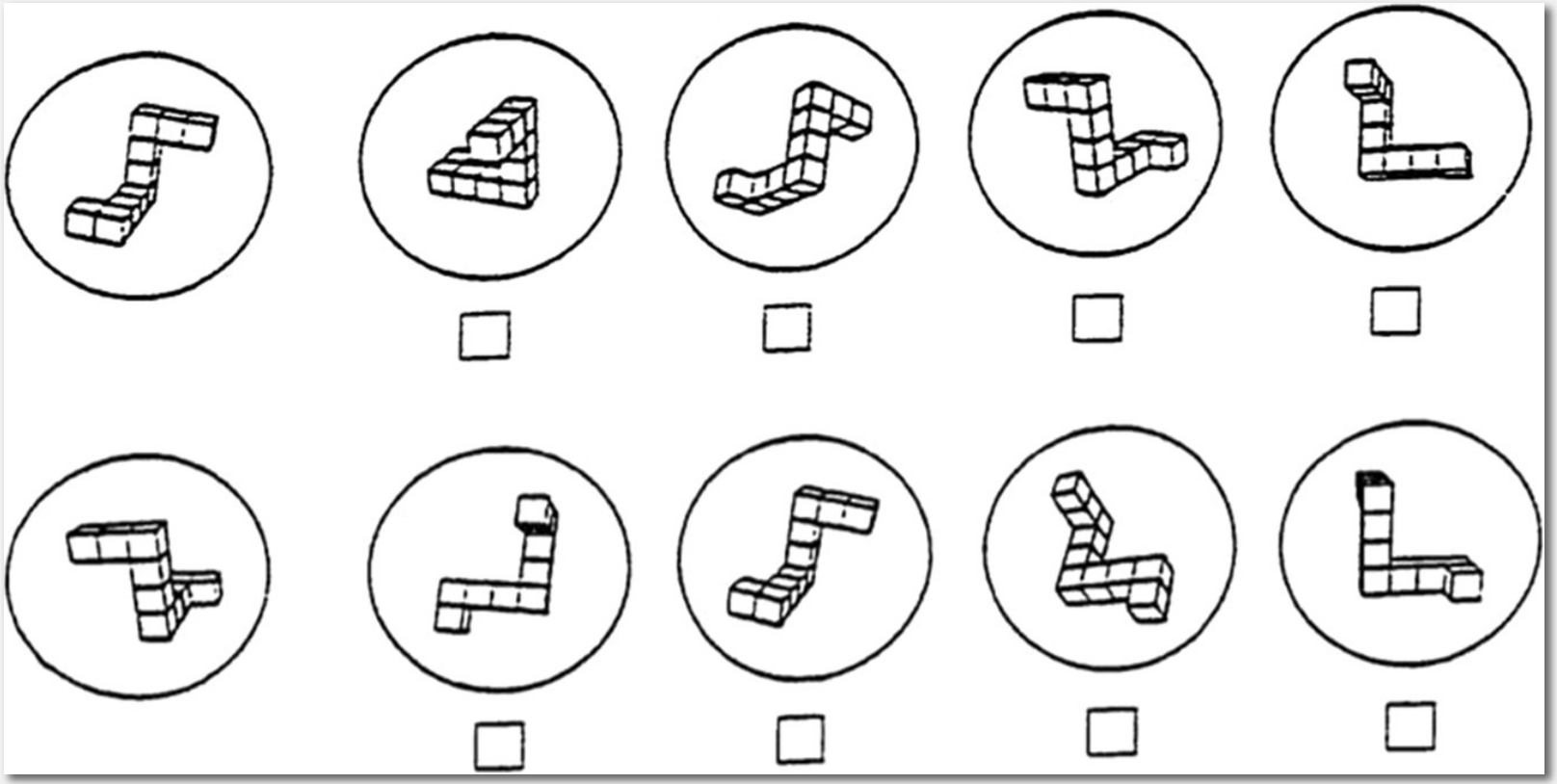

TRAIN
YOUR
BRAIN
WINNER
CRABBLE

Does Far Transfer Exist? Negative Evidence From Chess, Music, and Working Memory Training, Giovanni Sala et al., 2017
Working memory training in typically developing children: A multilevel meta-analysis, Giovanni Sala et al., 2020

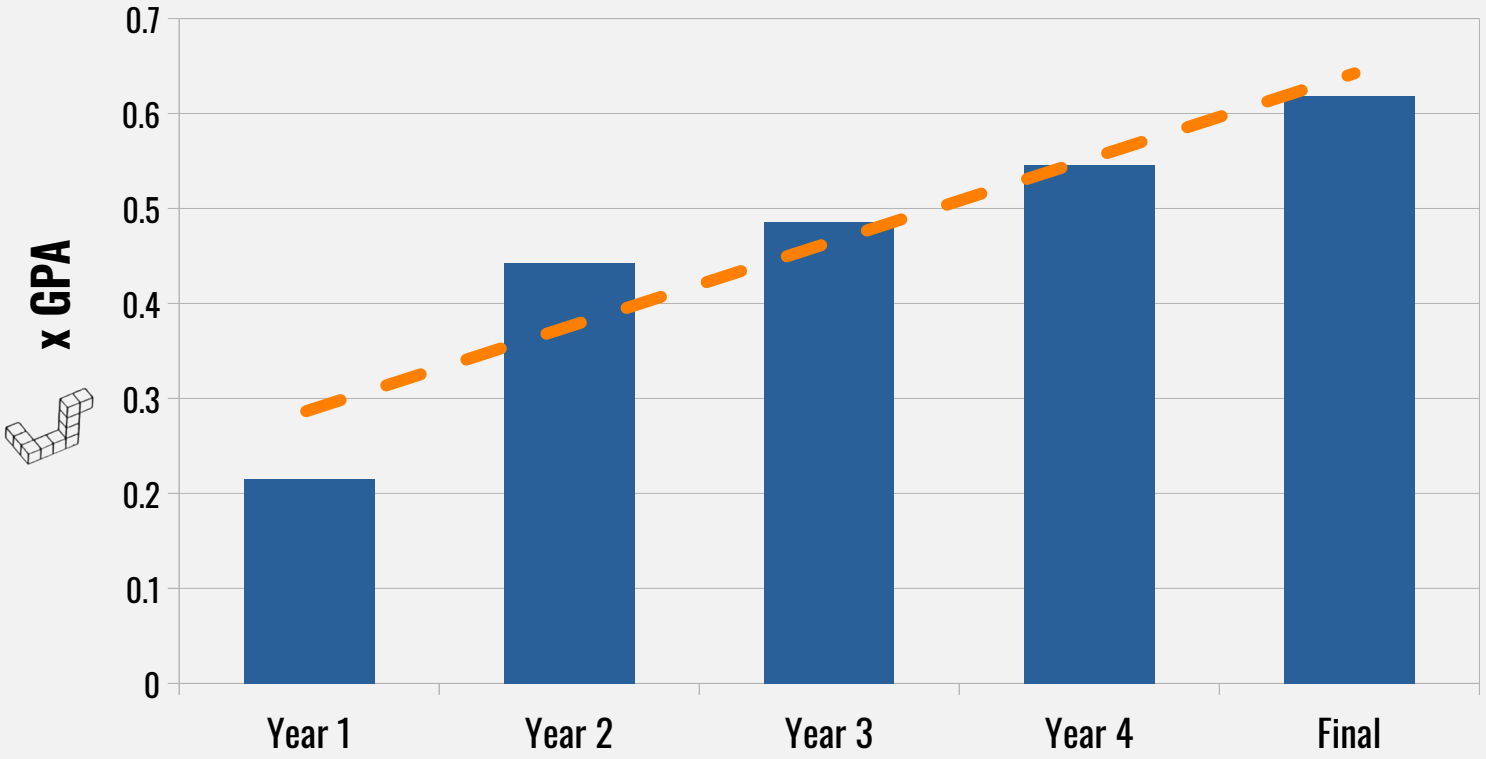
(…) **small to null effects are found on far-transfer** measures (i.e., fluid intelligence, attention, language, and mathematics).

/ Giovanni Sala & Fernand Gobet */*

Investigating sex differences, cognitive effort, strategy, and performance on a computerised version of the mental rotations test via eye tracking, Adam J. Toth et al., 2019



Relationships between an Early-Stage Spatial Skills Test and Final CS Degree Outcomes, Jack Parkinson et al., 2021



```
int main() {
  for (auto n=0; n <= 100; ++n) {
    std::print("{}: {}\n", n, *fizz_buzz(n));
  }
}

auto fizz_buzz = make_chain (
  [](auto n) { return n % 15 == 0 ? std::optional("FizzBuzz"s) : std::nullopt; },
  [](auto n) { return n % 3 == 0 ? std::optional("Fizz"s) : std::nullopt; },
  [](auto n) { return n % 5 == 0 ? std::optional("Buzz"s) : std::nullopt; },
  [](auto n) { return std::optional(std::to_string(n)); }
);
```

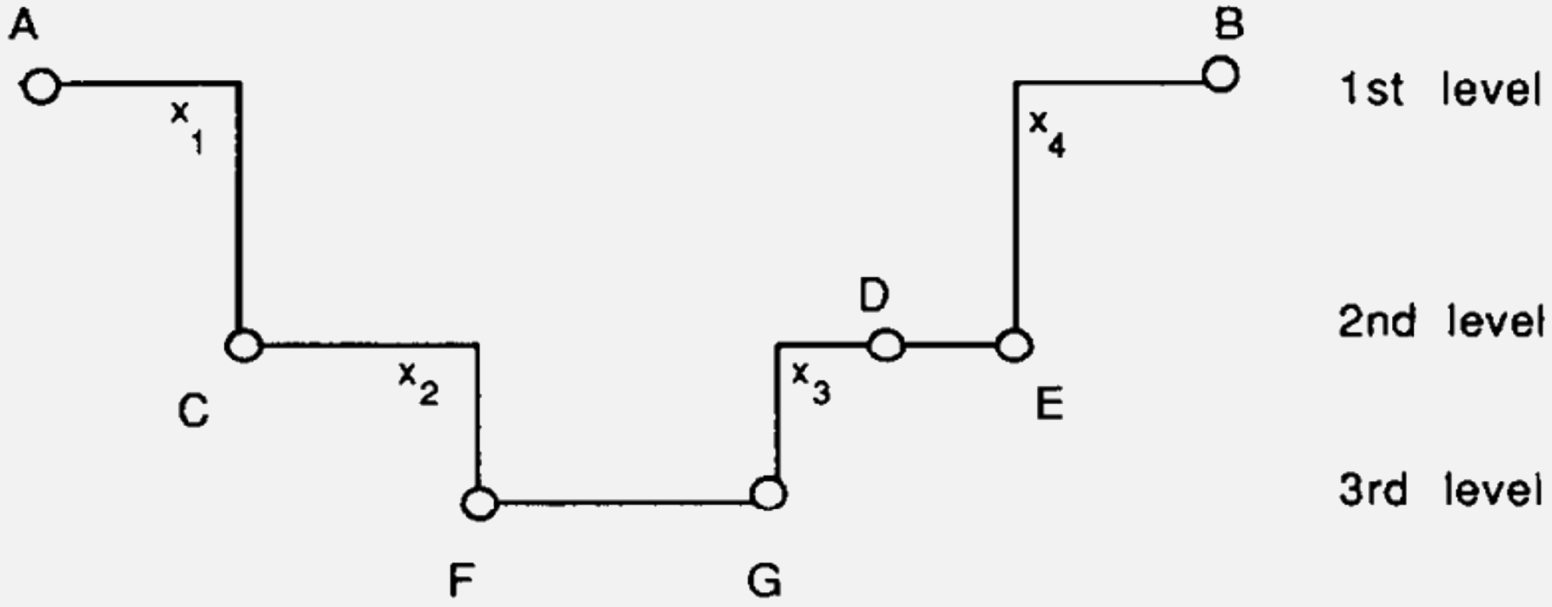


Figure 5 Landscape model of program comprehension

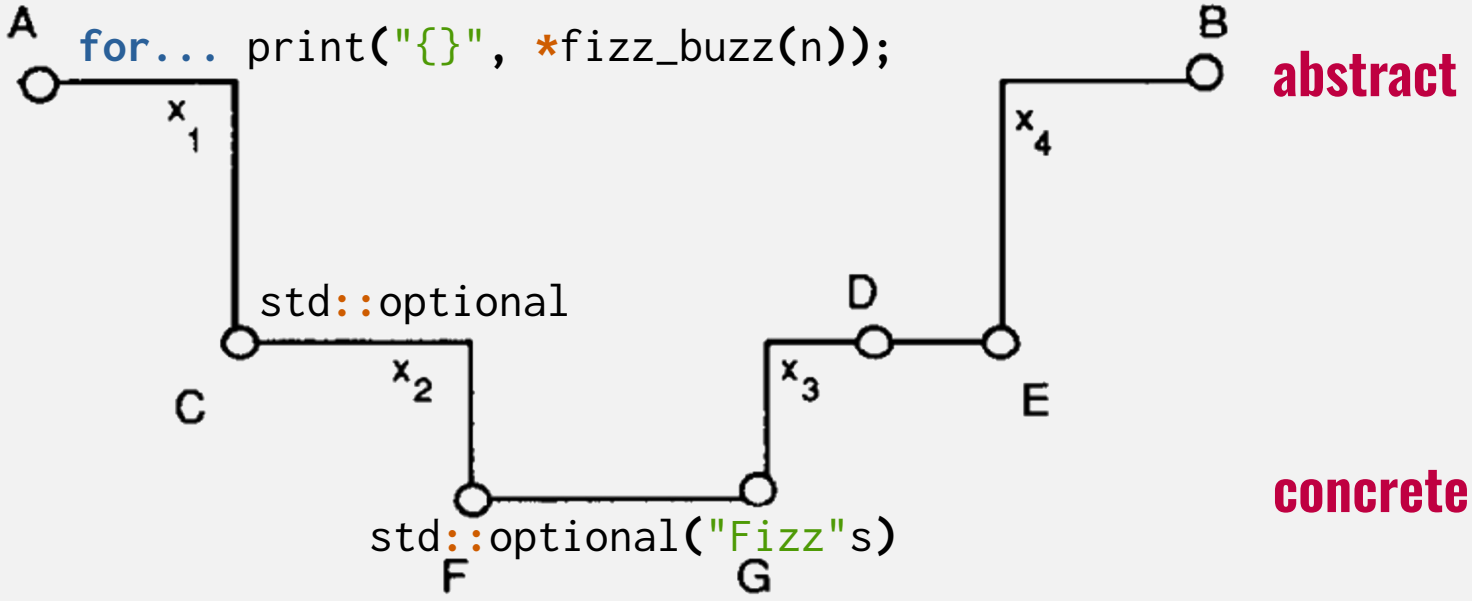


Figure 5 Landscape model of program comprehension

```cpp
for (auto n=0; n <= 100; ++n) {
  std::print("{}: {}\n", n, *fizz_buzz(n));
}

auto fizz_buzz = make_chain (
  [](auto n) { return n % 15 == 0 ? std::optional("FizzBuzz"s) : std::nullopt; },
  [](auto n) { return n % 3 == 0 ? std::optional("Fizz"s) : std::nullopt; },
  [](auto n) { return n % 5 == 0 ? std::optional("Buzz"s) : std::nullopt; },
  [](auto n) { return std::optional(std::to_string(n)); }
);

auto make_chain = []<typename...Fs>(Fs&&...fs){
  return [...fs=std::forward<Fs>(fs)](auto&&...args) -> decltype(auto) {
    using Res_t = std::common_type_t<std::invoke_result_t<Fs, decltype(args)...>...>;
    Res_t res{};
    ((res) || ((res = std::invoke(fs, args...))  || ...));
    return res;
  };
};
```

```
auto make_chain = []<typename...Fs>(Fs&&...fs){
  return [...fs=std::forward<Fs>(fs)](auto&&...args) -> decltype(auto) {
    using Res_t = std::common_type_t<std::invoke_result_t<Fs, decltype(args)...>...>;
    Res_t res{};
    ((res) || ((res = std::invoke(fs, args...))  || ...));
    return res;
  };
};
```

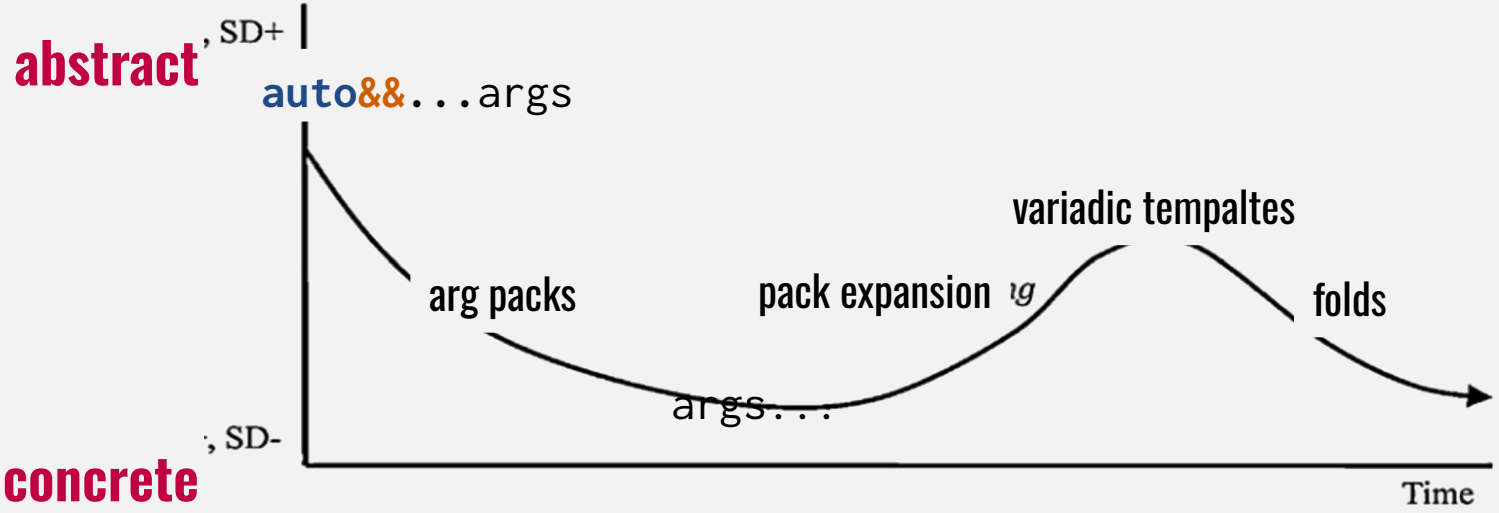
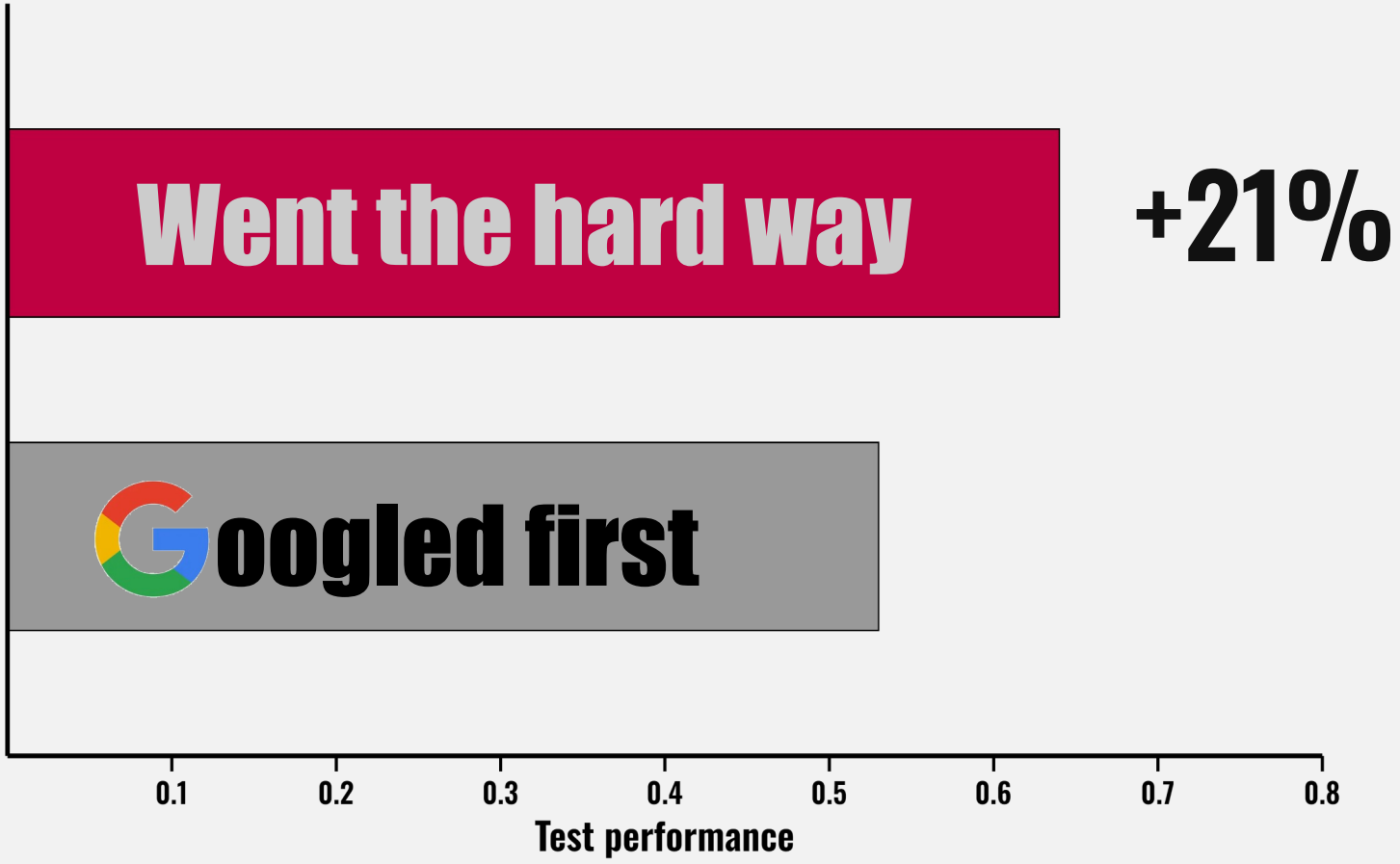


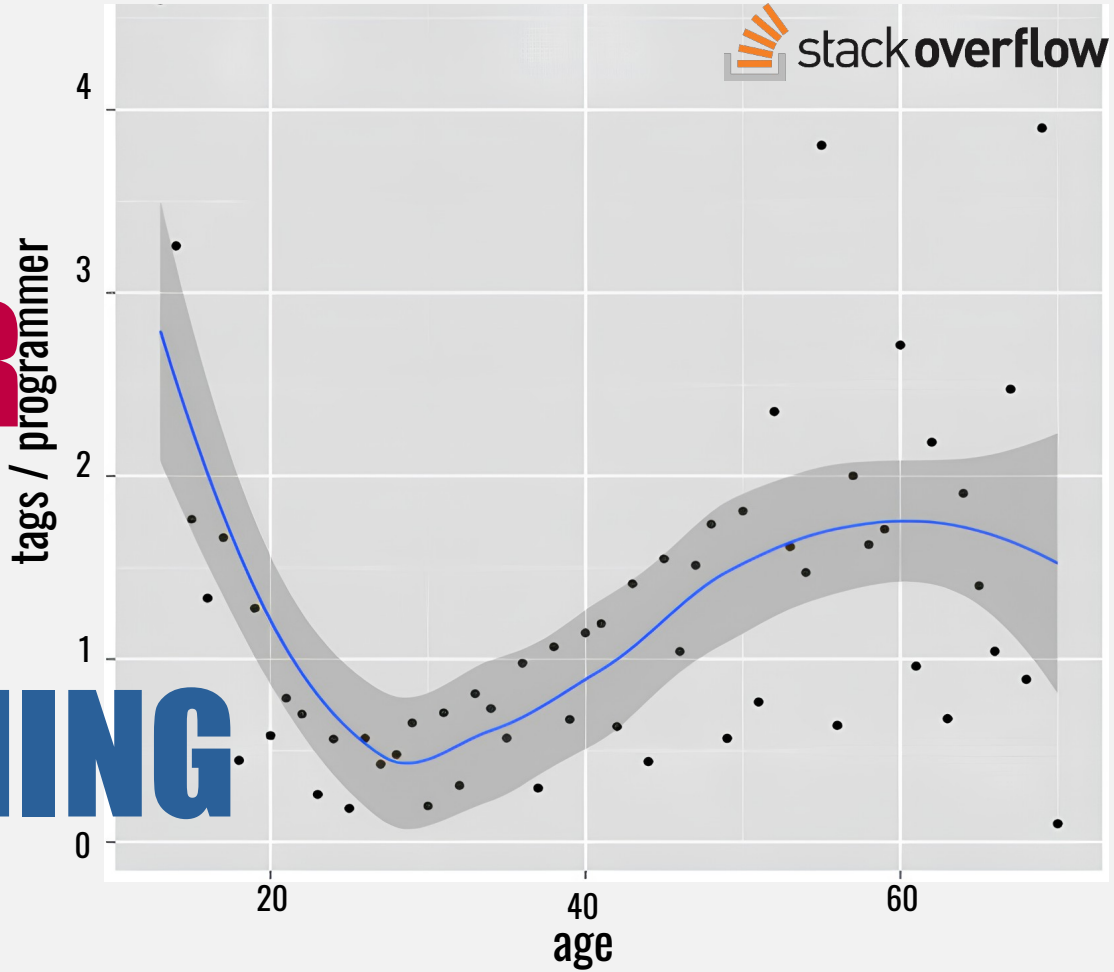
Fig. 7. Example of a semantic wave in History teaching.

Answer First or Google First? Using the Internet in ways that Enhance, not Impair, One's Subsequent Retention of Needed Information, Saskia Giebi et al., 2020



Correlates of programmer efficacy and their link to experience: a combined EEG and eye-tracking study,
Norman Peitek et al., 2022

Is Programming Knowledge Related To Age?, Patrick Morrison et al., 2013



YOU NEVER STOP LEARNING

```
auto process_buggy(auto text, auto pattern, auto callback) {
  auto pattern_size = pattern.size();
  auto pattern_hash = Hash{pattern};
  auto last_index = text.size() - pattern_size;

  auto text_hash = Hash{text.substr(0, pattern_size)};

  for (auto i = 0; i <= last_index; ++i) {
    if (pattern_hash == text_hash) {
      if (pattern == text.substr(i, pattern_size)) {
        callback(i);
      }
    }

   if (i != last_index) {
     text_hash.update(text[i - pattern_size]);
   }
}
```

Expert-novice knowledge of computer programming at different levels of abstraction, Nong Ye et al., 2007

```
for (auto i = 0; i <= last_index; ++i) {...}
```

Physical

`for` is a keyword

Logical

This is a `for`-i loop

Functional

The loop is used to shift a text window

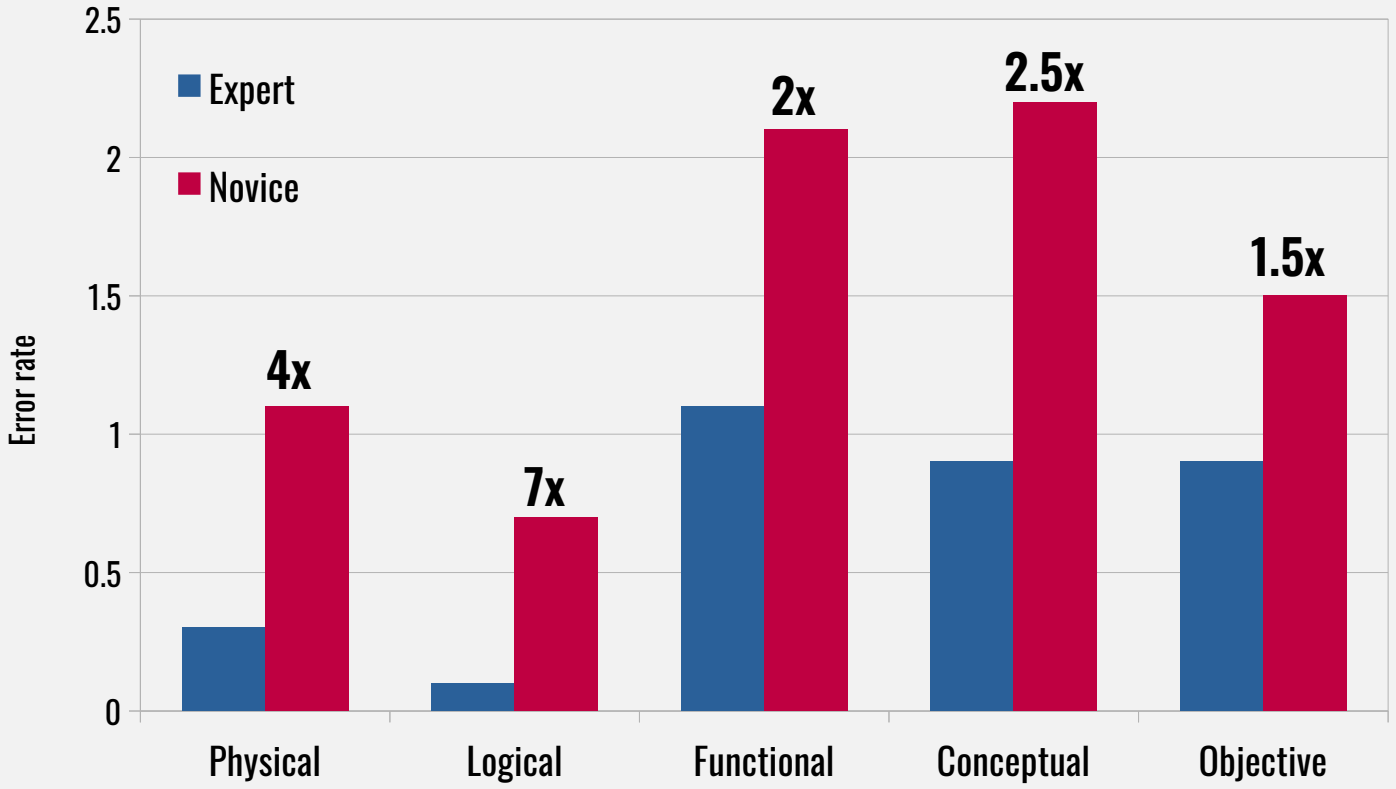
Conceptual

Sliding-window hash-based comparison

Objective

Part of finding a pattern in a text

Expert-novice knowledge of computer programming at different levels of abstraction, Nong Ye et al., 2007



Just fine:

```
std::size_t last_index{...};

for (std::size_t i = 0; i <= last_index; ++i) {}
```

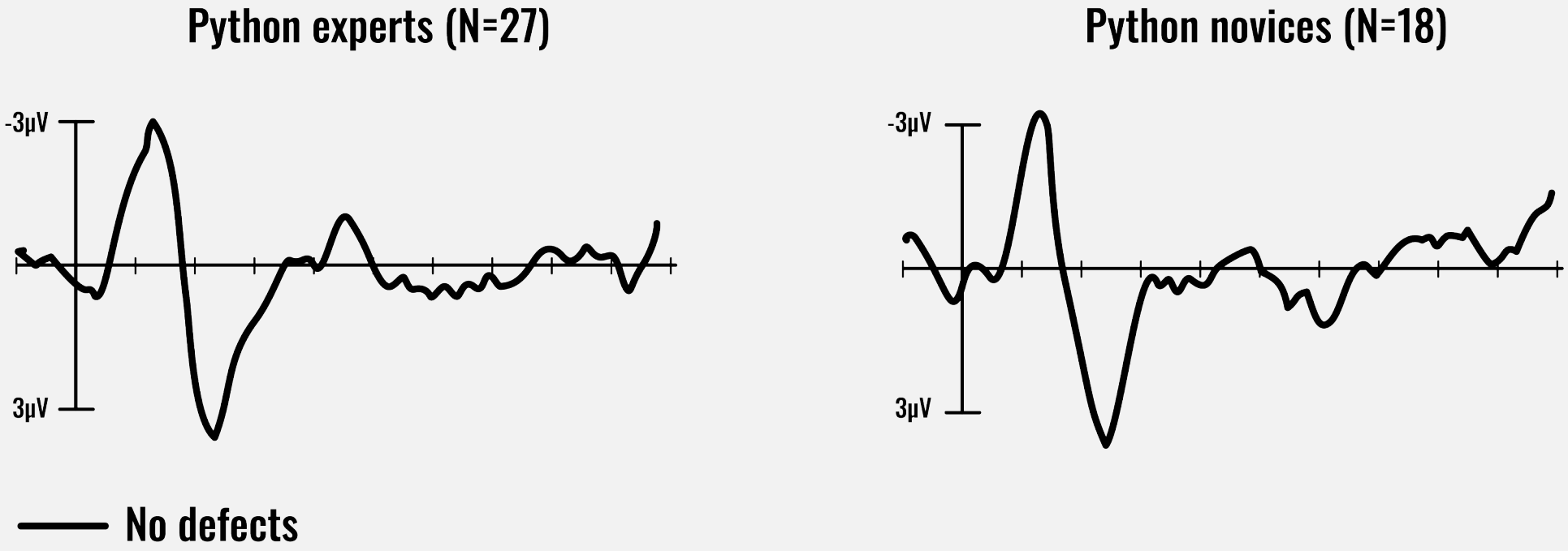
Sematically implausible (meaning violations):

```
for (std::size_t i = 0; has_pink_hair(i); ++i) {}
```

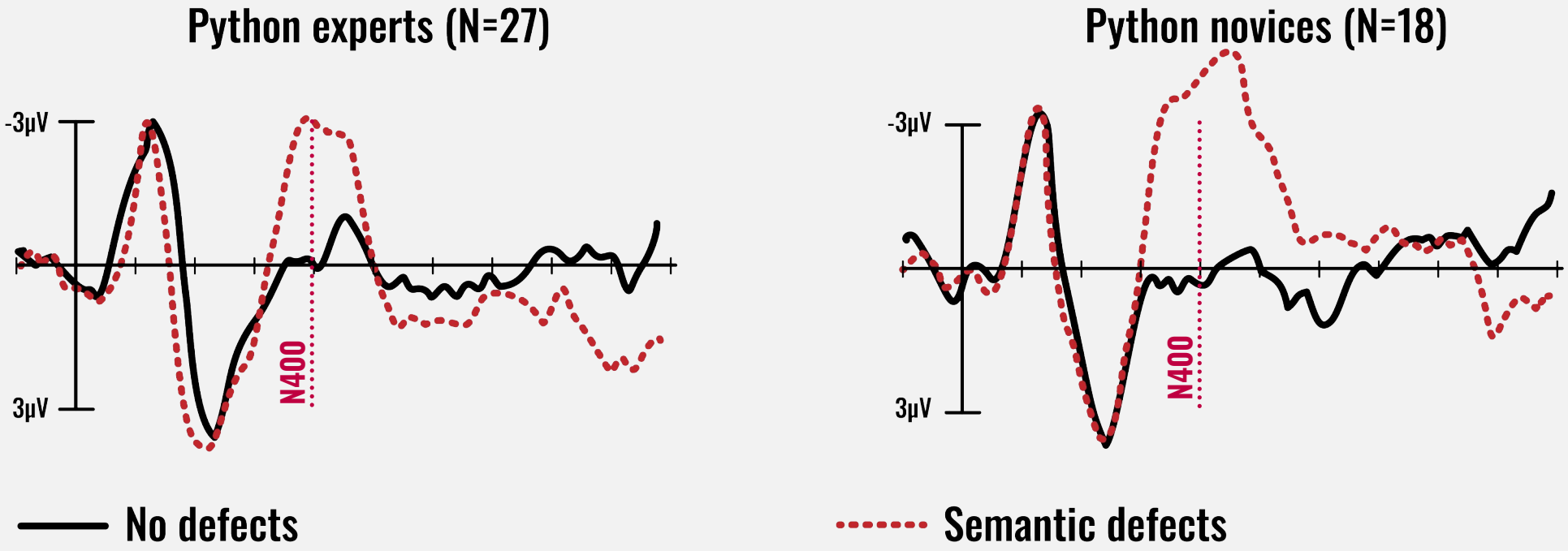
Syntactically suspicius (grammar violations):

```
for (int i = 0; i <= last_index; ++i) {}
```

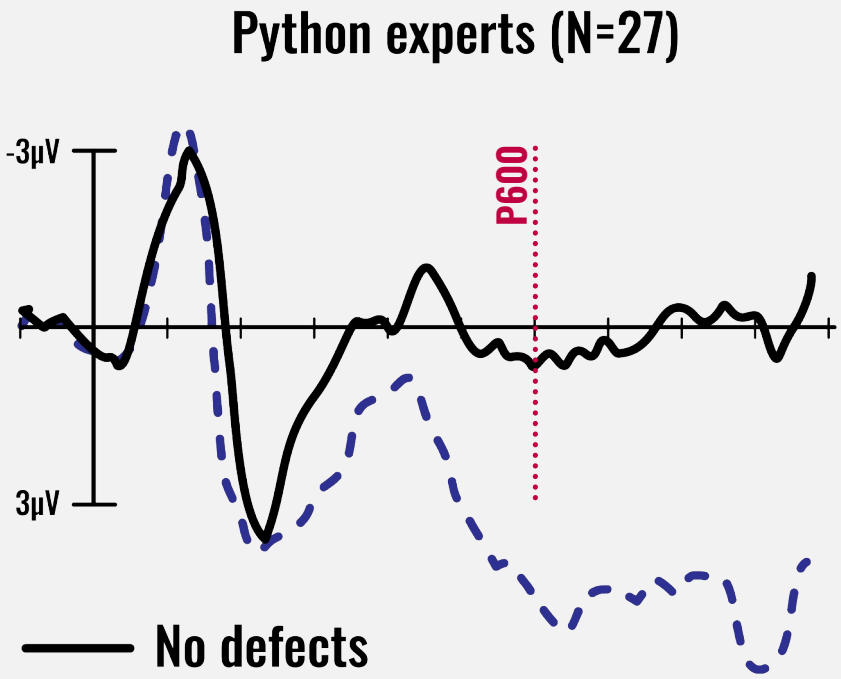
Computer programmers show distinct, expertise-dependent brain responses to violations in form and meaning when reading code, Chu-Hsuan Kuo et al., 2024

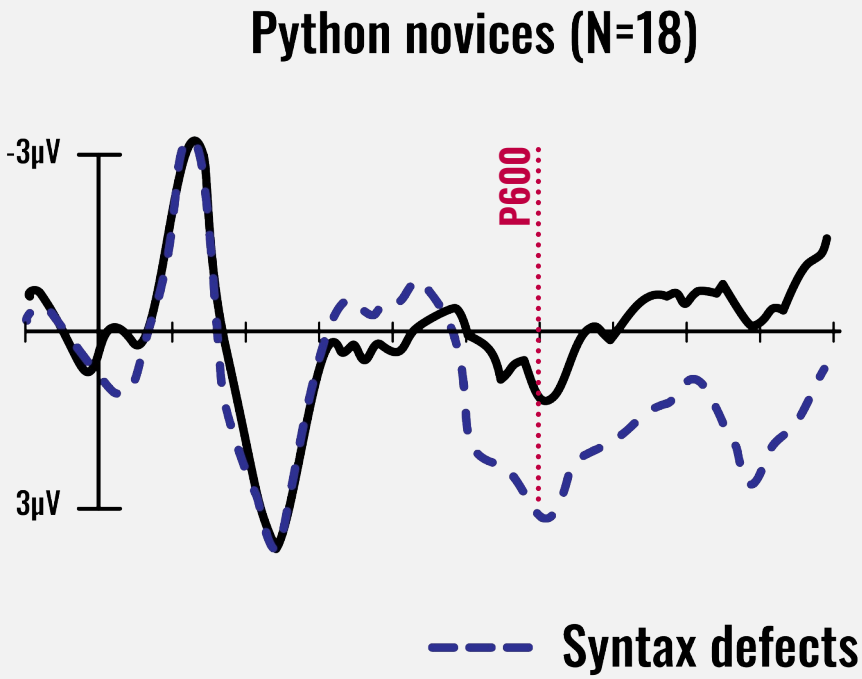


Computer programmers show distinct, expertise-dependent brain responses to violations in form and meaning when reading code, Chu-Hsuan Kuo et al., 2024



Computer programmers show distinct, expertise-dependent brain responses to violations in form and meaning when reading code, Chu-Hsuan Kuo et al., 2024





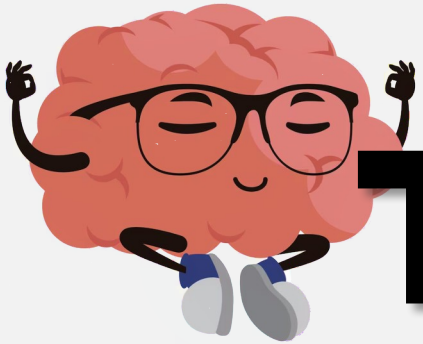
To become better at programming, read & write code, A LOT

Learning to program well is a never-ending journey

EVERYBODY was once a BEGINNER

```
READY.
LIST

1 DATA "ON THE KITCHEN TABLE"
2 DATA "MIND YOUR OWN BUSINESS"
3 DATA "NEXT WEEK OR THE WEEK AFTER"
4 DATA "NO, YOU REALLY SHOULDN'T"
5 READ A$, B$, C$, D$

10 X = RND(-TI)

20 PRINT "WHAT IS YOUR QUESTION"
25 INPUT QUES$

30 R = INT(RND(1)*4)+1
35 IF R = 1 THEN PRINT A$
40 IF R = 2 THEN PRINT B$
45 IF R = 3 THEN PRINT C$
50 IF R = 4 THEN PRINT D$

55 INPUT "MORE QUESTIONS"; AGAIN$
60 IF AGAIN$ = "YES" THEN GOTO 20
100 PRINT "GOODBYE!"
```

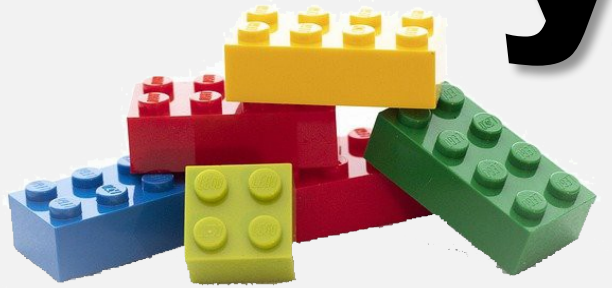

BE KIND
& GENTLE



Thank you!

Dawid Zalewski

zaldawid@gmail.com | linkedin.com/in/dawid-zalewski



Beginner's mind, expert's mind

how we think about, read, write and learn to code

Dawid Zalewski

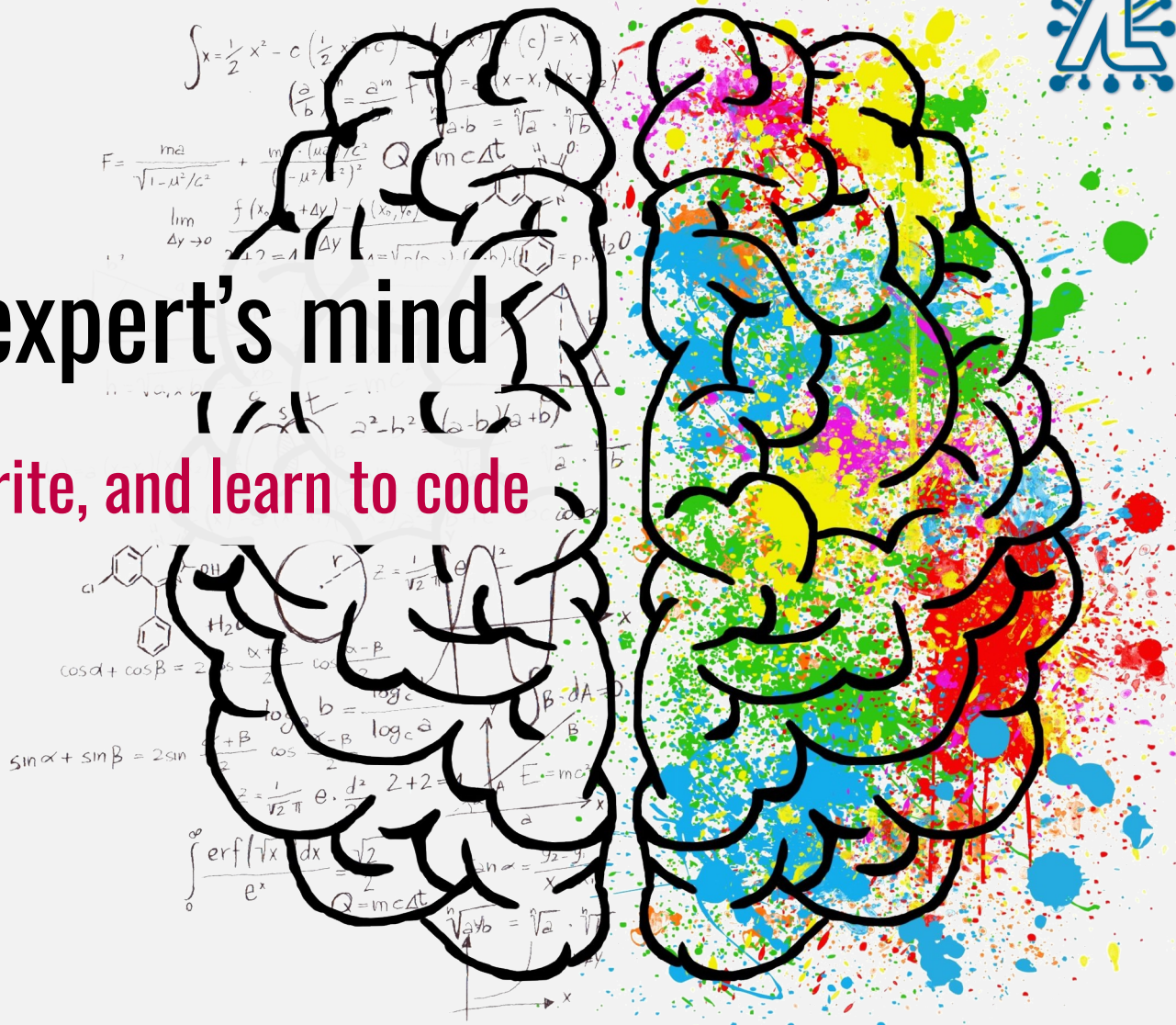
Beginner's mind, expert's mind

how we think about, read, write, and learn to code

Dawid Zalewski

Beginner's mind, expert's mind

how we think about, read, write, and learn to code



Dawid Zalewski

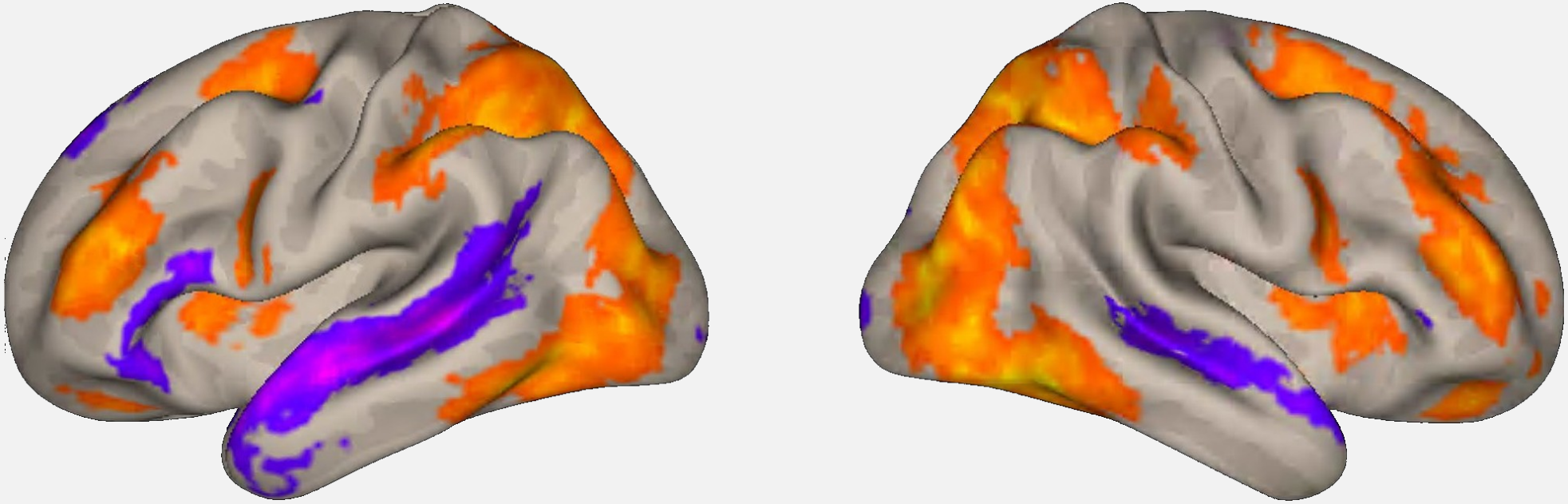
```cpp
constexpr auto a = "on the kitchen table";
constexpr auto b = "mind your own business";
constexpr auto c = "next week or the week after";
constexpr auto d = "no, you really shouldn't";

std::mt19937 rng(std::random_device{}());
std::uniform_int_distribution<int> uni(1, 4);
std::string question;
do {
    std::print("what is your question");
    std::getline(std::cin, question);

    auto r = uni(rng);
    if (r == 1) std::print(a);
    if (r == 2) std::print(b);
    if (r == 3) std::print(c);
    if (r == 4) std::print(d);

    std::print("more questions?");
    std::getline(std::cin, question);
} while (question == "yes");
```

```cpp
std::array answers = {
    "on the kitchen table",
    "mind your own business",
    "next week or the week after",
    "no, you really shouldn't"
};

auto generator = std::mt19937{std::random_device{}()};
std::string question;

do
{
    std::print("what is your question\n");
    std::getline(std::cin, question);

    std::ranges::sample(answers, std::ostream_iterator<std::string>(std::cout, "\n"), 1,
generator);

    std::print("more questions?");
    std::getline(std::cin, question);
} while (question == "yes");
```

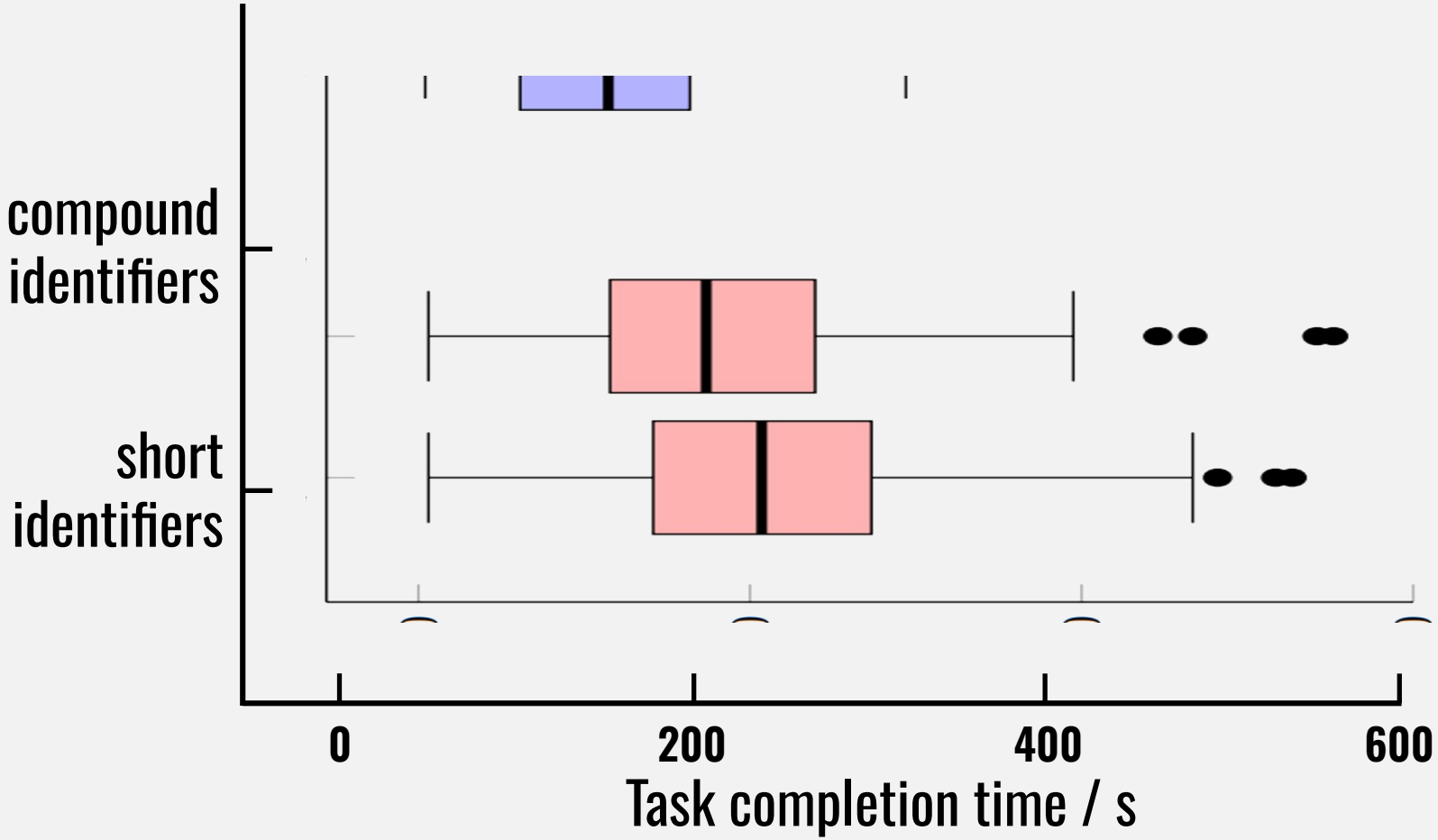
Comprehension of computer code relies primarily on domain-general executive brain regions, Anna A. Ivanova et al. 2020

Code comprehension > Text comprehension

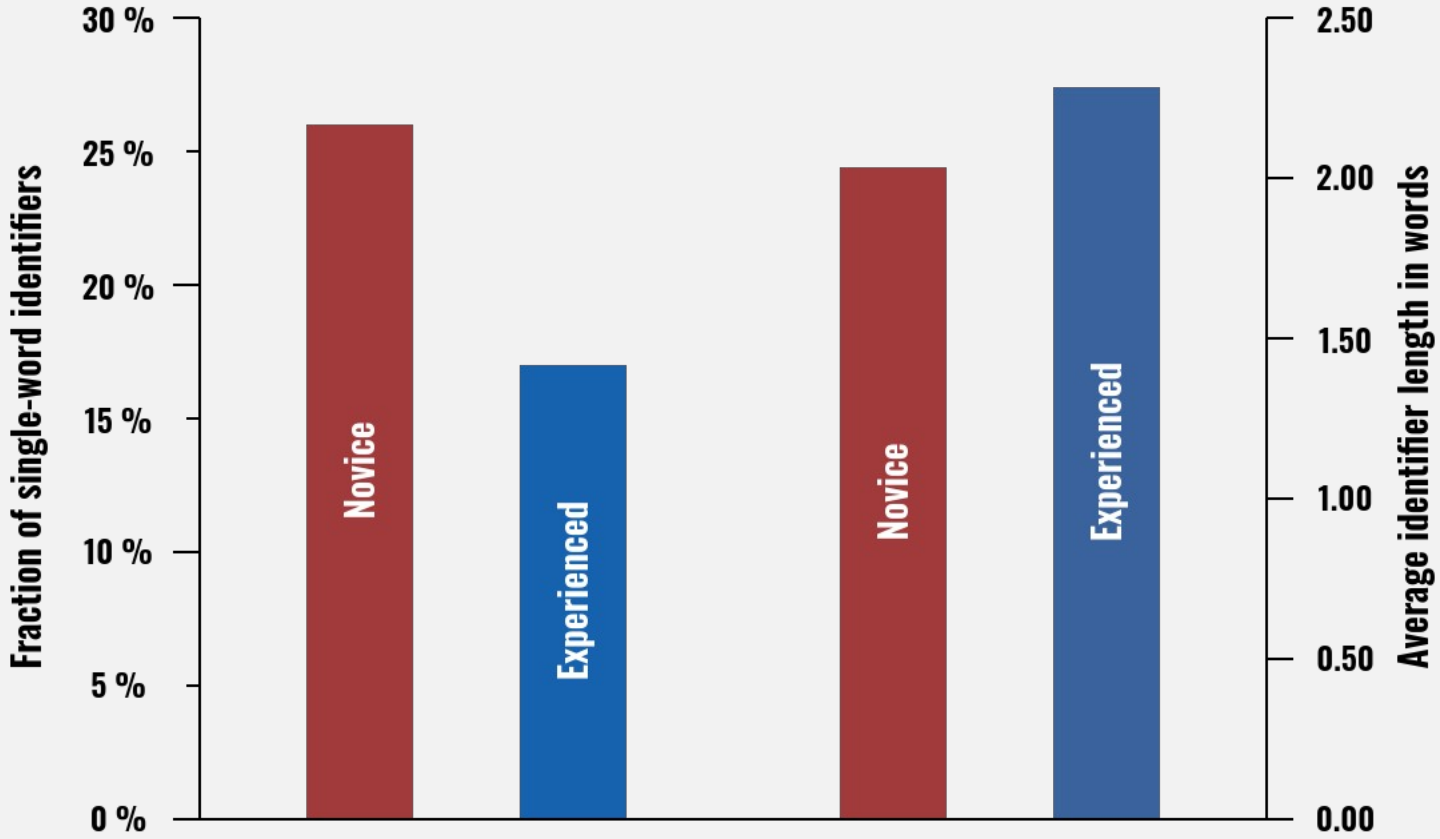
Text comprehension > Code comprehension



Descriptive compound identifier names improve source code comprehension, Andrea Schankin et al., 2018



How Developers Choose Names, Dror Feitelson et al., 2021



```
auto i = 1u;
for (;;) {
  auto check1 = [](auto i){auto s{0u}; while(i){ s += i%10u; i/=10u; } return s%3u; }(i);
  auto check2 = i * 0xcccccccdu > 0x33333333u;
  auto check3 = check1 + check2;
  auto check4 = i - (check1 * check2);

  if (check3 == 0)
    std::print("FizzBuzz\n");
  if (check1 == 0 && check2)
    std::print("Fizz\n");
  ff (!check2)
    if (check1 & 0xf7u)
      std::print("{}, Buzz\n", i);
  if (!(!(check4 > i) && !(check4 < i)))
    std::print("{}\n", i);

  if (100 < (i+=1)) break;
}
```

```
qsort :: Ord a => [a] -> [a]
qsort []     = []
qsort (p:xs) = (qsort smaller) ++ [p] ++ (qsort greater)
    where
        smaller = filter (< p) xs
        greater = filter (>= p) xs
```

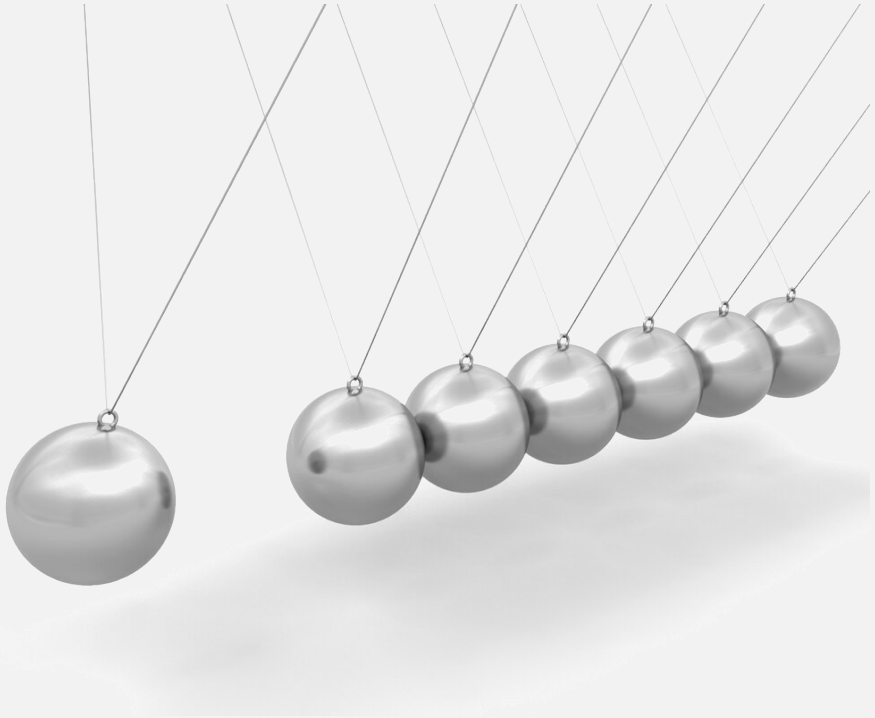
```
Q←{1≥≠ω:ω ◇ S←{α≠⍨α αα ω} ◇ ω((∇<S)⍤=S⍤(∇>S))ω⌷⍨?≠ω}
```

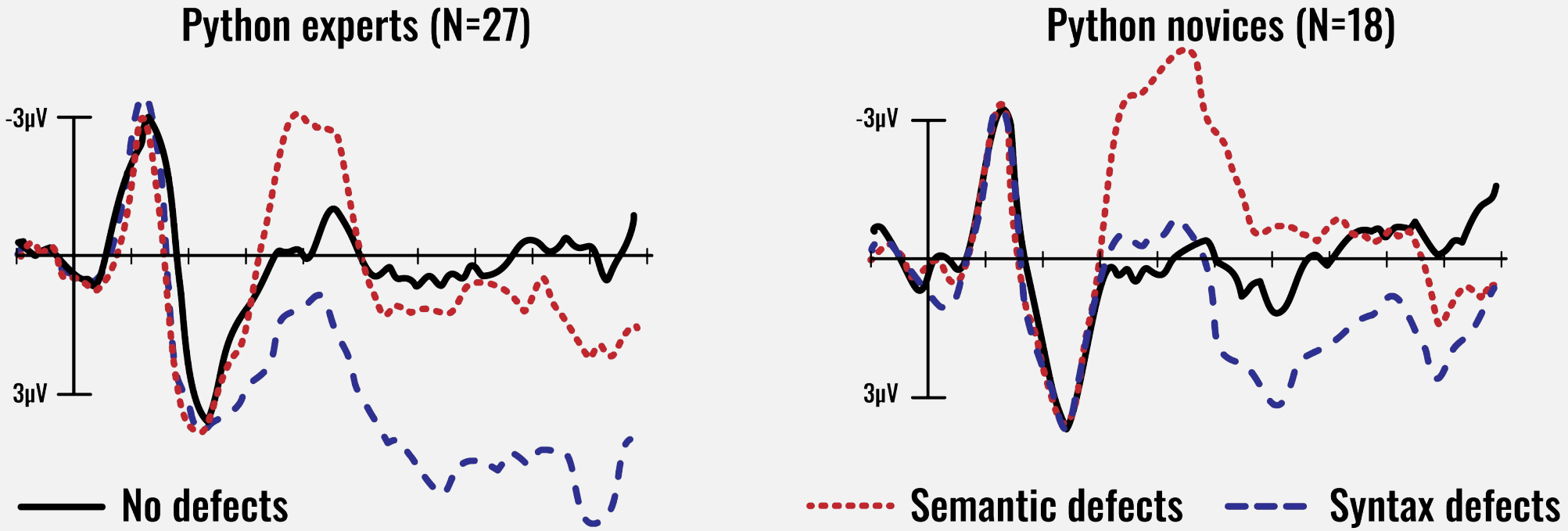
Learning second and subsequent programming languages is **easier** than learning a first programming language because many concepts and constructs are shared.

/ Jean Scholtz & Susan Wiedenbeck */*

(...) programs (...) are strongly influenced by (...) other languages they know. **This prevents them from taking full advantage of the capabilities of the new language**.

/ Jean Scholtz & Susan Wiedenbeck */*




Python experts (N=27)
-3μV
3μV
No defects
Python novices (N=18)
-3μV
3μV
Semantic defects
Syntax defects