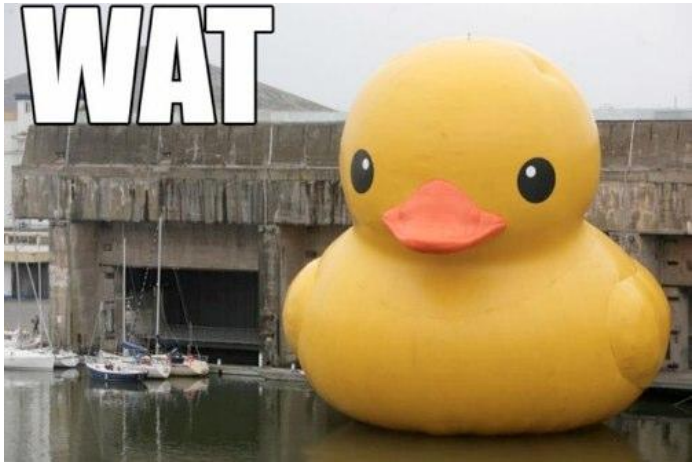
C++ WAT

Piotr Padlewski
piotr.padlewski@gmail.com
University of Warsaw,
intern with Clang team @ Google

C++ WAT

```
assert(map[“Hello world!”] == ‘e’);
```

```
int map = 1;
int t[100];
42[t]; // the same as t[42]
```



C++ WAT

```
int main(){<:]()<%[](){[:>()<%}();}();}();}
```

```
int main(){<:]()<%
            [](){
               [:>()<%}();
            }();
         }();
}
```

```
int main(){[](){
            [](){
               [](){}();
            }();
         }();
}
```

C++ WAT

```
void foo() {

    http://cpp.mimuw.edu.pl/

    printf("WAT??!");

    int n = 5;

    while(n --> 0) {
        //stuff
    }

    return (void)"Everything is fine";
}
```

C++ WAT

```
int main() try {
   // some stuff
}
catch(...)
{
    printf(“something is wrong”);
}
```

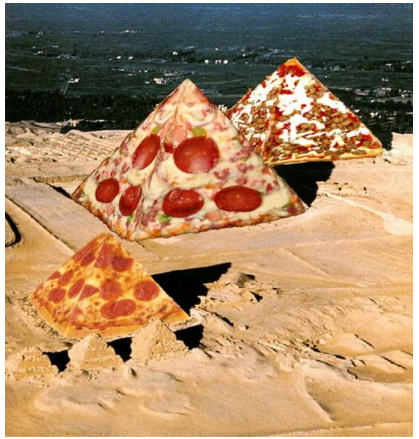

```
struct A : public B
{
    A() try : B()
    {
        // constructor body
    }
    catch (...)
    {

    }
};
```

```
typedef long long ll;

void foo(unsigned ll) {
        std::cout << "1\n";
}
void foo(unsigned long long) {
        std::cout << "2\n";
}
int main() {
        foo(2ull);
}
```

Since signed, unsigned, long, and short by default imply int, a type-name appearing after one of those specifiers is treated as the name being (re)declared."



int's destructor

```
int p;
p.~int(); // won't compile

int main() {
    using int_ = int;

    int_ myAwesomeInt;
    myAwesomeInt.~int_();
}
```

N. Lewycky realloc

Undefined Behavior Consequences Contest

```
#include <stdio.h>
#include <stdlib.h>

int main() {
  int *p = (int*)malloc(sizeof(int));
  int *q = (int*)realloc(p, sizeof(int));
  *p = 1;
  *q = 2;
  if (p == q)
    printf("%d %d\n", *p, *q);
}
```

Compiled with clang will produce:

1 2

self moving

What will happen?

```
std::vector<int> w(42);
w = std::move(w);
```

Undefined Behaviour!

```
std::min(2000000000, 2100000000);
std::min(2000000000, 3000000000);
std::min(2200000000, 3000000000);
```

error: no matching function for call to ‘min(int, long int)’
 std::min(2000000000, 3000000000);

```
int func(int x);
func((1, 2, 3, 4, 5));
```

Optimizations based on UBs

```
int table[4];
bool exists_in_table(int v)
{
    for (int i = 0; i <= 4; i++) {
        if (table[i] == v) return true;
    }
    return false;
}
```

```
int table[4];
bool exists_in_table(int v)
{
    return true;
}
```

fermat

```
int fermat (void)
{
  const int MAX = 1000;
  int a=1,b=1,c=1;
  while (1) {
    if (((a*a*a) == ((b*b*b)+(c*c*c)))) return 1;
    a++;
    if (a>MAX) {
      a=1;
      b++;
    }
    if (b>MAX) {
      b=1;
      c++;
    }
    if (c>MAX) {
      c=1;
    }
  }
  return 0;
}
```

fermat

```c
#include <stdio.h>
int main (void)
{
  if (fermat()) {
    printf ("Fermat's Last Theorem has been disproved.\n");
  } else {
    printf ("Fermat's Last Theorem has not been disproved.\n");
  }
  return 0;
}
```

Fermat's Last Theorem has been disproved.

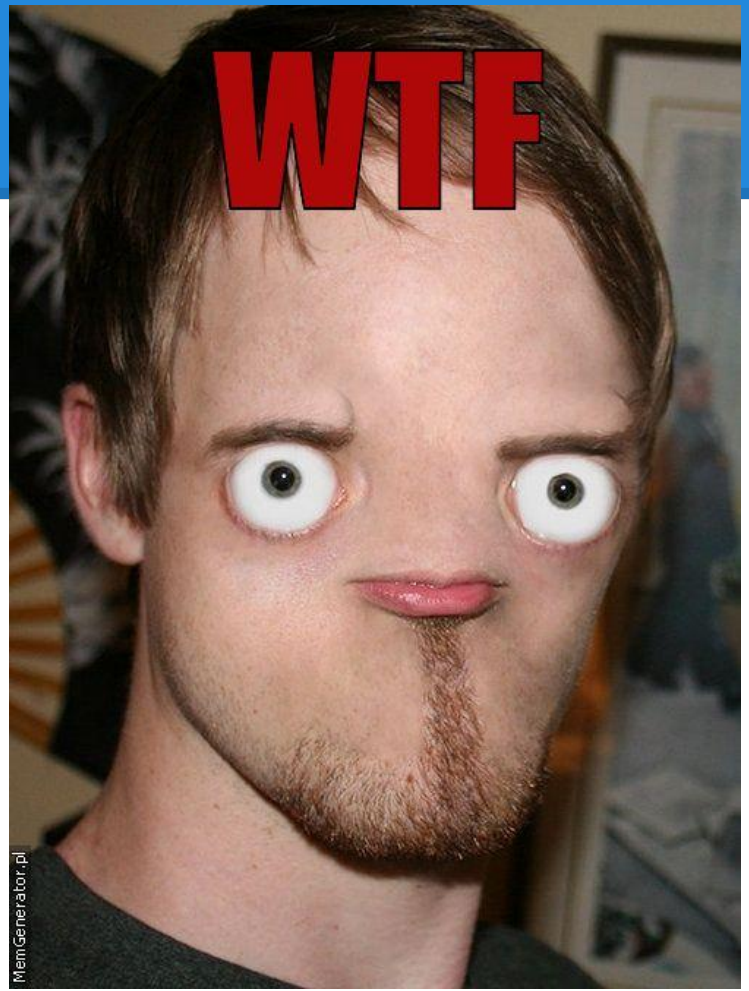
Weird error

```
void fun() {
    std::unordered_set<int64_t> goodVisitors_;
    { //inserting something to goodVisitors_; }


     std::vector <int64_t> goodVisitors(
                                        goodVisitors.begin(),
                                        goodVisitors.end());

     std::sort(goodVisitors.begin(), goodVisitors.end());

 }
```



Weird error

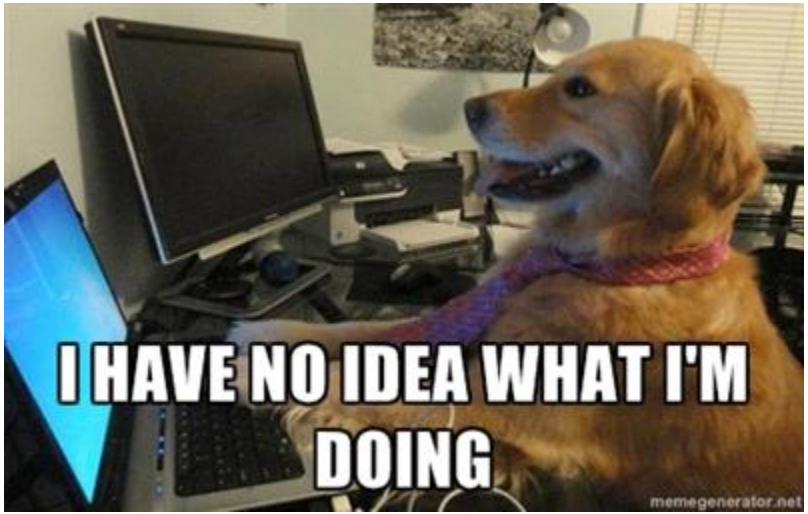
1. Program received signal SIGSEGV, Segmentation fault.
2. __memmove_ssse3_back () at ../sysdeps/x86_64/multiarch/memcpy-ssse3-back.S:2572
3. 2572 ../sysdeps/x86_64/multiarch/memcpy-ssse3-back.S: No such file or directory.

```
#include <cstdio>
int main() {

    long long a = (long long)&a;

    scanf("%lld", a);

    printf("%lld\n", a); //works fine lol
}
```



Braces are evil

```
const char* str = "cppcon";
int main() {
  vector<string> v{{str}};
  std::cout << v.size() << std::endl;
  for (auto &s : v)
    cout << "element [" << s << "]\n";
}
```

Will print:

1
element [cppcon]

Braces are evil

```
const char* str = "cppcon";
int main() {
  vector<string> v{{str, str, str}};
  std::cout << v.size() << std::endl;
  for (auto &s : v)
    cout << "element [" << s << "]\n";
}
```

Will print:

```
3
element [cppcon]
element [cppcon]
element [cppcon]
```

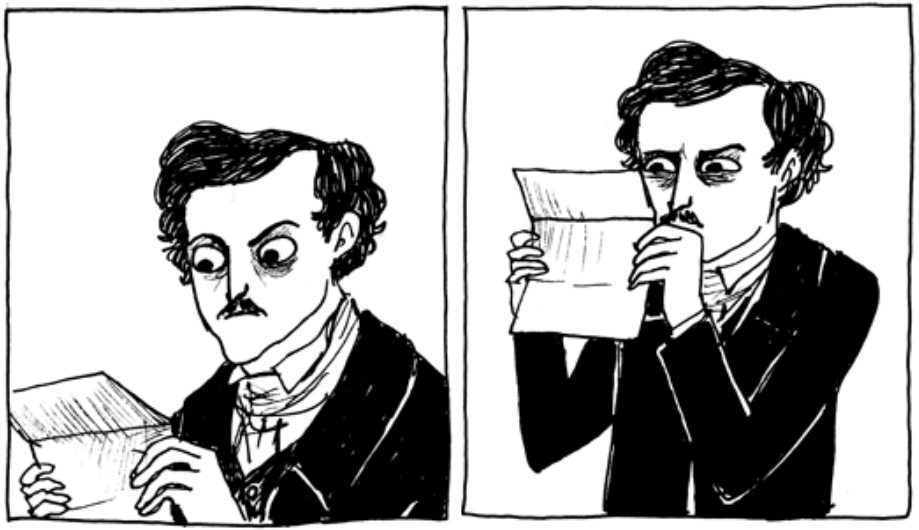
Braces are evil

```
const char* str = "cppcon";
int main() {
  vector<string> v{{str, str}};
  std::cout << v.size() << std::endl;
  for (auto &s : v)
    cout << "element [" << s << "]\n";
}
```

Will print:

1

element []



Braces are evil

```
std::vector<std::string> v{{"testing", "123"}};
```

What programmer wanted:

```
std::vector<std:string> v({"testing", "123"});
std::vector<std::string> v = {"testing", "123"};
```

What compiler did:

```
call vector<string>::vector(initializer_list<string>)
string::string("testing", "123");
// I got this 2 awesome iterators, let me build string for you!
```

Thanks

piotr.padlewski@gmail.com

Sources

http://kukuruku.co/hub/cpp/undefined-behavior-and-fermats-last-theorem

http://cppquiz.org/

http://www.reddit.com/r/cpp/comments/2ycbmj/c_wtfs/

http://blog.regehr.org/archives/767

Extra slides!

Tricky question

How to define function, that it will not be possible to call in any way?

(There will be no syntax to call this function)

Tricky question

```
 struct A {
  template <typename T>
  A() { } // Can’t be called because T can’t be
          // specified or deduced.

  template <typename T>
  operator T() { } //the same as above
};
```

switch for

```
int main() {
    int a = 2;
    int i;
    for (i = 0 ; i < 10 ; i++) {
        switch (a) {
            case 1:
                std::cout << "foo" << std::endl;
                break;
            case 2:
                std::cout << "foo2" << std::endl;
                break;
            default:
                std::cout << "bar" << std::endl;
        }
    }
}
```

```
foo2
foo2
foo2
foo2
foo2
foo2
foo2
foo2
foo2
foo2
```

switch for

```
int main() {
    int a = 2;
    int i;
    switch (a) {
        case 1:
            for (i = 0 ; i < 10 ; i++)
                std::cout << "foo" << std::endl;
            break;
        case 2:
            for (i = 0 ; i < 10 ; i++)
                std::cout << "foo2" << std::endl;
            break;
        default:
            for (i = 0 ; i < 10 ; i++)
                std::cout << "bar" << std::endl;
    }
}
```

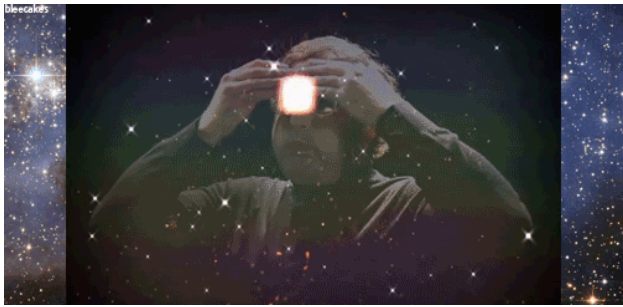
```
foo2
foo2
foo2
foo2
foo2
foo2
foo2
foo2
foo2
foo2
```

switch for

```
int main() {
    int a = 2;
    int i;
    switch (a) {
        for (i = 0 ; i < 10 ; i++) {
            case 1:
                std::cout << "foo" << std::endl;
                continue;
            case 2:
                std::cout << "foo2" << std::endl;
                continue;
            default:
                std::cout << "bar" << std::endl;
        }
    }
}
```

gcc out:

foo2
foo
foo
foo
foo
foo
foo
foo
foo
foo



clang out:

foo2