



LIGHTNING TALKS

ACCU2018

Wednesday 11th April

electricity is really just
organized lightning.

— George Carlin



THE RULES

subjects are open!
five minutes (max)
have fun



Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

The background of the image consists of rich red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed area behind them. Each side of the curtain is held back by a gold-colored rope with a matching tassel. The lighting is dramatic, highlighting the texture of the fabric and the metallic sheen of the tassels.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



OM NOM NOM



**STEPHEN
HAWKING**



**SAUSSAGE
ROLLS**

Guy Davidson - *A year in diversity*

Jon Jagger - *FizzBuzz in the C pre-processor*

Frances Buontempo - *Here beis a dragons*

Peter Sommerlad - *APRIL*

Cezary Bloch - *Shaderator*

Seb Rose - *Literal Misdirection*

Anna-Jayne - *Two Small Corrections*

Björn Fahlner - *My favourite memory leak*

Dom Davis - *Putting the away into go*

Gail Ollis - *Care of Magical Creatures*

Steve Love - *</rant>*

Pete Goodliffe - *The New C++ Interview*

A YEAR IN DIVERSITY

12 months by J Guy Davidson

@hatcat01

TAKING A PLEDGE



@hatcat01

GRILL THE COMMITTEE



AHA, AH HAHAHA AH HA



J. Guy Davidson @hatcat01 · Jul 14



Why isn't there a C++ diversity group called #Include ?



OOOPSY...



J. Guy Davidson

@hatcat01



Why isn't there a C++ diversity group called [#Include](#) ?

10:23 PM - 14 Jul 2017

16 Retweets 82 Likes



7



16



82



Add another Tweet



Kate Gregory @gregcons · 17 Jul 2017



Replying to @hatcat01

Let's start it. First get-together at [@CppCon](#); another at [@meetingcpp](#)



3



1



9



@hatcat01

WELL, KATE GREGORY SAID I SHOULD DO THIS, SO...



@hatcat01

LET'S ORGANISE



@hatcat01

MEETING C++

WHAT MAKES A GOOD C++ PROGRAMMER?

Intellectual acuity

Rigour

Perseverance



@hatcat01
_atcat01

SLACK CHANNEL FILLS UP



PRIVATE DISCORD GROUP



@hatcat01

READY FOR YOU TO JOIN IN

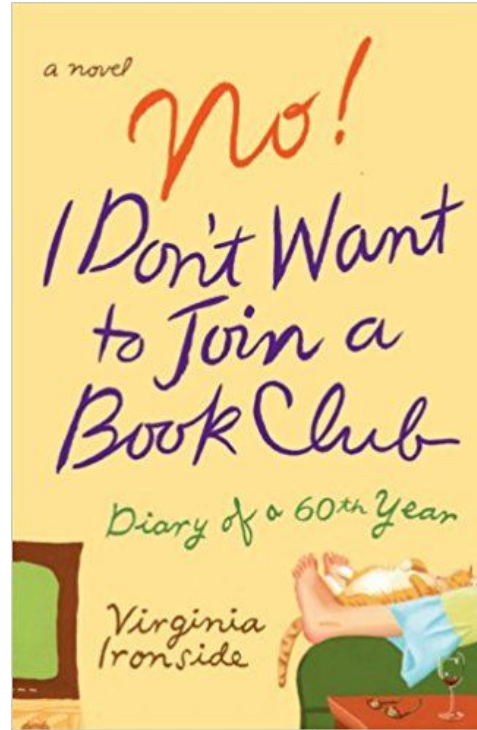


@hatcat01

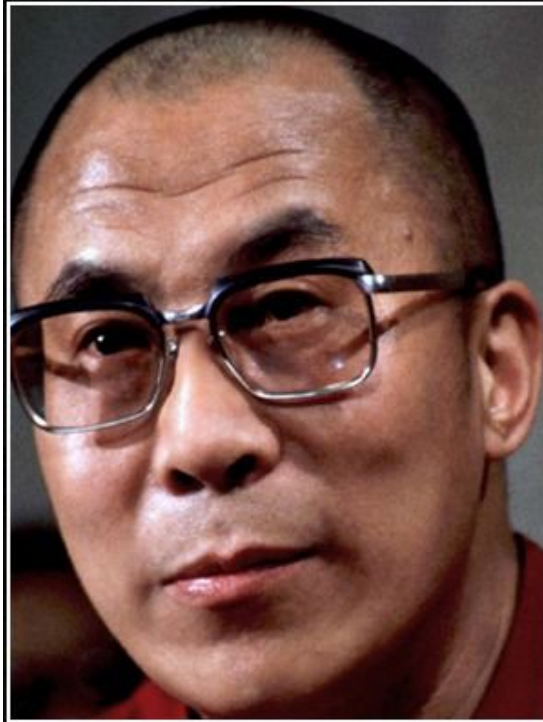
CODE OF CONDUCT



OR DON'T JOIN IN, THAT'S FINE



DISAGREEMENT IS FINE ALSO

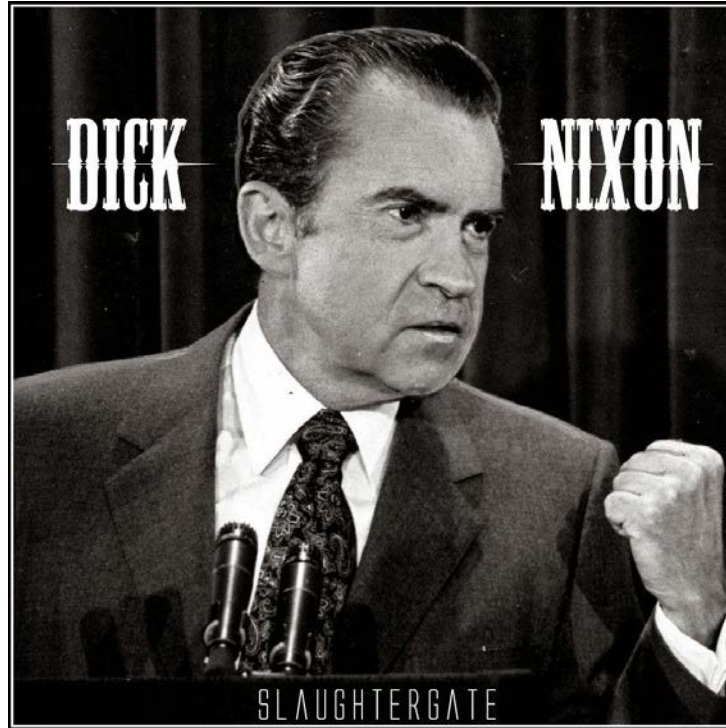


Disagreement is something normal.

— Dalai Lama —

AZ QUOTES

TROLLING AND SEA-LIONING THOUGH...



@hatcat01

LOOK! HERE! NOW!

<https://www.includecpp.org/>

[@include_cpp](#)

<https://discord.gg/Sy9r7P9>

<https://github.com/include-cpp>

MAKE THE POOL BIGGER



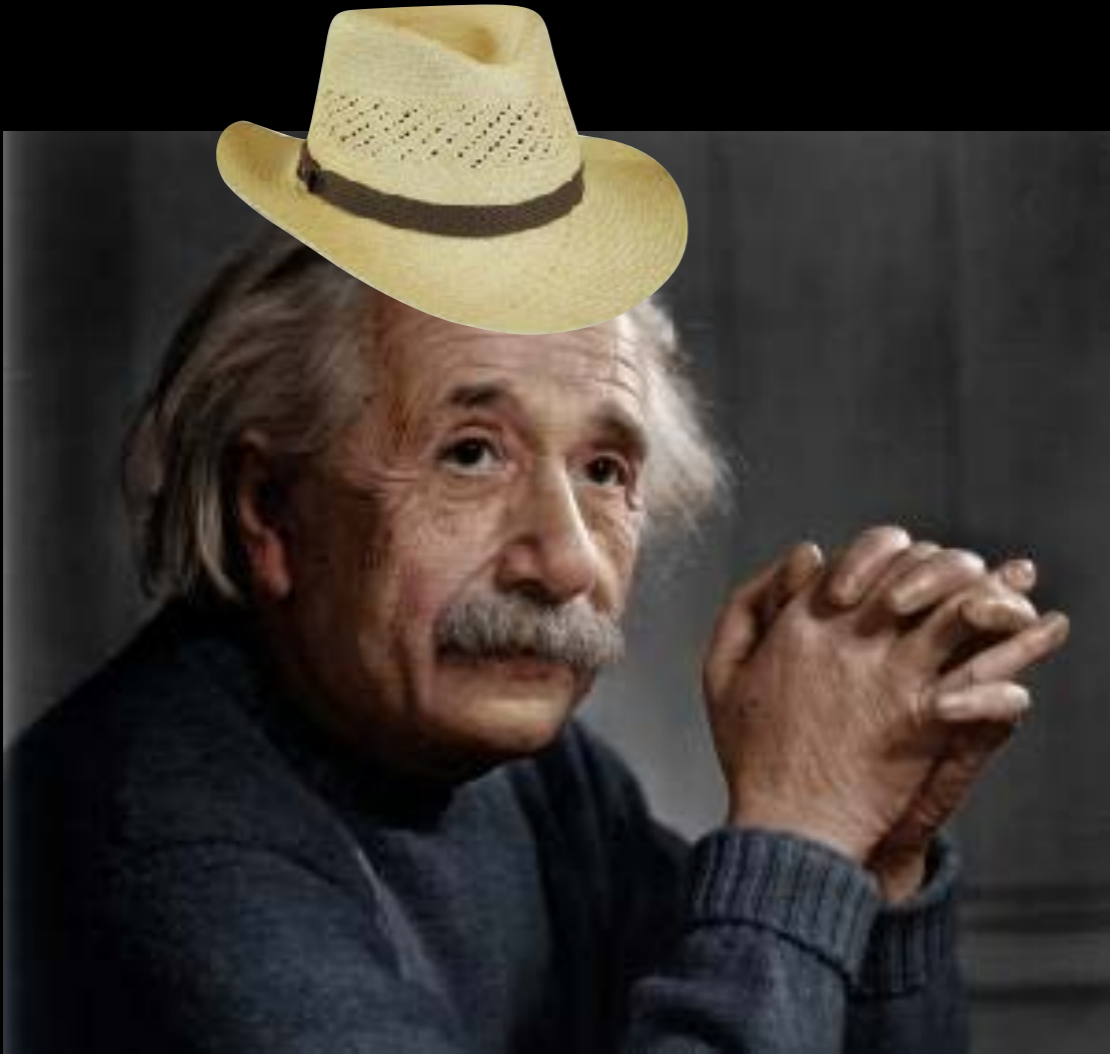
@hatcat01

****THANK YOU!****

You know what to do.

The background of the image consists of rich, red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed stage area. Each side of the curtain is held back by a decorative gold-colored tassel. The lighting is dramatic, highlighting the texture and folds of the fabric.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



LOOKS TASTY



**ALBERT
EINSTEIN**



EGGS BENEDICT

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

FizzBuzz in the C PreProcessor

jon@jaggersoft.com



This was just for fun!
I'm not suggesting you
actually use the
pre-processor like this...

Kudos to Paul Fultz II

<https://github.com/pfultz2/Cloak>

```
EVAL(REPEAT(100, INC, 0))
```



```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
```



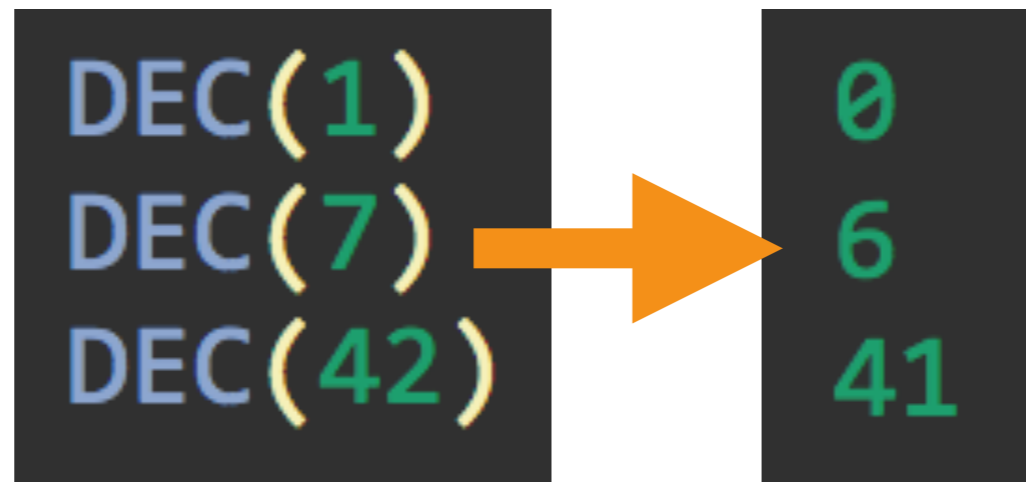
```
EVAL (REPEAT(100, INC, 0))
```



```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
```

```
#define REPEAT(n, macro, i) \  
    WHEN(n) \  
    ( \  
        macro( i ), \  
        REPEAT( DEC(n), macro, macro(i) ) \  
    )
```

```
#define REPEAT(n, macro, i) \  
    WHEN(n) \  
    ( \  
        macro( i ), \  
        REPEAT( DEC(n), macro, macro(i) ) \  
    )
```



```
#define DEC(x)      CAT(DEC_, x)

#define DEC_0      0
#define DEC_1      0
#define DEC_2      1
#define DEC_3      2
#define DEC_4      3
#define DEC_5      4
#define DEC_6      5
#define DEC_7      6
#define DEC_8      7
#define DEC_9      8
#define DEC_10     9
#define DEC_11     10
#define DEC_12     11
#define DEC_13     12
#define DEC_14     13
#define DEC_15     14
#define DEC_16     15
```

```
#define REPEAT(n, macro, i) \  
    WHEN(n) \  
    ( \  
        macro( i ), \  
        REPEAT( DEC(n), macro, macro(i) ) \  
    )
```

```
WHEN(0)(42)  
WHEN(1)(42)
```



```
42
```

```
#define WHEN(c)          IF(c)(EXPAND, EAT)
#define IF(c)           IIF(BOOL(c))
#define EXPAND(...)    __VA_ARGS__
#define EAT(...)

#define IIF(c)          CAT(IIF_, c)
#define IIF_0(t, ...)  __VA_ARGS__
#define IIF_1(t, ...)  t
```

```
WHEN(7)(42) --> IF(7)(EXPAND, EAT)(42)
--> IIF(BOOL(7))(EXPAND, EAT)(42)
--> IIF(1)(EXPAND, EAT)(42)
--> IIF_1(EXPAND, EAT)(42)
--> EXPAND(42)
--> 42
```

```
WHEN(0)(42) --> IF(0)(EXPAND, EAT)(42)
--> IIF(BOOL(0))(EXPAND, EAT)(42)
--> IIF(0)(EXPAND, EAT)(42)
--> IIF_0(EXPAND, EAT)(42)
--> EAT(42)
-->
```

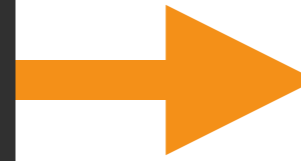
```
#define WHEN(c)      IF(c)(EXPAND, EAT)
#define IF(c)       IIF(BOOL(c))
#define EXPAND(...) __VA_ARGS__
#define EAT(...)

#define IIF(c)      CAT(IIF_, c)
#define IIF_0(t, ...) __VA_ARGS__
#define IIF_1(t, ...) t
```



```
#define BOOL(x) COMPL(NOT(x))  
  
#define COMPL(b) CAT(COMPL_, b)  
#define COMPL_0 1  
#define COMPL_1 0
```

```
BOOL(0)  
BOOL(1)  
BOOL(42)
```



```
0  
1  
1
```

```
#define BOOL(x) COMPL(NOT(x))
```

```
#define COMPL(b) CAT(COMPL_, b)
```


```
#define COMPL_0 1
```

```
#define COMPL_1 0
```


```
#define NOT(x) CHECK(CAT(NOT_, x))
#define NOT_0 ~, 1,

#define CHECK(...) CHECK_N(__VA_ARGS__, 0,)
#define CHECK_N(_, n, ...) n
```

```
NOT(42) --> CHECK(CAT(NOT_, 42))
--> CHECK(NOT_42)
--> CHECK_N(NOT_42, 0,)
```



```
NOT(0) --> CHECK(CAT(NOT_, 0))
--> CHECK(NOT_0)
--> CHECK(~, 1, )
--> CHECK_N(~, 1, 0,)
```



```
#define REPEAT(n, macro, i) \  
    WHEN(n) \  
    ( \  
        macro( i ), \  
        REPEAT( DEC(n), macro, macro(i) ) \  
    )
```

Macros cannot be recursive...

```

#define REPEAT(n, macro, i) \
    WHEN(n) \
    ( \
        macro( i ), \
        OBSTRUCT(REPEAT_INDIRECT) () \
        ( \
            DEC(n), macro, macro(i) \
        ) \
    )
#define REPEAT_INDIRECT() REPEAT

```

A macro can be tricked into being recursive!

```

#define OBSTRUCT(id) id DEFER(EMPTY)()
#define DEFER(id) id EMPTY()
#define EMPTY()

```

```
EVAL(REPEAT(100, INC, 0))
```



```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
```

```
#define INC(x)    CAT(INC_, x)

#define INC_0    1
#define INC_1    2
#define INC_2    3
#define INC_3    4
#define INC_4    5
#define INC_5    6
#define INC_6    7
#define INC_7    8
#define INC_8    9
#define INC_9    10
#define INC_10   11
#define INC_11   12
#define INC_12   13
#define INC_13   14
```

```
EVAL(REPEAT(100,CYCLE3,0))
```



```
1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2,
```



```
#define CYCLE3(f) CYCLE3_ ## f
```

```
#define CYCLE3_0 1
```

```
#define CYCLE3_1 2
```

```
#define CYCLE3_2 3
```

```
#define CYCLE3_3 1
```

```
EVAL(REPEAT(100, CYCLE5, 0))
```



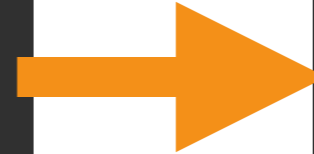
```
1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 1, 2, 3, 4,
```



```
#define CYCLE5(f) CYCLE5_ ## f  
  
#define CYCLE5_0 1  
#define CYCLE5_1 2  
#define CYCLE5_2 3  
#define CYCLE5_3 4  
#define CYCLE5_4 5  
#define CYCLE5_5 1
```



```
NTH_ARG(1, a)  
NTH_ARG(2, a, b)  
NTH_ARG(3, a, b, c)
```



```
a  
b  
c
```

```
#define NTH_ARG(N,...) NTH_ARG_ ## N(__VA_ARGS__)

#define NTH_ARG_1(N,...) N
#define NTH_ARG_2(_1,N,...) N
#define NTH_ARG_3(_1,_2,N,...) N
#define NTH_ARG_4(_1,_2,_3,N,...) N
#define NTH_ARG_5(_1,_2,_3,_4,N,...) N
#define NTH_ARG_6(_1,_2,_3,_4,_5,N,...) N
#define NTH_ARG_7(_1,_2,_3,_4,_5,_6,N,...) N
#define NTH_ARG_8(_1,_2,_3,_4,_5,_6,_7,N,...) N
#define NTH_ARG_9(_1,_2,_3,_4,_5,_6,_7,_8,N,...) N
#define NTH_ARG_10(_1,_2,_3,_4,_5,_6,_7,_8,_9,N,...) N
#define NTH_ARG_11(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,N,...) N
#define NTH_ARG_12(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,N,...) N
#define NTH_ARG_13(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,N,...) N
#define NTH_ARG_14(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,N,...) N
#define NTH_ARG_15(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,N,...) N
#define NTH_ARG_16(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,N,...) N
#define NTH_ARG_17(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,_16,...) N
#define NTH_ARG_18(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,_16,...) N
#define NTH_ARG_19(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,_16,...) N
#define NTH_ARG_20(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,_16,...) N
#define NTH_ARG_21(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,_16,...) N
#define NTH_ARG_22(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,_16,...) N
#define NTH_ARG_23(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,_16,...) N
#define NTH_ARG_24(_1,_2,_3,_4,_5,_6,_7,_8,_9,_10,_11,_12,_13,_14,_15,_16,...) N
```

1 #define NTL_A00_01... NTL_A00_01... N
2 #define NTL_A00_02... N
3 #define NTL_A00_03... N
4 #define NTL_A00_04... N
5 #define NTL_A00_05... N
6 #define NTL_A00_06... N
7 #define NTL_A00_07... N
8 #define NTL_A00_08... N
9 #define NTL_A00_09... N
10 #define NTL_A00_10... N
11 #define NTL_A00_11... N
12 #define NTL_A00_12... N
13 #define NTL_A00_13... N
14 #define NTL_A00_14... N
15 #define NTL_A00_15... N
16 #define NTL_A00_16... N
17 #define NTL_A00_17... N
18 #define NTL_A00_18... N
19 #define NTL_A00_19... N
20 #define NTL_A00_20... N
21 #define NTL_A00_21... N
22 #define NTL_A00_22... N
23 #define NTL_A00_23... N
24 #define NTL_A00_24... N
25 #define NTL_A00_25... N
26 #define NTL_A00_26... N
27 #define NTL_A00_27... N
28 #define NTL_A00_28... N
29 #define NTL_A00_29... N
30 #define NTL_A00_30... N
31 #define NTL_A00_31... N
32 #define NTL_A00_32... N
33 #define NTL_A00_33... N
34 #define NTL_A00_34... N
35 #define NTL_A00_35... N
36 #define NTL_A00_36... N
37 #define NTL_A00_37... N
38 #define NTL_A00_38... N
39 #define NTL_A00_39... N
40 #define NTL_A00_40... N
41 #define NTL_A00_41... N
42 #define NTL_A00_42... N
43 #define NTL_A00_43... N
44 #define NTL_A00_44... N
45 #define NTL_A00_45... N
46 #define NTL_A00_46... N
47 #define NTL_A00_47... N
48 #define NTL_A00_48... N
49 #define NTL_A00_49... N
50 #define NTL_A00_50... N
51 #define NTL_A00_51... N
52 #define NTL_A00_52... N
53 #define NTL_A00_53... N
54 #define NTL_A00_54... N
55 #define NTL_A00_55... N
56 #define NTL_A00_56... N
57 #define NTL_A00_57... N
58 #define NTL_A00_58... N
59 #define NTL_A00_59... N
60 #define NTL_A00_60... N
61 #define NTL_A00_61... N
62 #define NTL_A00_62... N
63 #define NTL_A00_63... N
64 #define NTL_A00_64... N
65 #define NTL_A00_65... N
66 #define NTL_A00_66... N
67 #define NTL_A00_67... N
68 #define NTL_A00_68... N
69 #define NTL_A00_69... N
70 #define NTL_A00_70... N
71 #define NTL_A00_71... N
72 #define NTL_A00_72... N
73 #define NTL_A00_73... N
74 #define NTL_A00_74... N
75 #define NTL_A00_75... N
76 #define NTL_A00_76... N
77 #define NTL_A00_77... N
78 #define NTL_A00_78... N
79 #define NTL_A00_79... N
80 #define NTL_A00_80... N
81 #define NTL_A00_81... N
82 #define NTL_A00_82... N
83 #define NTL_A00_83... N
84 #define NTL_A00_84... N
85 #define NTL_A00_85... N
86 #define NTL_A00_86... N
87 #define NTL_A00_87... N
88 #define NTL_A00_88... N
89 #define NTL_A00_89... N
90 #define NTL_A00_90... N
91 #define NTL_A00_91... N
92 #define NTL_A00_92... N
93 #define NTL_A00_93... N
94 #define NTL_A00_94... N
95 #define NTL_A00_95... N
96 #define NTL_A00_96... N
97 #define NTL_A00_97... N
98 #define NTL_A00_98... N
99 #define NTL_A00_99... N
100 #define NTL_A00_100... N

```
#define FIZZ_BUZZ(n) \  
  FIZZ_BUZZ_ \  
  ( \  
    n, \  
    NTH_ARG(n, EVAL(REPEAT(100,CYCLE3,0))), \  
    NTH_ARG(n, EVAL(REPEAT(100,CYCLE5,0))) \  
  )  
  
#define FIZZ_BUZZ_(n,f,b)          FIZZ_BUZZ_PRIMITIVE(n,f,b)  
#define FIZZ_BUZZ_PRIMITIVE(n,f,b) FIZZ_BUZZ_ ## f ## _ ## b (n)
```

```
#define FIZZ_BUZZ_1_1(n) n
#define FIZZ_BUZZ_1_2(n) n
#define FIZZ_BUZZ_1_3(n) n
#define FIZZ_BUZZ_1_4(n) n
#define FIZZ_BUZZ_1_5(_) Buzz

#define FIZZ_BUZZ_2_1(n) n
#define FIZZ_BUZZ_2_2(n) n
#define FIZZ_BUZZ_2_3(n) n
#define FIZZ_BUZZ_2_4(n) n
#define FIZZ_BUZZ_2_5(_) Buzz

#define FIZZ_BUZZ_3_1(_) Fizz
#define FIZZ_BUZZ_3_2(_) Fizz
#define FIZZ_BUZZ_3_3(_) Fizz
#define FIZZ_BUZZ_3_4(_) Fizz
#define FIZZ_BUZZ_3_5(_) FizzBuzz
```

Testing!

```
# -E == preprocess only  
# -P == dont show #line's  
gcc -std=c99 -E -P fizz_buzz.tests.h
```

```
#include "fizz_buzz.h"  
  
#define ASSERT(e,a)    ASSERT_(e,a)  
#define ASSERT_(e,a)  STR(ASSERT_ ## e ## _ ## a)  
  
#include ASSERT(FIZZ_BUZZ_1_EQUALS, FIZZ_BUZZ(1))  
#include ASSERT(FIZZ_BUZZ_3_EQUALS, FIZZ_BUZZ(3))  
#include ASSERT(FIZZ_BUZZ_5_EQUALS, FIZZ_BUZZ(5))  
#include ASSERT(FIZZ_BUZZ_15_EQUALS, FIZZ_BUZZ(15))
```

...

fizz_buzz.tests.h

Create one empty *file* per assertion...

```
ASSERT_FIZZ_BUZZ_1_EQUALS_1
```

```
ASSERT_FIZZ_BUZZ_3_EQUALS_Fizz
```

```
ASSERT_FIZZ_BUZZ_5_EQUALS_Buzz
```

```
ASSERT_FIZZ_BUZZ_15_EQUALS_FizzBuzz
```

```
...
```

failing

```
#include ASSERT(FIZZ_BUZZ_1_EQUALS, FIZZ_BUZZ(99))
```

tests.h

```
fizz_buzz.tests.h:6:1: fatal error: ASSERT_FIZZ_BUZZ_1_EQUALS_Fizz: No such file or directory  
#include ASSERT(FIZZ_BUZZ_1_EQUALS, FIZZ_BUZZ(99))  
  ^~~~~~  
compilation terminated.
```

stderr

passing

```
stdout
```


A fly fisherman wearing a green cap and a dark jacket is standing in a river, holding a fishing rod. The river is surrounded by dense evergreen trees and a blue sky with white clouds. The text is overlaid on the image in a white, bold, sans-serif font, enclosed in a white dashed rectangular border.

**50% off
consultancy near
an in-season
salmon river**

jon@jaggersoft.com

The background of the image consists of rich, red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed stage area. Each side of the curtain is held back by a decorative gold-colored tassel. The lighting is dramatic, highlighting the texture and folds of the fabric.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



MY FAVOURITE!

NIKOLA TESLA



TIRAMISU

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

Here ~~be~~is a dragons

Magic!

Frances Buontempo

@fbuontempo

Lindenmayer Systems

- L-systems
 - <http://python3.codes/drawing-fractals-with-lindenmayer-systems/>
- Recursion
- Grammars
- Trees, ferns...
- Self-similar
 - fractals



Dragon t-shirt

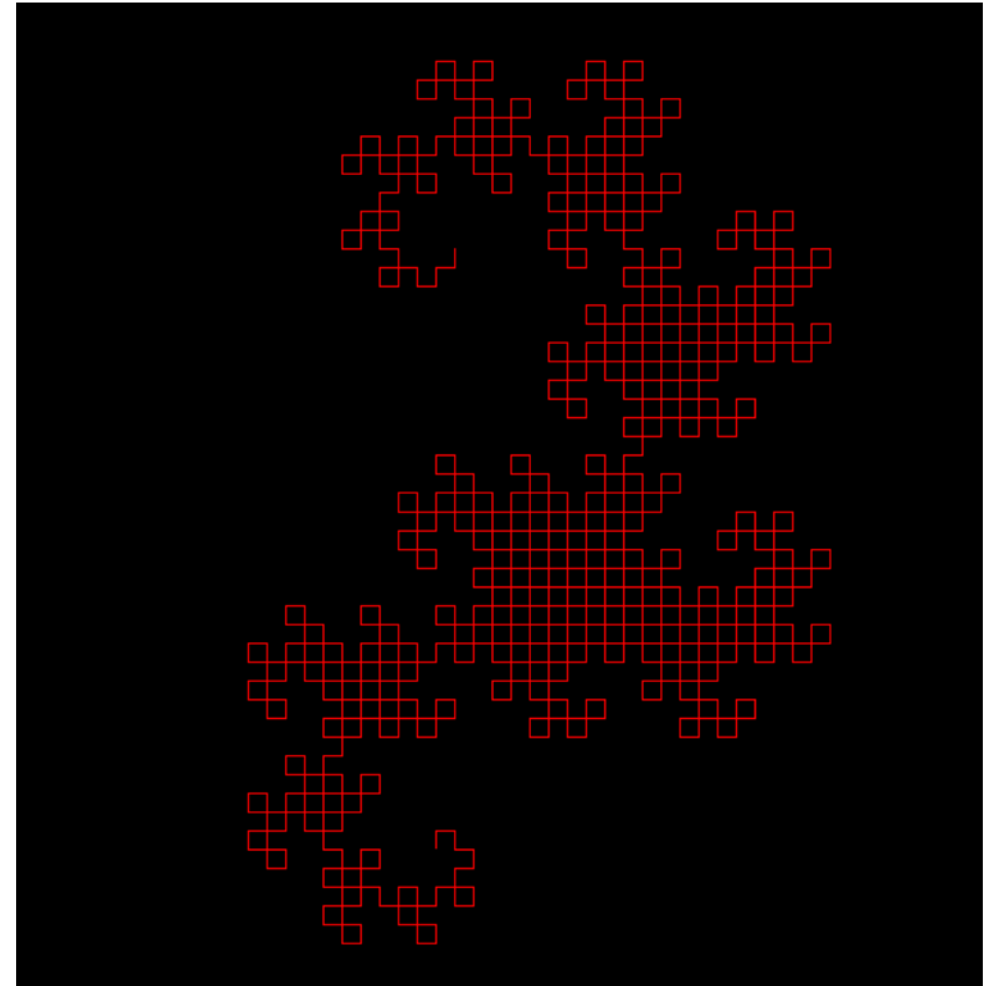
```
from turtle import*


def X(n):
    if n>0:    L("X+YF+",n)
def Y(n):
    if n>0:    L("-FX-Y",n)

def L(s,n):
    for c in s:
        if c=='-': lt(90)
        elif c=='+': rt(90)
        elif c=='X': X(n-1)
        elif c=='Y': Y(n-1)
        elif c=='F': fd(12)

bgcolor('black')
pencolor('red')
up()
goto(-20, 120)
down()
X(10)
hideturtle()

mainloop()
```



The background of the image consists of rich, red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed area behind them. Each side of the curtain is held back by a decorative gold-colored tassel. The lighting is dramatic, highlighting the texture and folds of the fabric.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



TASTY



ISAAC NEWTON



NECTARINES

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

{The Problem

```
std::unique_ptr<char, decltype(std::free) *>  
    t_copy { strdup(t), std::free };  
// takes the address of std::free!  
// function to function pointer conversion
```

Is Unspecified in the std!

see also:

http://stackoverflow.com/questions/27440953/stdunique_ptr-for-c-functions-that-need-free/

Why?

```
std::unique_ptr<std::FILE, decltype(&std::fclose)> fp(std::fopen("demo.txt", "r"),
                                                    &std::fclose);
if(fp) // fopen could have failed; in which case fp holds a null pointer
    std::cout << (char)std::fgetc(fp.get()) << '\n';
```

Thou Shalt Not Specialize `std` Function Templates!



Document #: WG21 P0551R3
Date: 2018-03-16
Project: JTC1.22.32 Programming Language C++
Audience: LWG
Reply to: Walter E. Brown <webrown.cpp@gmail.com>



The details: *addressable functions*

6 Let F denote a standard library function ([global.functions]), a standard library static member function, or an instantiation of a standard library function template. Unless F is designated an *addressable function*, the behavior of a C++ program is unspecified (possibly ill-formed) if it explicitly or implicitly attempts to form a pointer to F . [Note: Possible means of forming such pointers include application of the unary `&` operator ([expr.unary.op]), `addressof` ([specialized.addressof]), or a function-to-pointer standard conversion ([conv.func]). — end note] Moreover, the behavior of a C++ program is unspecified (possibly ill-formed) if it attempts to form a reference to F or if it attempts to form a pointer-to-member designating either a standard library non-static member function ([member.functions]) or an instantiation of a standard library member function template.

Exception (so far): `iostream` manipulators

A Workaround Proposal?

P0984R0 - All (*)()-Pointers Replaced by Ideal Lambdas

Document Number:	P0984R0
Date:	2018-04-01
Project:	Programming Language C++
Audience:	EWG/LEWG
Target:	C++20



The closure type for a non-generic *lambda-expression* with no *lambda-capture* whose constraints (if any) are satisfied is called a *Ideal Lambda*. An *Ideal Lambda* has a conversion function to pointer to function with C++ language linkage(10.5) having the same parameter and return types

call operator template specialization. An *Ideal Lambda* furthermore defines an overload for the unary `operator&()` that returns the result of the said conversion to function pointer. [*Note*: That operator overload guarantees that existing code bases that invalidly take the address of a standard library function continue to work as expected. — *end note*]

Names that are defined as functions in C shall be defined as `functions constexpr inline auto variables` initialized from an *Ideal Lambda* in the C++ standard library, unless the C++ standard defines overloads of said function. In that case the names defined as functions in C shall be defined as functions. ^[1]

More...


P0984R0 - All (*)()-Pointers Replaced by Ideal Lambdas

~~It is unspecified whether any~~ All non-overloaded non-template non-member functions in the C++ standard library shall be defined as constexpr inline auto variables initialized from an Ideal Lambda. For the purpose of this standard these variables are called FOOL (Function ObsOleted by Lambda). [*Note: This mechanism allows many wrong programs that take the address of a standard library function to conform to this standard. — end note*] It is unspecified wether any overloaded or templated non-member functions are defined as inline(10.1.6).



How must I do it now?

```
const std::string filename = "./hello1.txt";
auto close=[](auto fd){::close(fd);};
{
    auto file = unique_resource(::open(filename.c_str(),
                                     O_CREAT|O_RDWR, 06)
                              , close);
    ::write(file.get(), "Hello World!\n", 12u);
    ASSERT(file.get() != -1);
}
```



Calls are OK!

Solution for unique_ptr:


```
struct free_deleter{
    template <typename T>
    void operator()(T *p) const {
        std::free(const_cast<std::remove_const_t<T>*>(p));
    }
};
template <typename T>
using unique_C_ptr=std::unique_ptr<T, free_deleter>;

static_assert(sizeof(char *)==sizeof(unique_C_ptr<char>), "");
// compiles!
```

Wrap the call in a class!

lambdas/decltype(lambda) works in the future

Stay tuned for FOOL!

The background of the image consists of rich, red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed area behind them. Each side of the curtain is held back by a decorative gold-colored tassel. The lighting is dramatic, highlighting the texture and folds of the fabric.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



FFFEED ME!



MARIE CURIE



**VERY HOT
CURRY**

Guy Davidson - A year in diversity

Jon Jagger - FizzBuzz in the C pre-processor

Frances Buontempo - Here beis a dragons

Peter Sommerlad - APRIL

Cezary Bloch - Shaderator

Seb Rose - Literal Misdirection

Anna-Jayne - Two Small Corrections

Björn Fahlner - My favourite memory leak

Dom Davis - Putting the away into go

Gail Ollis - Care of Magical Creatures

Steve Love - </rant>

Pete Goodliffe - The New C++ Interview

Shaderator Compute Shader debugging and Unit Testing with C++

by Cezary Bloch

<https://github.com/cezbloch/shaderator>

Shaders

- Programs run on GPU
- Executed in parallel
- Originally for shading polygons eg. in games



Compute Shaders/Kernels

- Skip rendering pipeline
- Used for GPGPU
- Used a lot outside gaming industry - research, finance, AI
- Performance boost

```
StructuredBuffer<unsigned int> Input : register( t0 );  
RWStructuredBuffer<unsigned int> Data : register( u0 );
```

```
groupshared unsigned int shared_data[BITONIC_BLOCK_SIZE];
```

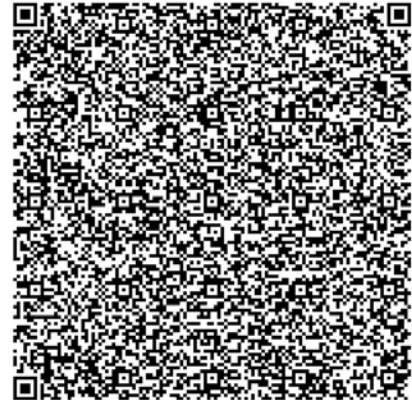
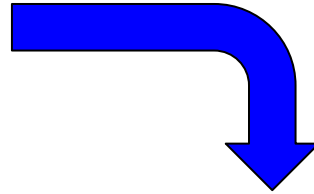
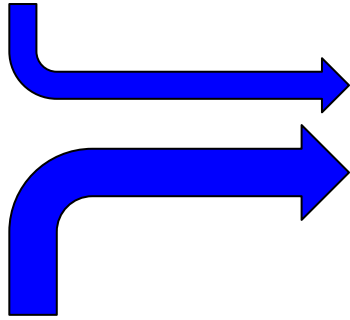
```
[numthreads(BITONIC_BLOCK_SIZE, 1, 1)]
```

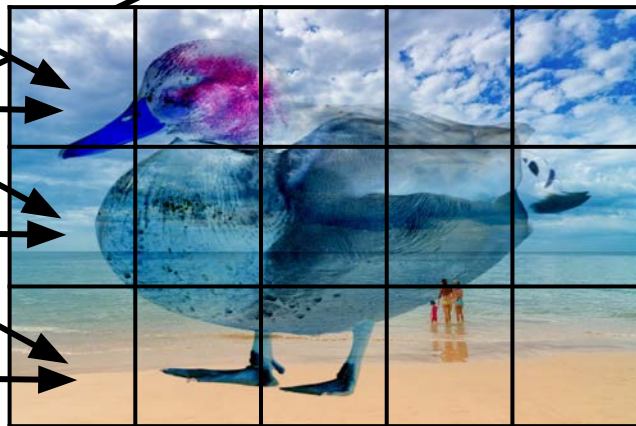
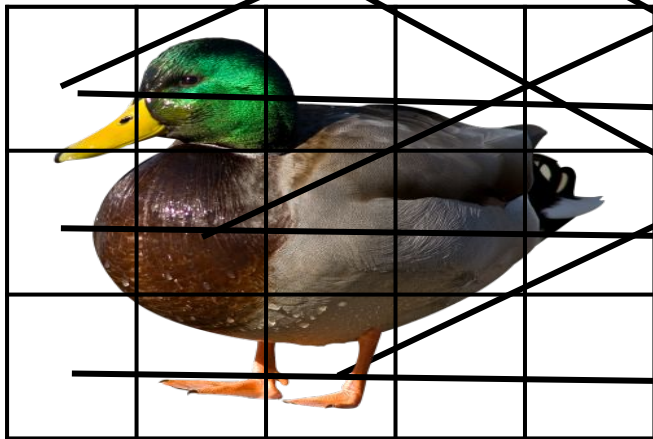
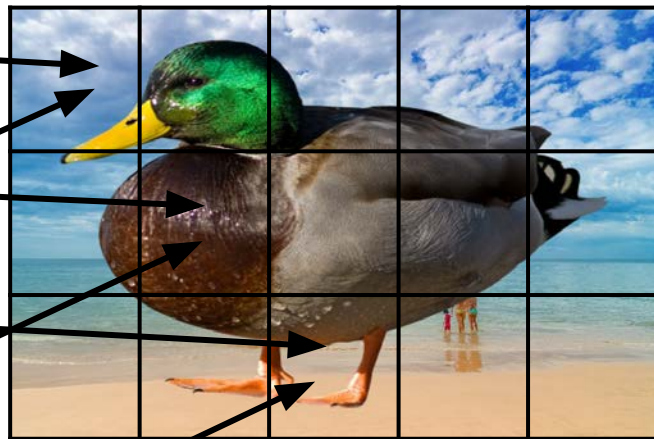
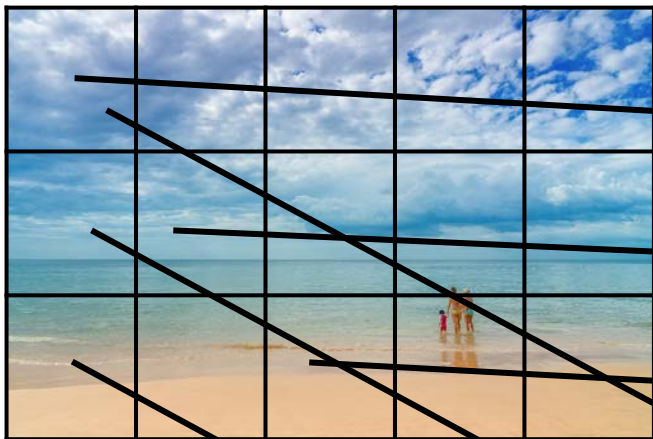
```
void BitonicSort( uint3 Gid : SV_GroupID, uint3 DTid : SV_DispatchThreadID,  
                 uint3 GTid : SV_GroupThreadID, uint GI : SV_GroupIndex )
```

```
{  
    shared_data[GI] = Data[DTid.x];  
    GroupMemoryBarrierWithGroupSync();  
  
    for (unsigned int j = g_iLevel >> 1 ; j > 0 ; j >>= 1)  
    {  
        unsigned int result = ((shared_data[GI & ~j] <= shared_data[GI | j]) ==  
                                (bool)(g_iLevelMask & DTid.x)) ? shared_data[GI ^ j] : shared_data[GI];  
        GroupMemoryBarrierWithGroupSync();  
        shared_data[GI] = result;  
        GroupMemoryBarrierWithGroupSync();  
    }  
    Data[DTid.x] = shared_data[GI];  
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```







In C++

Full debugging support

- Step through
- Data Conditional Breakpoints
- Thread freeze
- Memory & variable look-up
- Assertions/Exceptions
- Unit testing
- Logging/Tracing/Tracepoints



On GPU

Complicated to set-up and limited debugging

- **Step through**
Draw calls required, Record executing and 'replay'
- **Breakpoints/Tracepoints**
On one kernel only
- **Memory & variable look-up**
Some values not available
- **Unit testing**
Check the output buffer
- **Assertions/Exceptions**
- **Logging/Tracing**

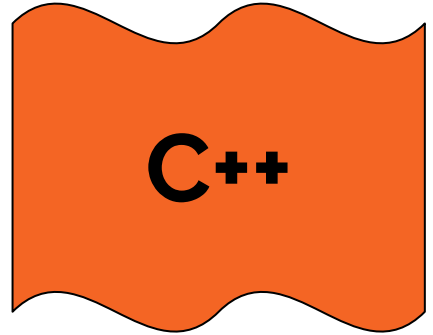
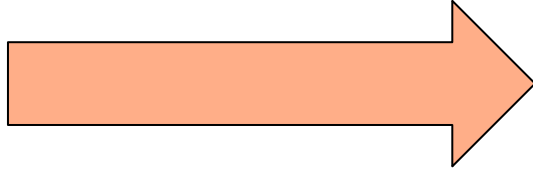


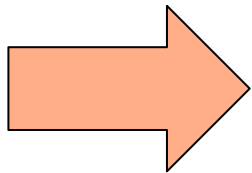
How to bring all the IDE and C++ language features to Compute Shaders on GPU?



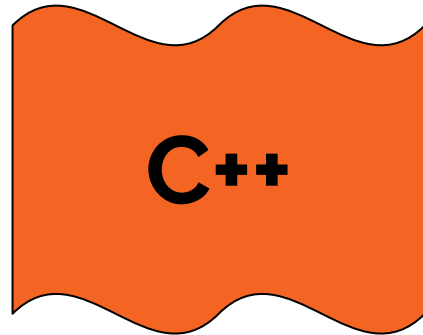
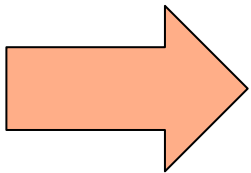
Tip

If it looks **like a duck**, swims **like a duck**, and **quacks like a duck**, then it probably is a duck





?



Shaderator

Macro Magic

Dispatch Engine

Vector operations

HLSL Types

GLSL Types

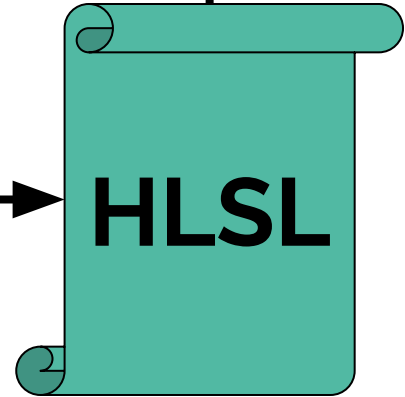


Test Fixture



Executor

- + `set_constants (value_1, value_2)`
- + `set_input_buffer (buffer)`
- + `create_output_buffer (size)`
- + `dispatch ()`



HLSL



shaderator.h

```

StructuredBuffer<unsigned int> SHADERATOR_REGISTER_T(Input, 0);
RWStructuredBuffer<unsigned int> SHADERATOR_REGISTER_U(Data, 0);

groupshared unsigned int shared_data[BITONIC_BLOCK_SIZE];

SHADERATOR_NUM_THREADS(BITONIC_BLOCK_SIZE, 1, 1)
void BitonicSort(SHADERATOR_SV_DispatchThreadID(DTid),
                SHADERATOR_SV_GroupID(Gid),
                SHADERATOR_SV_GroupThreadID(GTid),
                SHADERATOR_SV_GroupIndex(GI))
{
    shared_data[GI] = Data[DTid.x];
    GroupMemoryBarrierWithGroupSync();

    for (unsigned int j = g_iLevel >> 1; j > 0; j >>= 1)
    {
        unsigned int result = ((shared_data[GI & ~j] <= shared_data[GI & j]) ? shared_data[GI & ~j] : shared_data[GI & j]);
        GroupMemoryBarrierWithGroupSync();
        shared_data[GI] = result;
        GroupMemoryBarrierWithGroupSync(); // ≤ 330ms elapsed
    }

    Data[DTid.x] = shared_data[GI];
}
    
```

Memory 1

Address: 0x00007FF741810790

0x00007FF7418108C0	41892998	42934151	42938095
0x00007FF7418108CC	43021413	43287081	43685427
0x00007FF7418108D8	44521004	44683535	45364010
0x00007FF7418108E4	45372457	45469149	46533543
0x00007FF7418108F0	47079645	47154226	47906564
0x00007FF7418108FC	48369600	48632677	48747620
0x00007FF741810908	50308392	50596092	50600519
0x00007FF741810914	51002888	51187722	51313534
0x00007FF741810920	52296240	52511861	55064898
0x00007FF74181092C	55189161	55456913	56135828
0x00007FF741810938	56533607	57118212	57439490
0x00007FF741810944	57674087	57809853	58502998
0x00007FF741810950	58693791	59175185	60286310
0x00007FF74181095C	60837065	61673846	61861858
0x00007FF741810968	61914557	62233967	62409287
0x00007FF741810974	62482867	63090289	64367950
0x00007FF741810980	64938454	65058609	65248391
0x00007FF74181098C	65520689	65828522	66324009
0x00007FF741810998	66353165	67852245	67855871
0x00007FF7418109A4	69277413	69648101	69690108
0x00007FF7418109B0	70332119	70616528	71113646
0x00007FF7418109BC	71455437	71978175	73012593

GPU Threads Autos Memory 1 Threads Modules Watch 1

Locals

Name	Value	Type
DTid	{x=486 y=0 z=0 ...}	glm::vec<3>
GI	486	unsigned in
Gid	{x=0 y=0 z=0 ...}	glm::vec<3>
GTid	{x=486 y=0 z=0 ...}	glm::vec<3>
j	1	unsigned in
result	254152185	unsigned in

- Search
- Run All | Run... | Playlist
- Shaderator (2 tests)
- ComputeShaderUnitTests
 - TestBitonicSortEmptyTr
 - GlslRayTracerUnitTests (1)
 - TestRayAtOrigin 1 ms

Summary

Last Test Run Passed (Total Run)

2 Tests Passed

—
Faster development

Less errors

Quick problem diagnosis

Protection against regressions

Same code for C++ and HLSL

Enhancement of existing tools

What people are saying?

What a great idea! I've not heard of anybody else doing this.

I have been using the same approach for over 6 years and it's by far the best way to develop HLSL shaders.


If we don't have tests build with Shaderator I'm not changing our shader!

—
shaderator

github.com

GT-X 970

INVIDIA

A background of rich red, vertically pleated curtains with gold tassels on the sides, framing the central text.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**

GREEN
GORGEOUSNESS...



GALILEO GALILEI



GUACAMOLE

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

Literal *misdirection*

Previously on ...

ACCU lightning talks

Previously on ...

ACCU lightning talks

UK

1 gallon = 8 pints

US

1 gallon = 8 pints

Previously on ...

ACCU lightning talks

UK

1 gallon = 8 pints

1 pint = 20 fl. oz.

US

1 gallon = 8 pints

1 pint = 16 fl. oz.

Previously on ...

ACCU lightning talks

UK

1 gallon = 8 pints

1 pint = 20 fl. oz.

1 fl. oz. = 28.41 ml

US

1 gallon = 8 pints

1 pint = 16 fl. oz.

1 fl. oz. = 29.57 ml

Previously on ...

ACCU lightning talks

UK

1 gallon = 8 pints

1 pint = 20 fl. oz.

1 fl. oz. = 28.41 ml

1 gallon = 4,545.6 ml

US

1 gallon = 8 pints

1 pint = 16 fl. oz.

1 fl. oz. = 29.57 ml

1 gallon = 3,785.0 ml

- *Chuck drinks 6 pints at his local bar,*
- *Reggie drinks 5 pints at the pub.*

Who drank fewer pints?

Who drank less beer?

*“I’m literally bursting for a pee,”
says Reggie.*

**Is a “*Meaning of Life*” moment
coming?**

**Has Reggie failed a *BBC R4*
test?**

Literally or figuratively

literally, adv. ... 3.b. Used as an intensive before a figurative expression.

- *The American Heritage Dictionary of the English Language, 2016*

Oxford comma

The canning, processing, preserving, freezing, drying, marketing, storing, packing for shipment or distribution of:

- (1) Agricultural produce;*
- (2) Meat and fish products; and*
- (3) Perishable foods.*

The drivers argued, due to a lack of a comma between “packing for shipment” and “or distribution”, the law refers to the single activity of “packing”, not to “packing” and “distribution” as two separate activities. As the drivers distribute – but do not pack – the goods, this would make them eligible for overtime pay.

US Court of Appeals, First Circuit, March 2017

Example mapping

Story

Rule

Rule

Rule

Question

Question

Question

Example

The one where...

Example

The one where...

Example

The one where...

Example

The one where...

<https://cucumber.io/blog/2015/12/08/example-mapping-introduction>

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

The background of the image consists of rich, red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed stage area. Each side of the curtain is held back by a decorative gold-colored tassel. The lighting is dramatic, highlighting the texture and folds of the fabric.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



YE OLDE YUM



ARCHIMEDES



ANCHOVIES

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

Two Small Corrections

Anna-Jayne Metcalfe
@annajayne
anna@riverblade.co.uk

Riverblade Ltd
www.riverblade.co.uk



Two Small Corrections

What Stories Can You Tell?

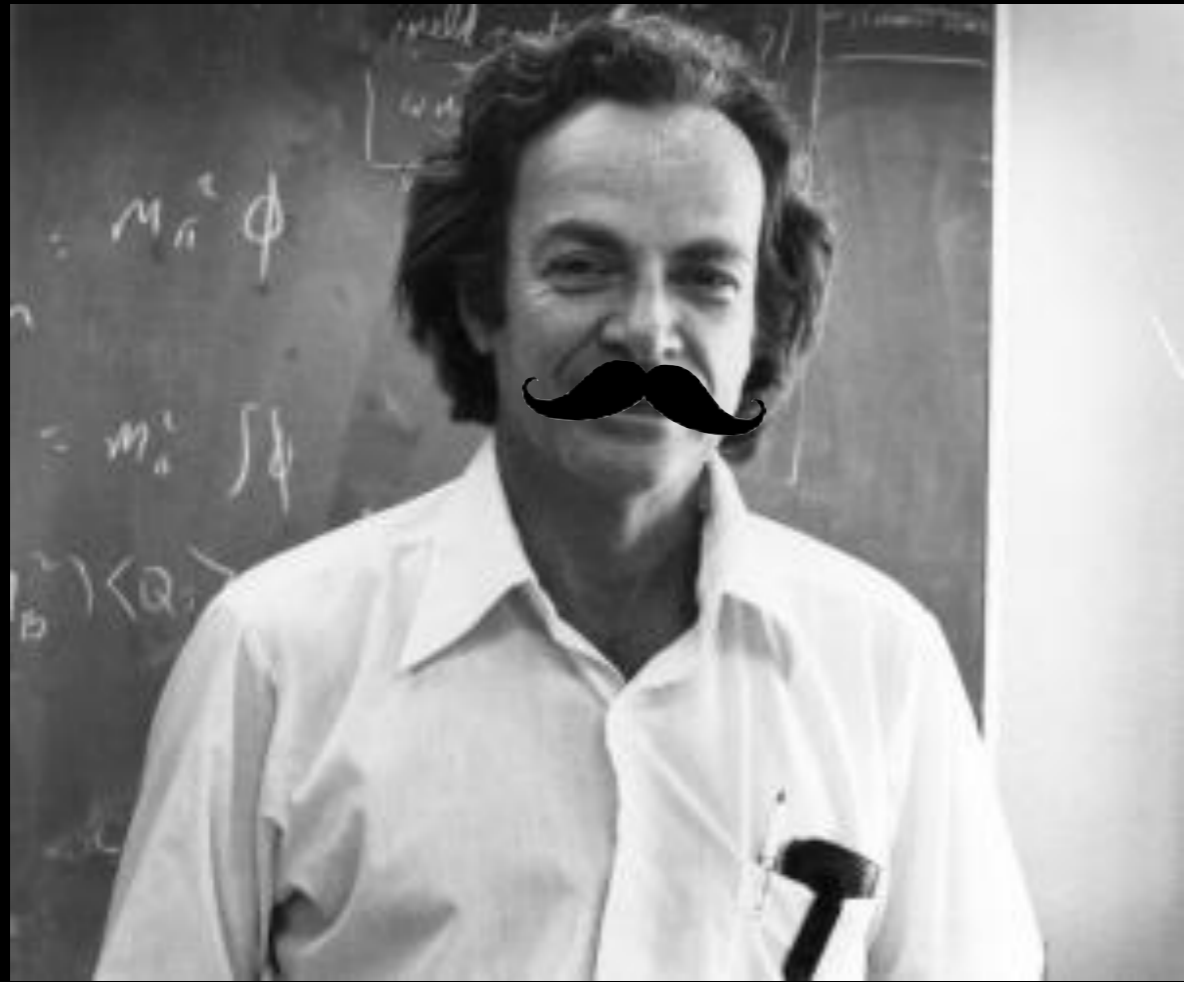
What about the people you know?

Are you listening?

#caffeinedrivendevelopment

The background of the image consists of rich, red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed stage area. Each side of the curtain is held back by a gold-colored rope with a matching tassel. The lighting is dramatic, highlighting the texture of the fabric and the central text.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



MY STOMACH'S
RUMBLING



**RICHARD
FEYNMAN**



FISH

(ANY, HE'S NOT FUSSY)

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

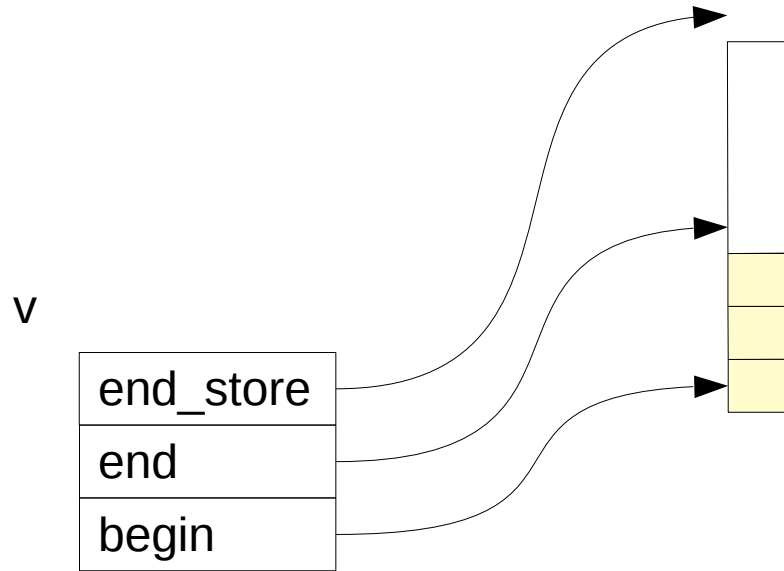
My favourite memory leak

```
#include <vector>

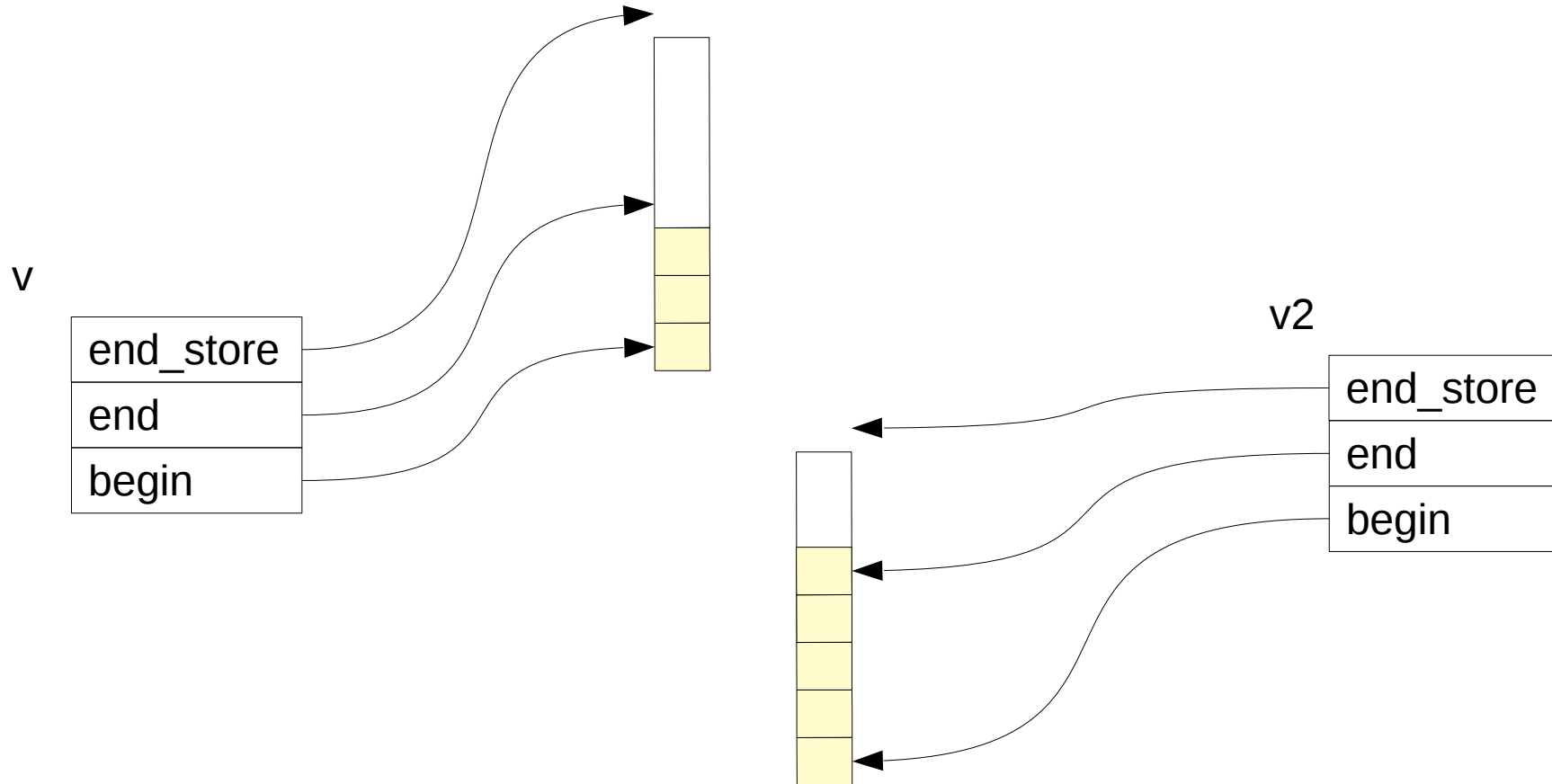
struct V : std::vector<V> {};

int main()
{
    V v;
    v.emplace_back();
    v.swap(v.front());
}
```

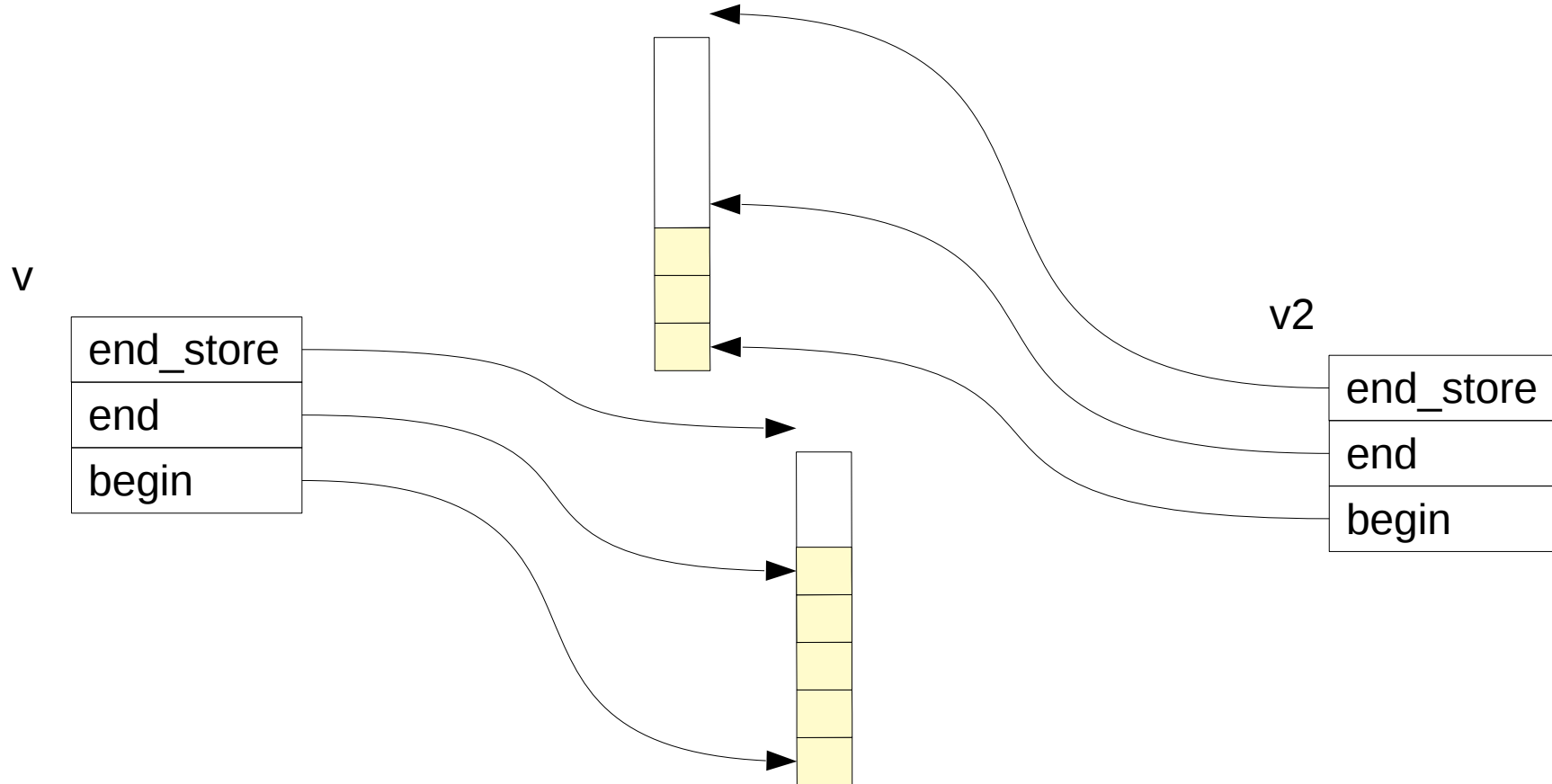
My favourite memory leak



My favourite memory leak

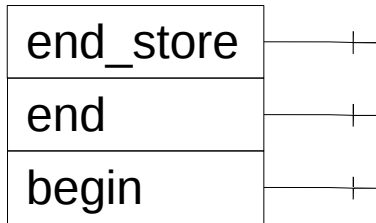


My favourite memory leak



My favourite memory leak

v



```
#include <vector>
```

```
struct V :  
std::vector<V> {};
```

```
int main()  
{
```

```
V v;
```

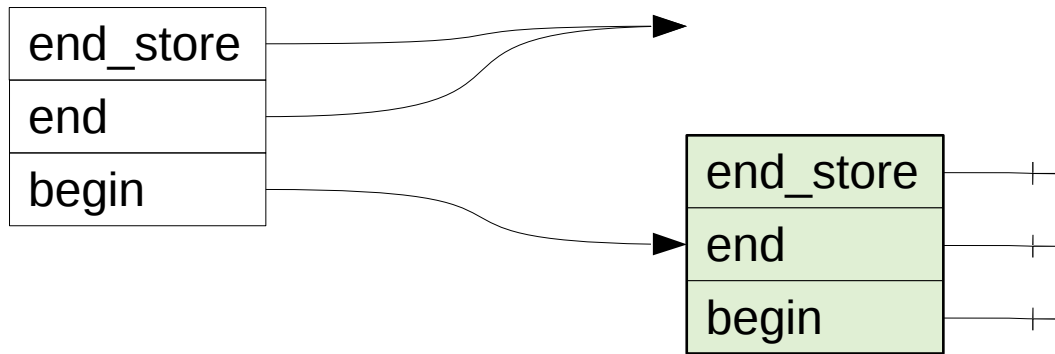
```
v.emplace_back();
```

```
v.swap(v.front());
```

```
}
```

My favourite memory leak

v



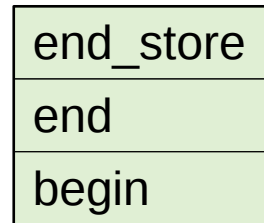
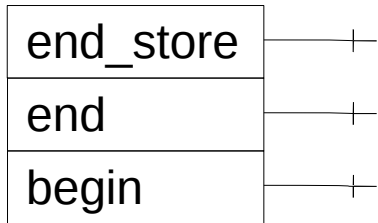
```
#include <vector>
```

```
struct V :  
std::vector<V> {};
```

```
int main()  
{  
    V v;  
    v.emplace_back();  
    v.swap(v.front());  
}
```

My favourite memory leak

v



```
#include <vector>
```

```
struct V :  
std::vector<V> {};
```

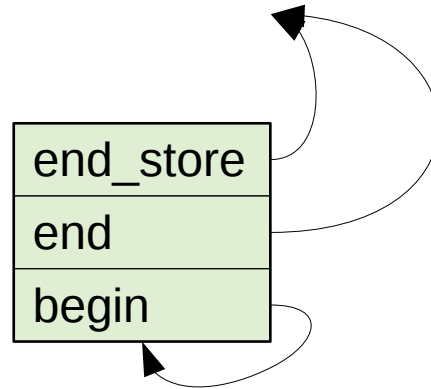
```
int main()  
{  
    V v;  
    v.emplace_back();  
    v.swap(v.front());  
}
```


My favourite memory leak

```
#include <vector>
```

```
struct V :  
std::vector<V> {};
```

```
int main()  
{  
    V v;  
    v.emplace_back();  
    v.swap(v.front());  
}
```



My favourite memory leak

Björn Fahller



bjorn@fahller.se



[@bjorn_fahller](https://twitter.com/bjorn_fahller)

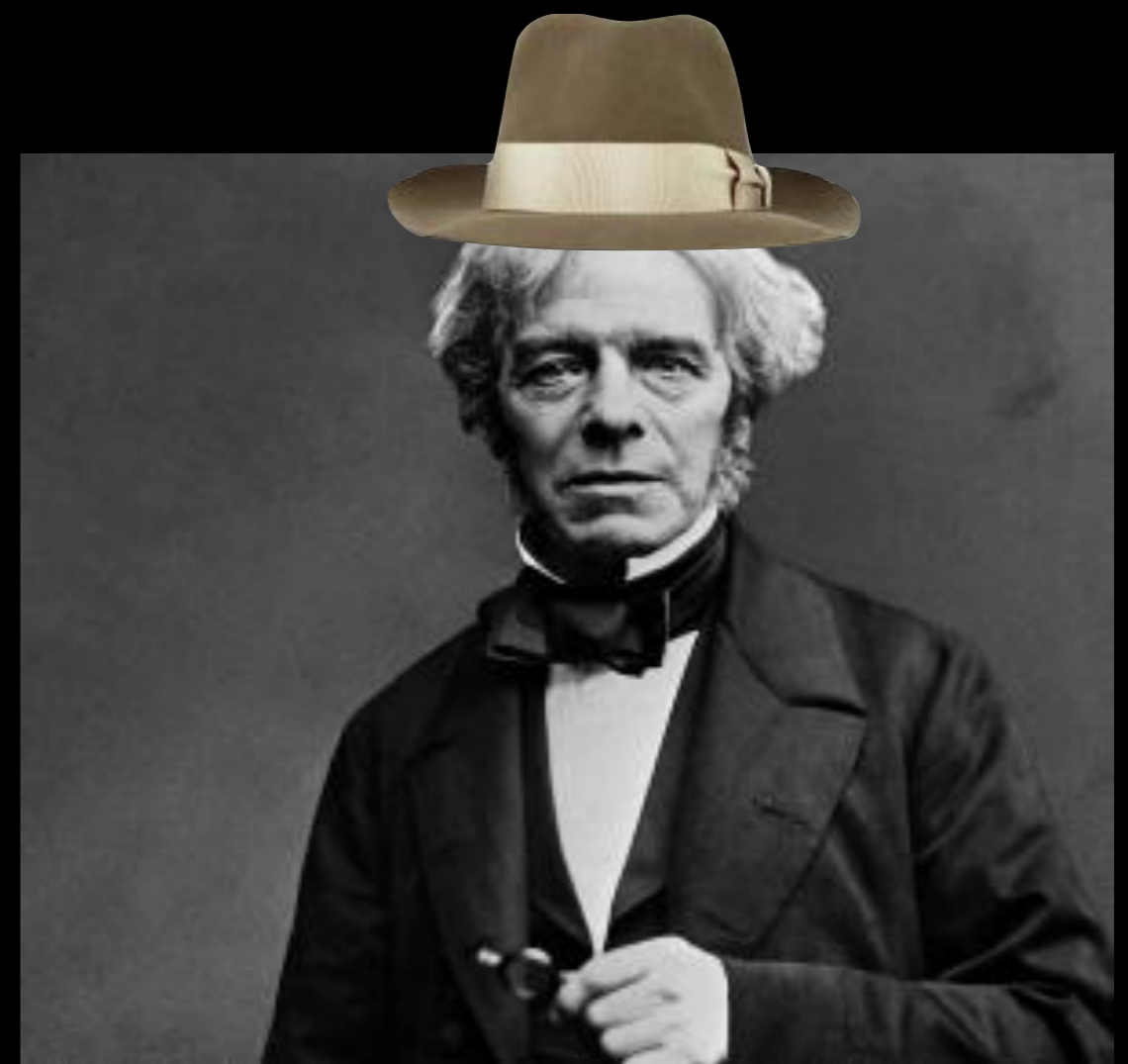


[@rollbear](https://twitter.com/rollbear) *cpplang, swedencpp*



The background of the image consists of rich red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed area behind them. Each side of the curtain is held back by a gold-colored rope with a matching tassel. The lighting is dramatic, highlighting the texture of the fabric and the metallic sheen of the tassels.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



FIX ME THAT FOOD

**MICHAEL
FARADAY**



FOCACCIA

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

```
for i := 0; i <= 5; i++ {  
    fmt.Printf("%d\n", i)  
}
```

```
const FIVE = 5 // Five
```

```
for i := 0; i <= FIVE; i++ {  
    fmt.Printf("%d\n", i)  
}
```



```
const FIVE = 6 // Tau
```

```
for i := 0; i <= FIVE; i++ {  
    fmt.Printf("%d\n", i)  
}
```

```
const maxIterations = 5

for i := 0; i <= maxIterations; i++ {
    fmt.Printf("%d\n", i)
}
```

```
for i := 0; i <= maxIterations; i++ {  
    fmt.Printf("%d, %d\n", i, maxIterations)  
}
```

A background of rich red, vertically pleated curtains with gold tassels on the sides. The text is centered in a bold, white, sans-serif font.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



A PARTICULARLY
FINE FOODSTUFF



MAX PLANCK



PANETTONE

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview



*Care of
Magical
Creatures*

*By
Gail Ollis*

I really enjoyed researching my paper!

I really enjoyed researching my paper!



I really enjoyed researching my paper!

It was definitely uncomfortable to speak in front of a lot of people especially in a very “male” orientated course which sort of makes me feel like I shouldn't be there!

#metoo

#mansplained



#notsecretary



#notprincess





STORIES

FOR
BOYS

WHO
DARE

-TO BE-
DIFFERENT

TRUE TALES OF AMAZING BOYS
WHO CHANGED THE WORLD WITHOUT
KILLING DRAGONS



The background of the image consists of rich, red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed area behind them. Each side of the curtain is held back by a decorative gold-colored tassel. The lighting is dramatic, highlighting the texture and folds of the fabric.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**



MOST EDIBLE

**AMEDEO
AVOGADRO**



APPLES

Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - `</rant>`
Pete Goodliffe - The New C++ Interview

RANKING HACKERS

Steve Love // essennell.love@gmail.com // [@IAmSteveLove](https://twitter.com/IAmSteveLove)

The background of the image consists of rich, red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed area behind them. Each side of the curtain is held back by a gold-colored rope with a matching tassel. The lighting is dramatic, highlighting the texture and folds of the fabric.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**

MMMMmmmm...

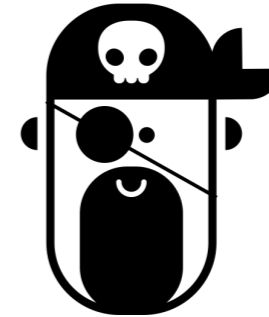
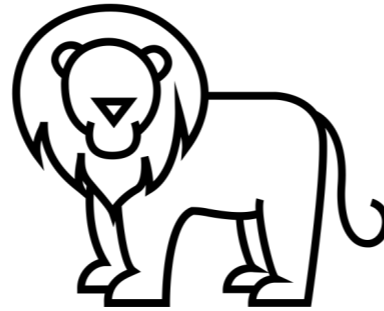
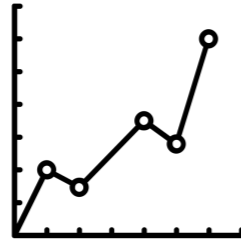
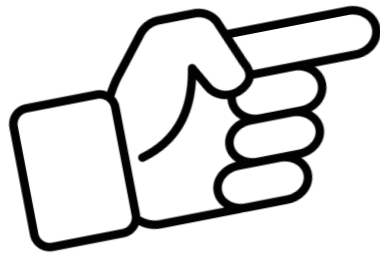


NIELS BOHR



BURRITOS

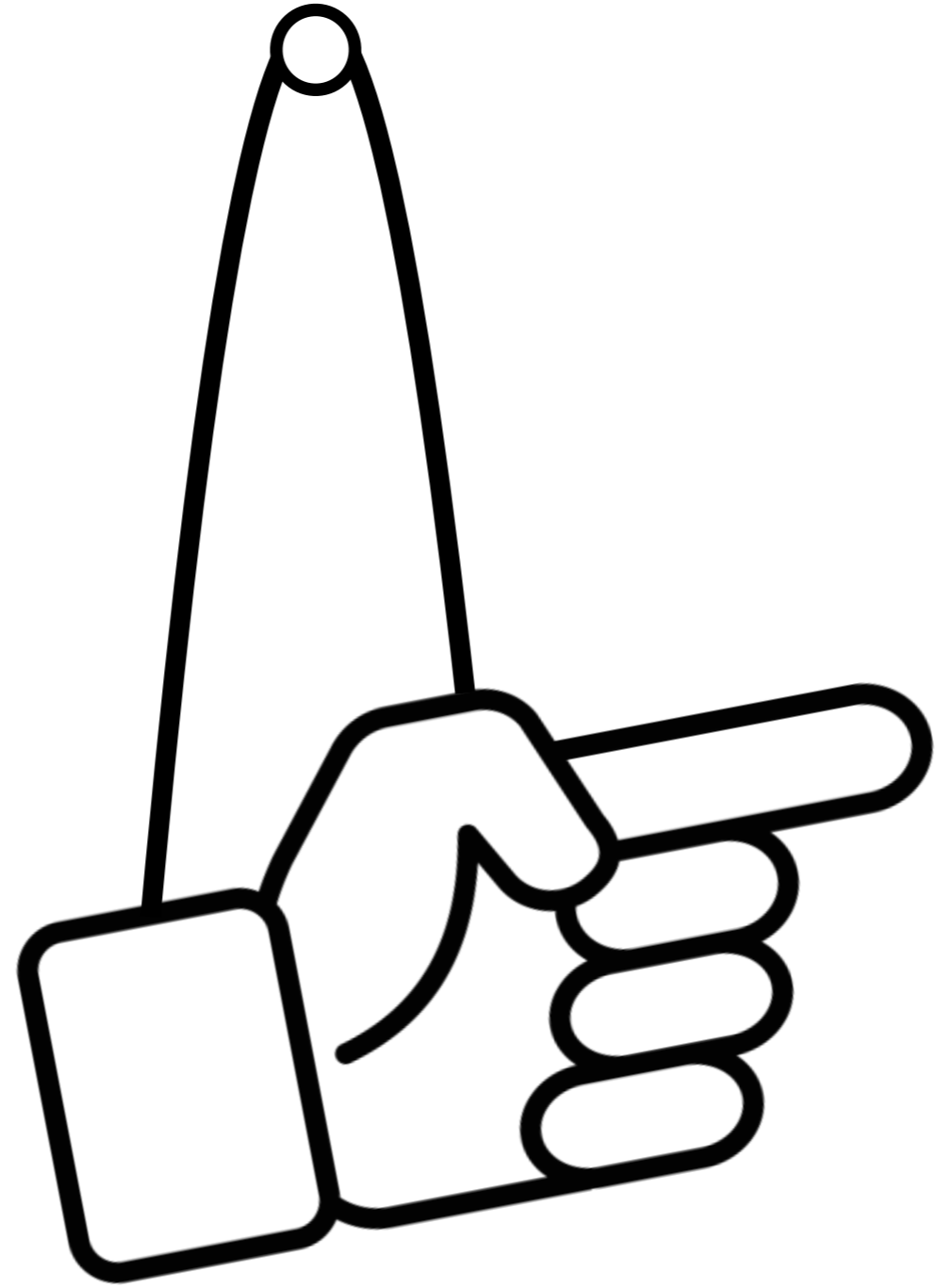
Guy Davidson - A year in diversity
Jon Jagger - FizzBuzz in the C pre-processor
Frances Buontempo - Here beis a dragons
Peter Sommerlad - APRIL
Cezary Bloch - Shaderator
Seb Rose - Literal Misdirection
Anna-Jayne - Two Small Corrections
Björn Fahlner - My favourite memory leak
Dom Davis - Putting the away into go
Gail Ollis - Care of Magical Creatures
Steve Love - </rant>
Pete Goodliffe - The New C++ Interview

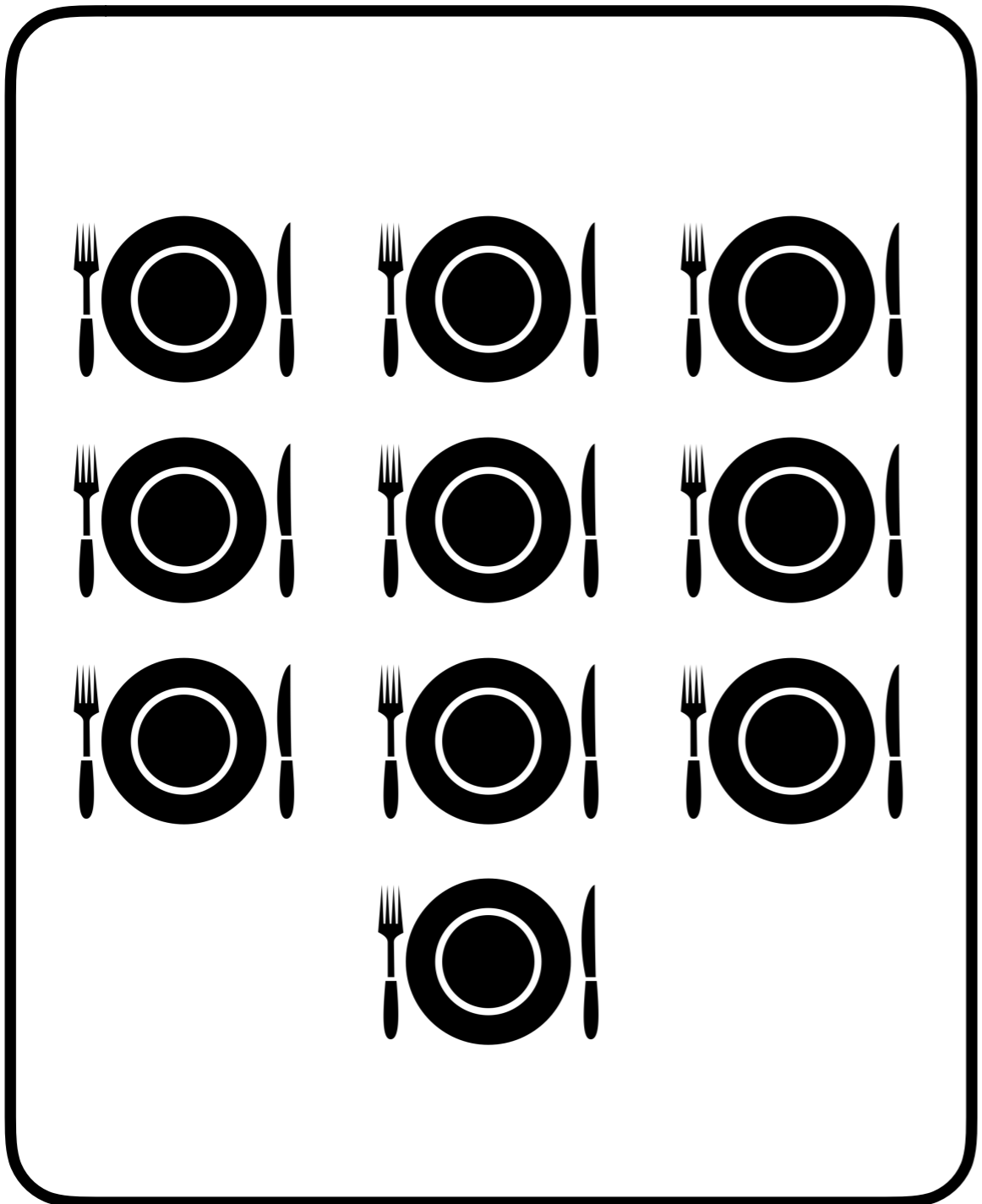


THE NEW C++ INTERVIEW

(WITH APOLOGIES TO YOUR SANITY)

Pete Goodliffe
pete@goodliffe.net
@petegoodliffe

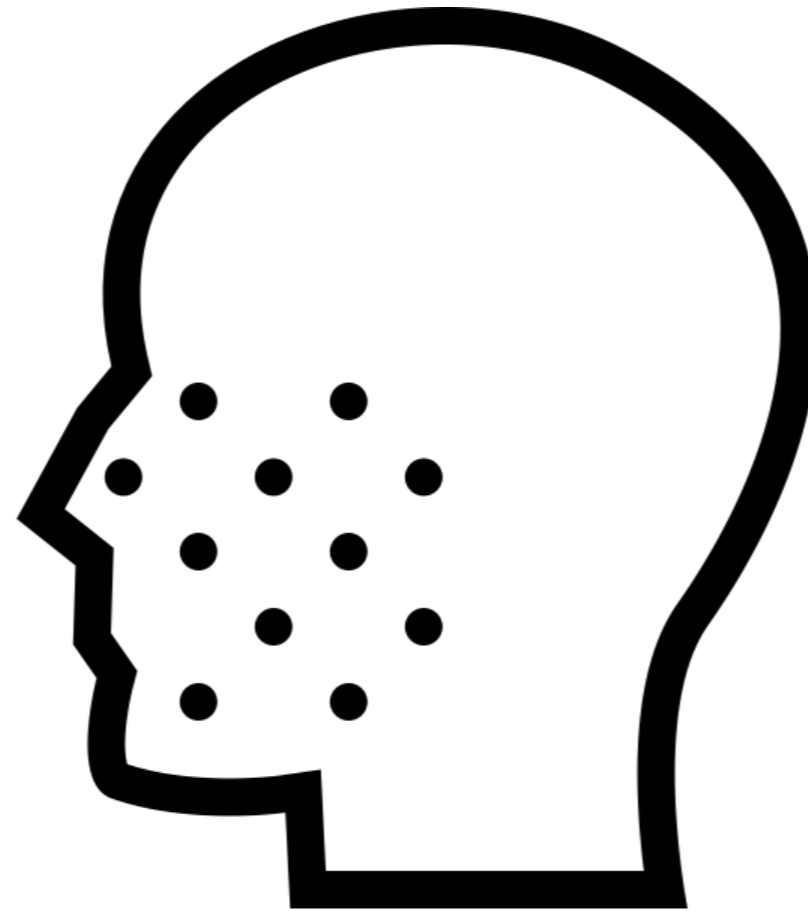




Hello!

Hi

APPROGRAMMING



idiom

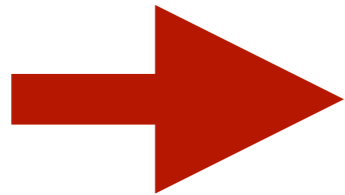
class

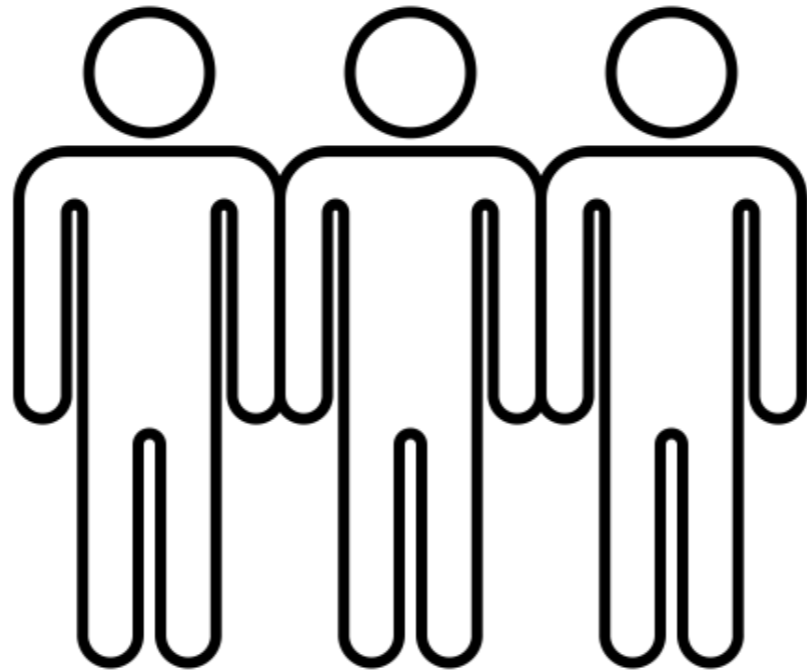
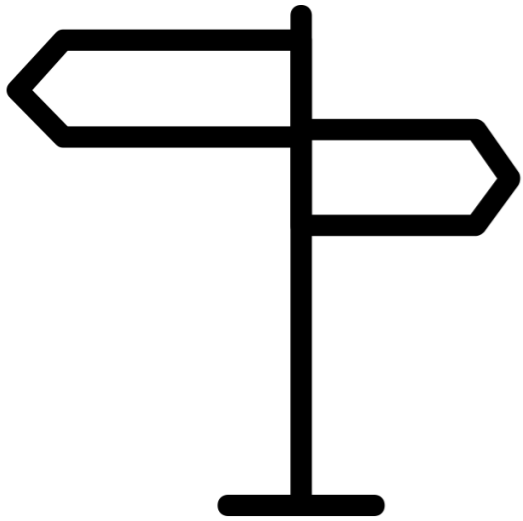
class

~~**class**~~

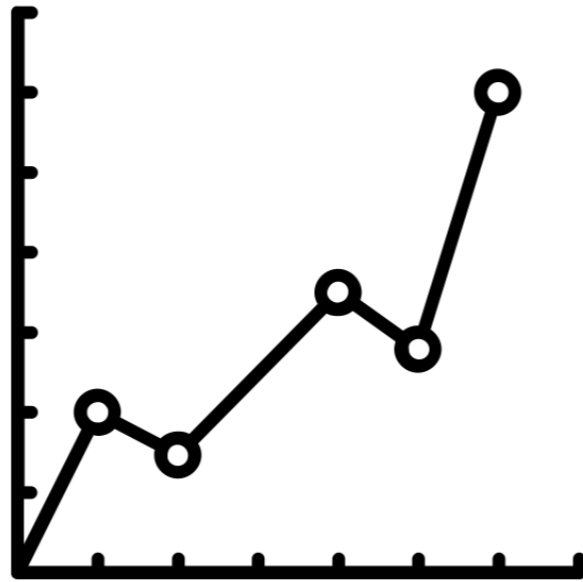
class

class

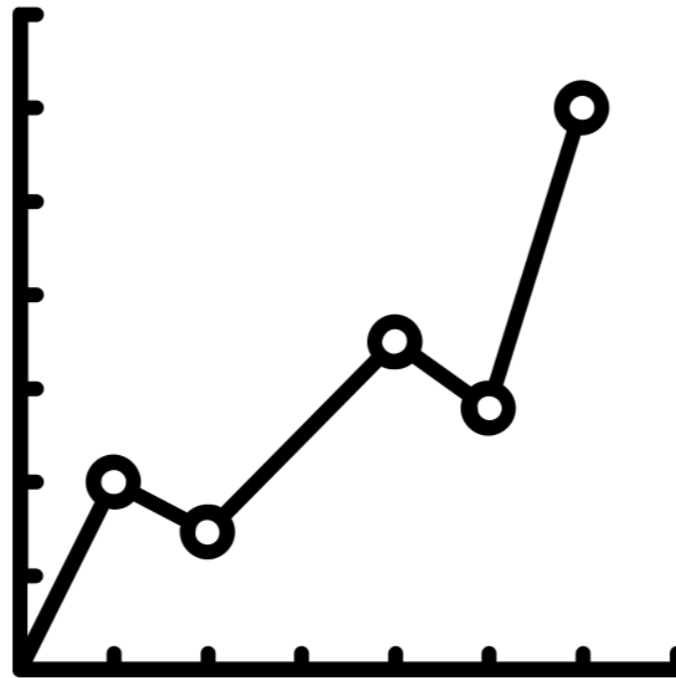
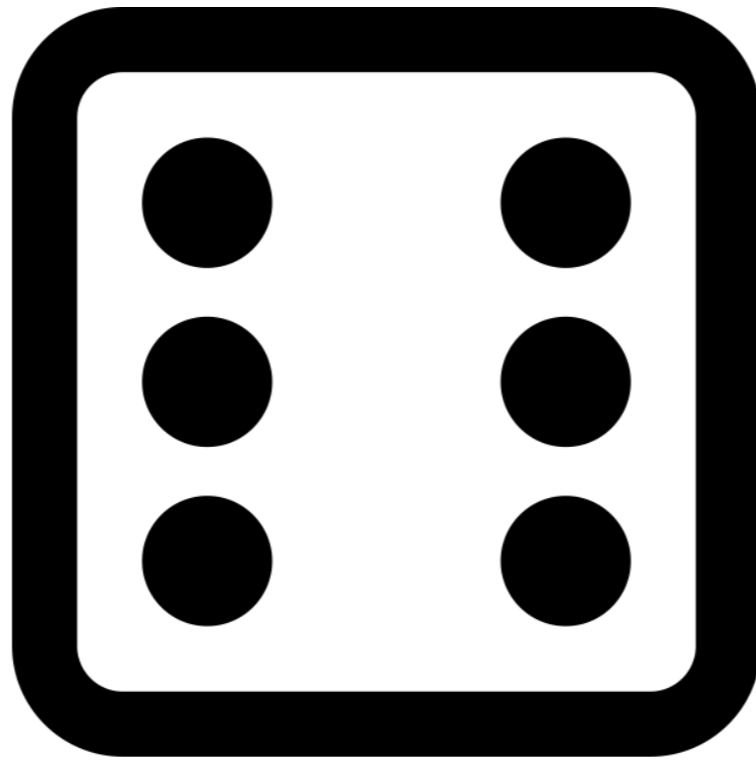


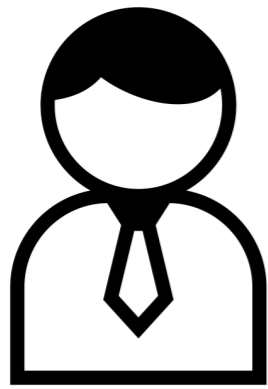


```
try  
{
```

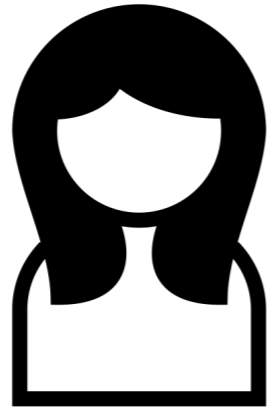


```
}  
catch (...)  
{  
    // aaaghr!  
}
```

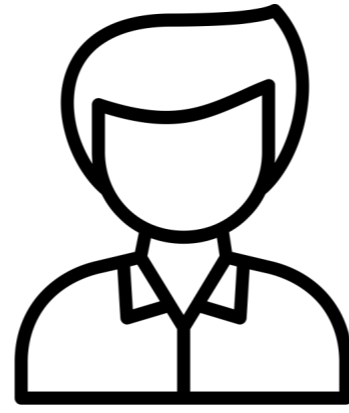




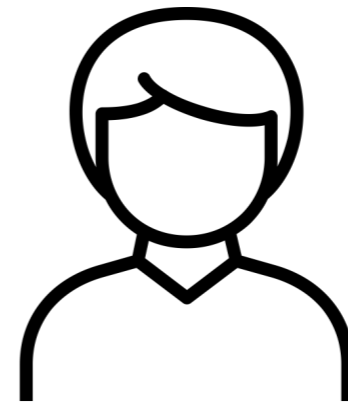
father



mother



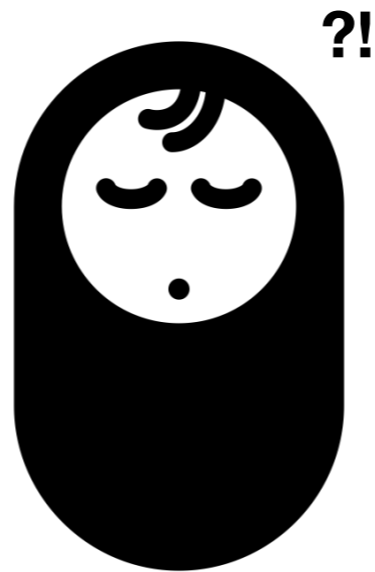
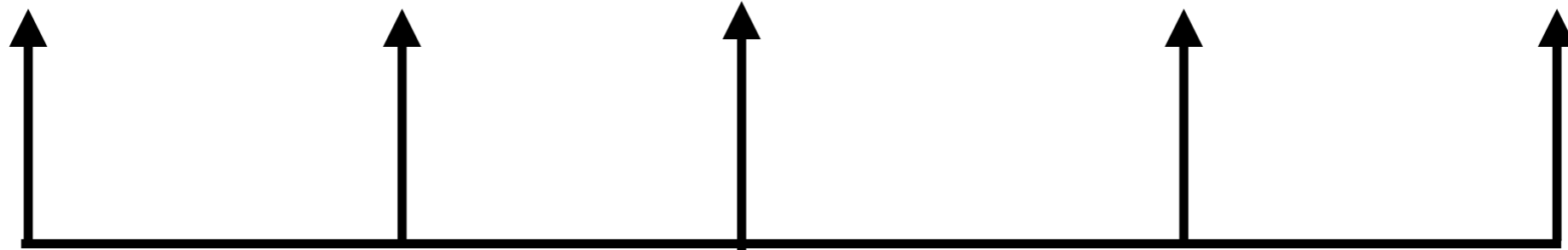
other father



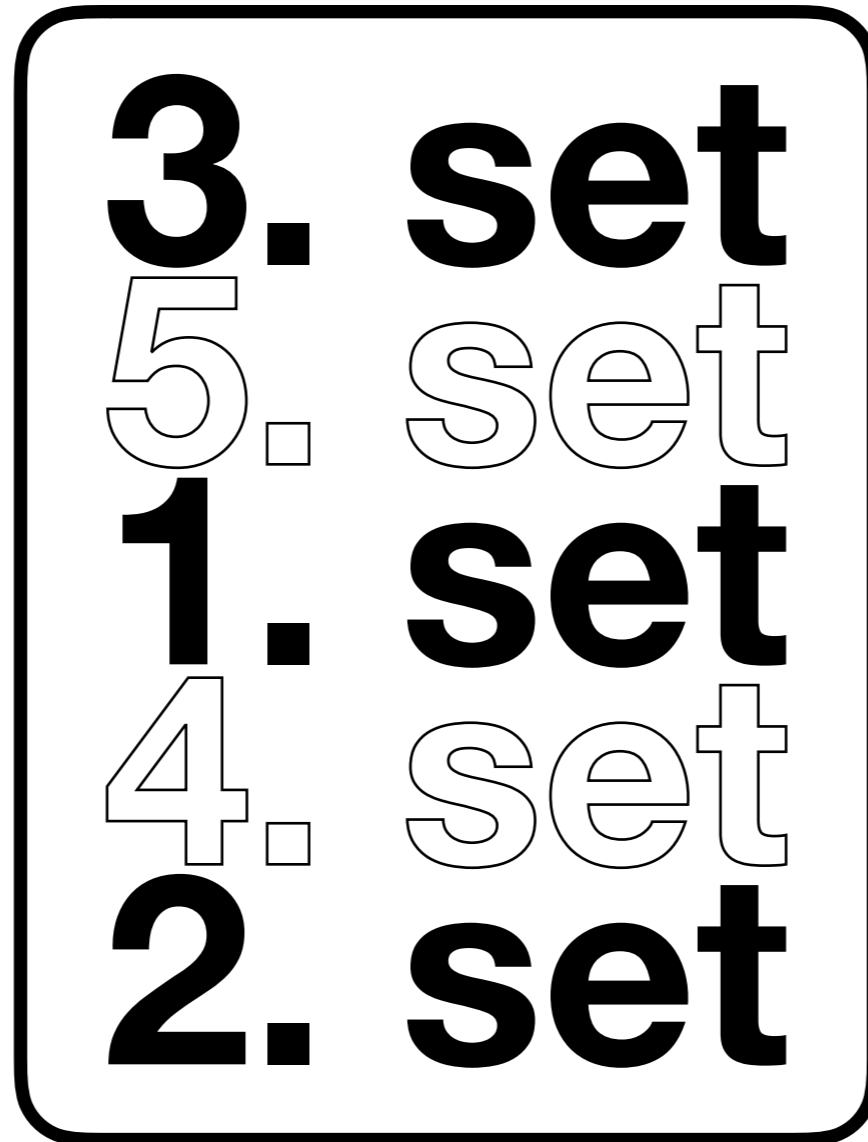
debatable

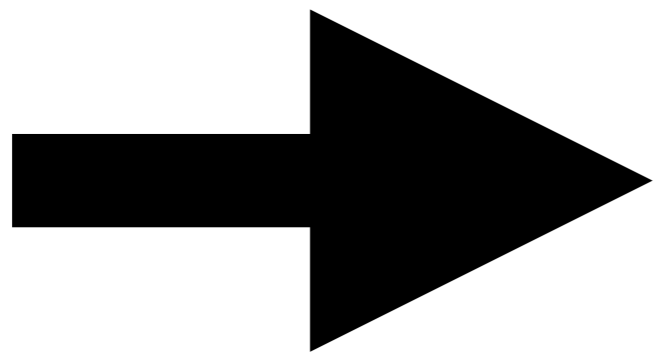


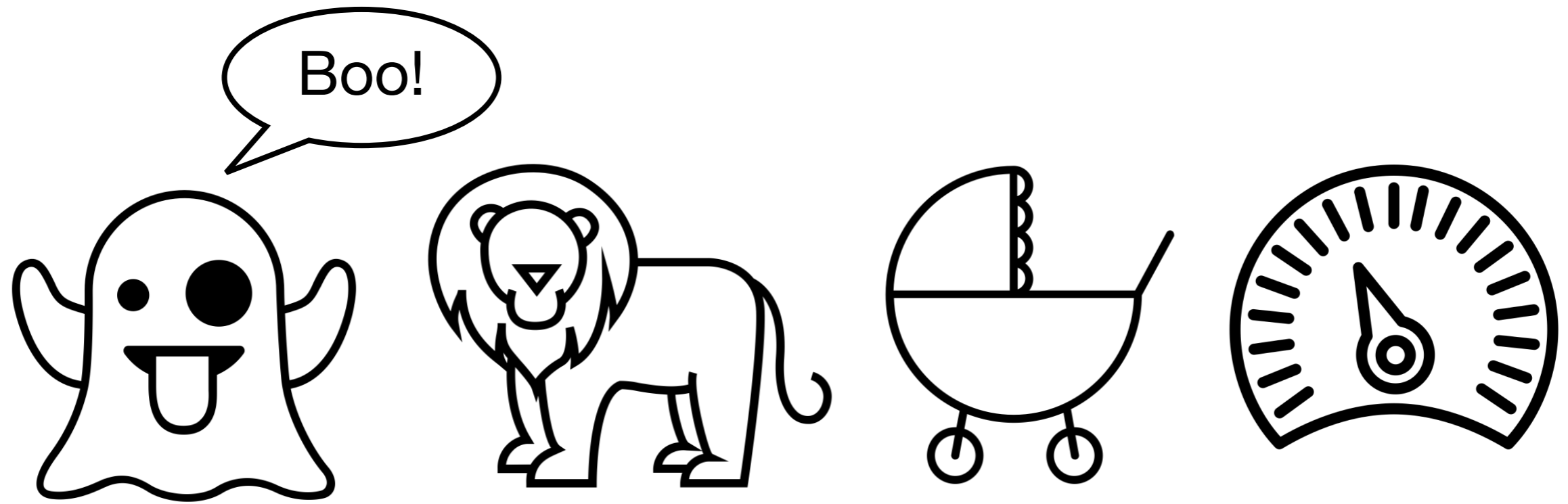
mother #2



gonorrhoea







char

.....
NAME

times 2

← 100 miles →

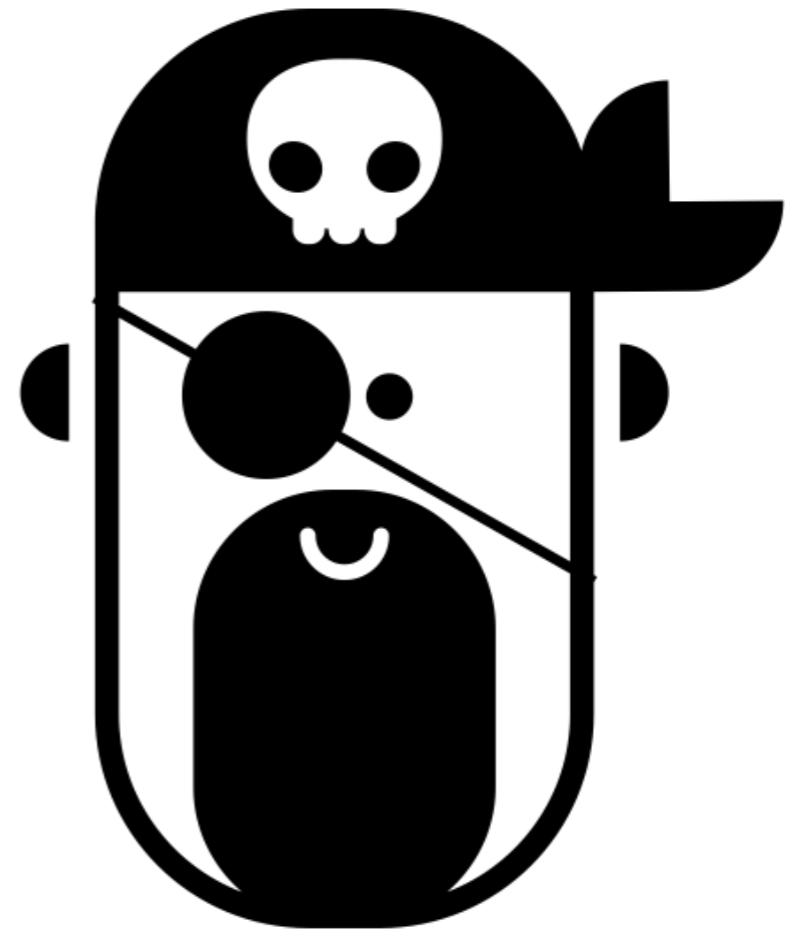


invitation

**this is a party
only for people in
the club**



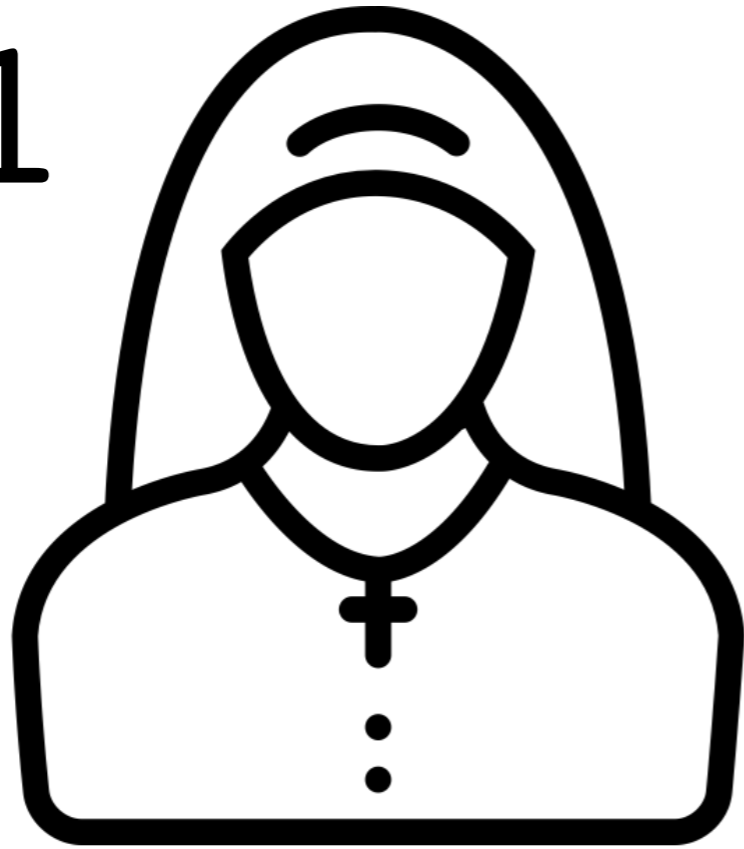
&



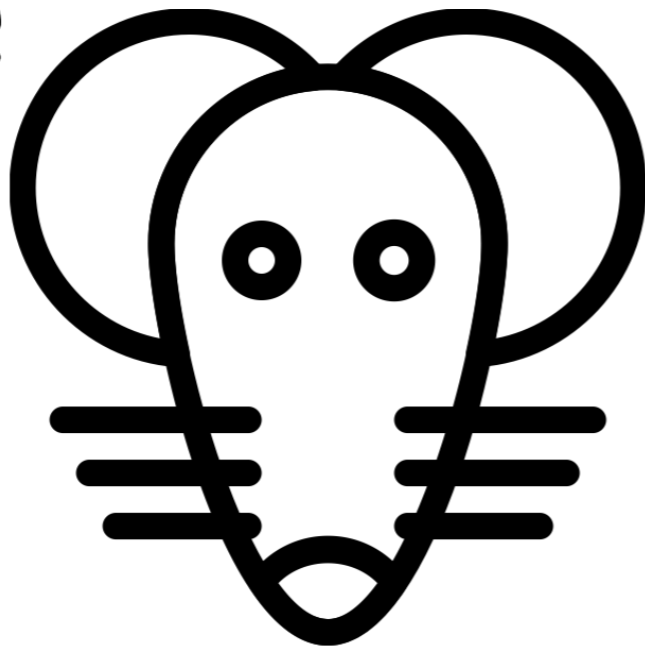
- 1. substitution principle**
2. substitution principle
- 3. substitution principle**
4. substitution principle
- 5. substitution principle**
6. substitution principle
- 7. substitution principle**
8. substitution principle
- 9. substitution principle**
10. substitution principle

function — function — function — function — function — function

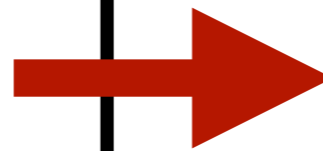
0x01



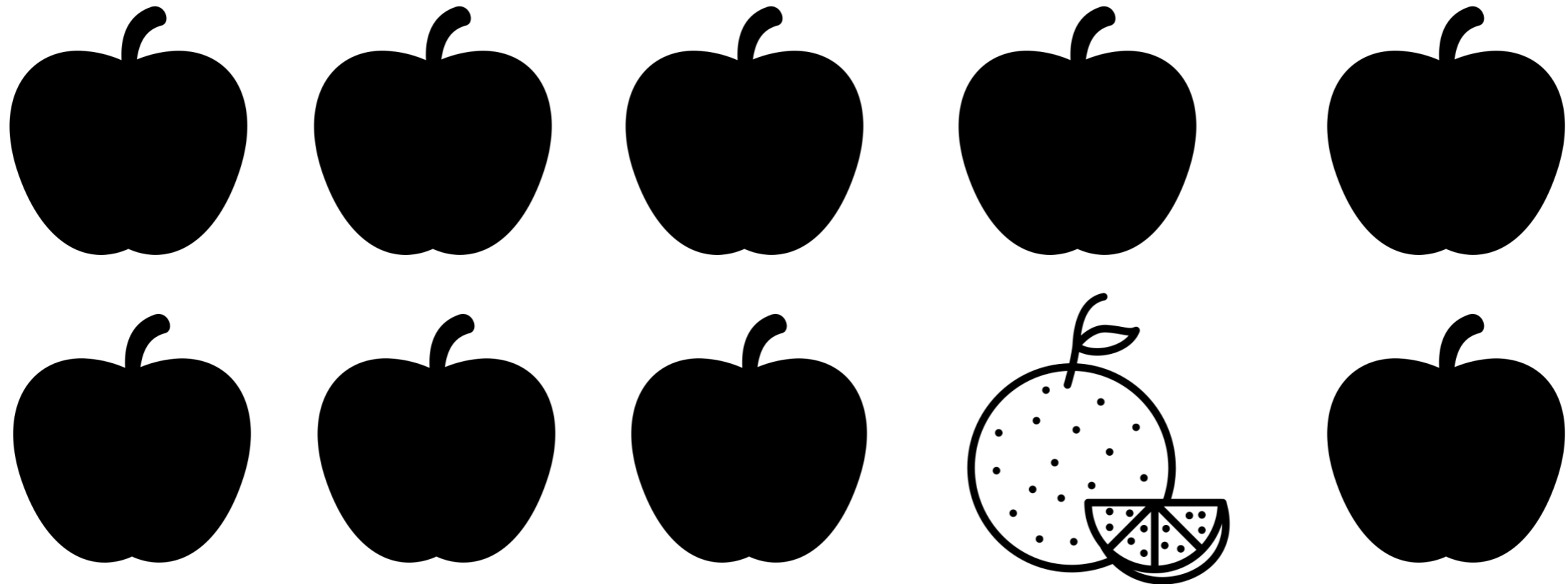
0x0e



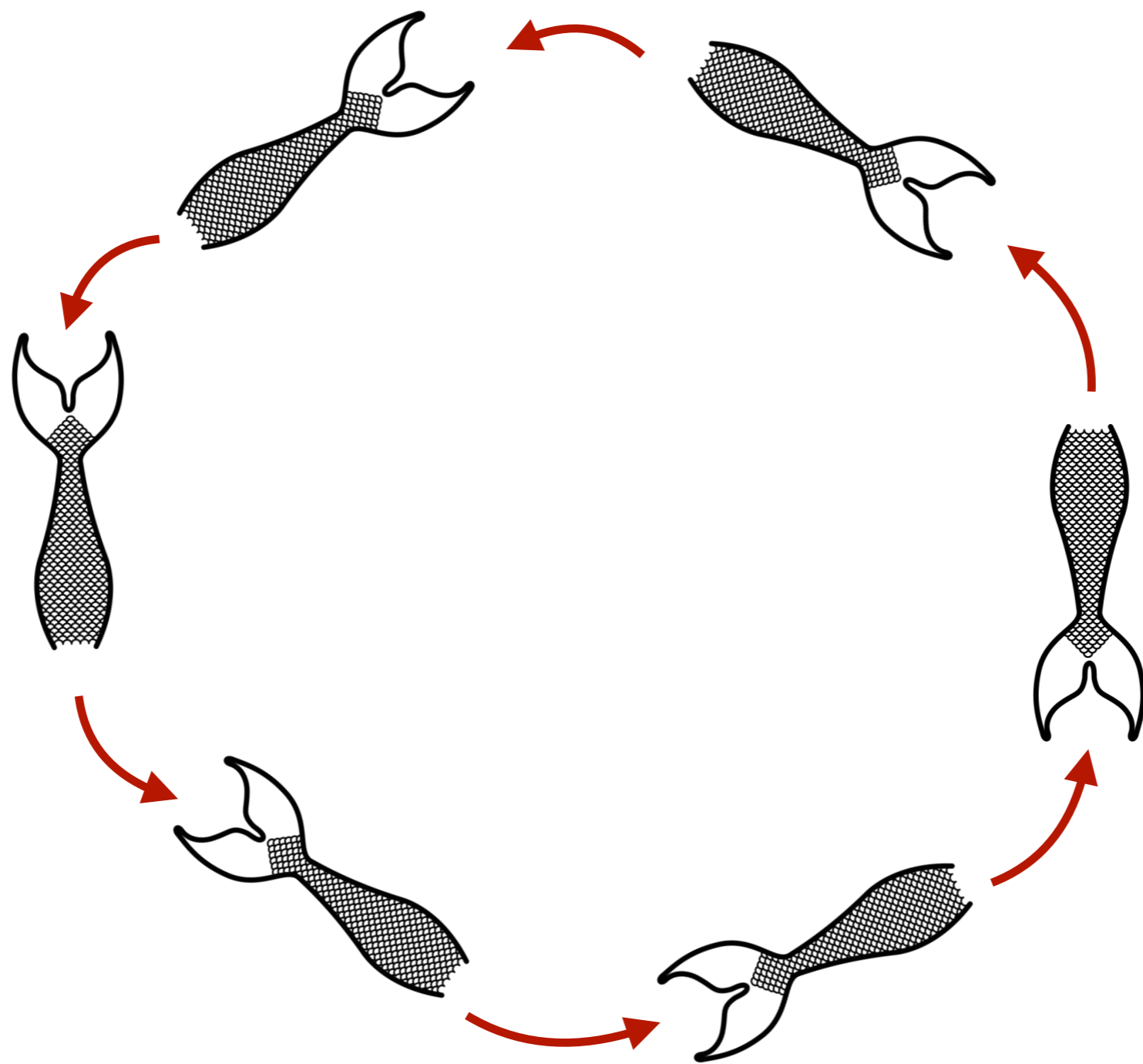
.....pete.....
.....robert.....
.....freeda.....
.....charles.....
.....alice.....

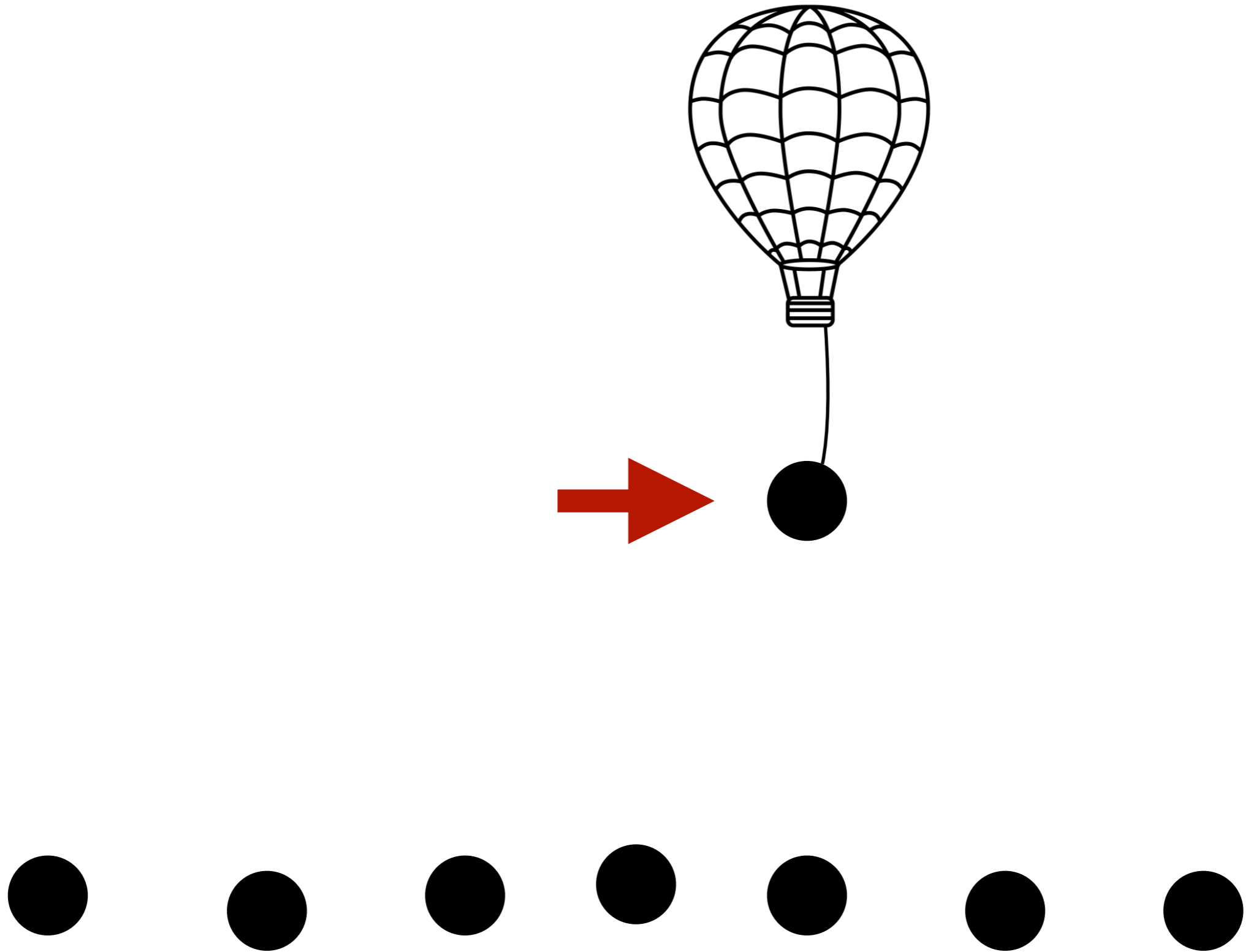


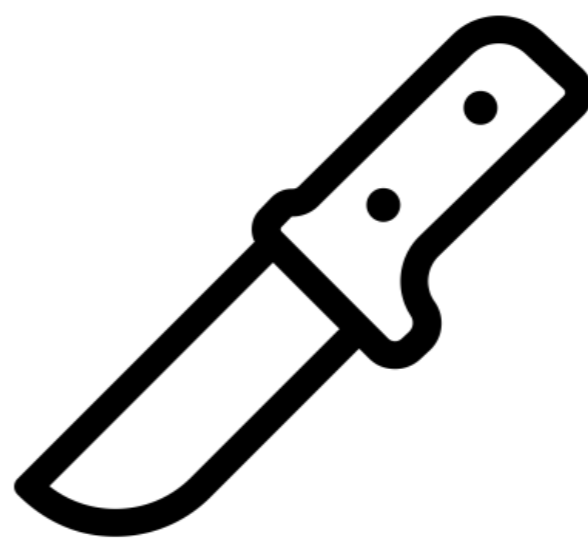
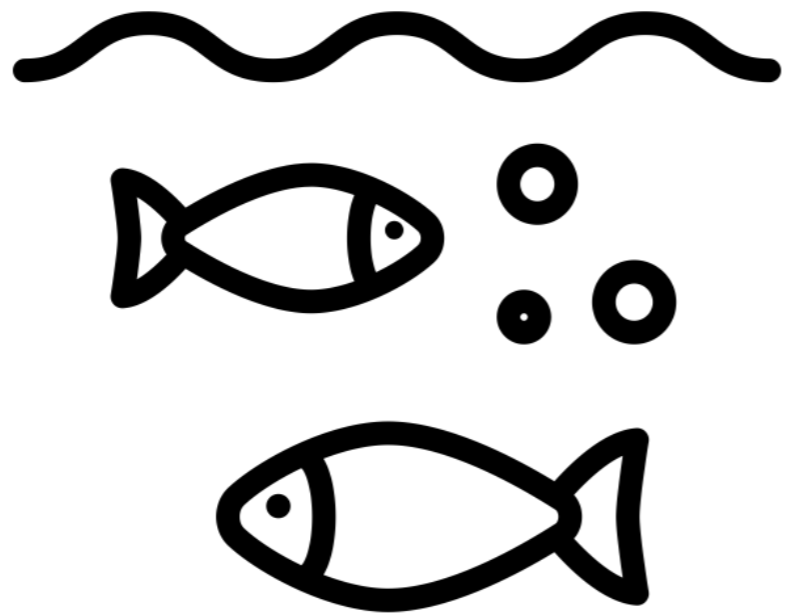
you swapped it!




this is not a problem









NO!!!



that way...

gluttony
lust
avarice
pride
wrath
vanity
sloth

Bill: \$1000

The background of the image consists of rich red, vertically pleated curtains. The curtains are drawn back on both sides, revealing a dark, shadowed area behind them. Each side of the curtain is held back by a decorative gold-colored tassel. The lighting is dramatic, highlighting the texture and folds of the fabric.

**FAMOUS
PHYSICIST'S
FAVOURITE
FOODSTUFF**

FOOOOOD



BLAISE PASCAL



PEAS



THANKS!