Cpp Undefined Behavior

2025-07-29

What Code Leads to Undefined Behavior (UB)?

Let's Take a Look at Dangerous Code

Dangerous Code Example 1

```
int* p = nullptr;

int value = *p;
```

Dangerous Code Example 2

```
std::mutex mtx;

mtx.lock();
mtx.lock();
```

Dangerous Code Example 3

```
int a = 10;
int b = 0;

int c = a / b;
```

Dangerous Code Example 4

```
int x;
int y = x + 5;
```

Dangerous Code Example 5

```
int* ptr = new int(10);
delete ptr;

std::cout << *ptr << std::endl;
```

Dangerous Code Example 6

```
int arr[5] = {1, 2, 3, 4, 5};

std::cout << arr[10] << std::endl;
```

Dangerous Code Example 7

```
char* p = "Yello"; // pre c++11

p[0] = 'H';
```

Dangerous Code Example 8

```
int i = 5;

i++ + ++i;
```

Dangerous Code Example 9

```
int f(int a) {
        if (a < 5) return 0;
}
```

Dangerous Code Example 10

```
std::cout << sizeof(char) << std::endl;
```

Examples

- Dereference a nullptr
- Lock twice in one thread
- Divide by zero
- Read uninitialized memory
- Use after free

Examples (cont)

- Array access outside bounds
- Allegedly clever expressions
- ODR violations
- Non-void function without return statement

Effects of UB

- SIGSEV
- Hard Crash
- Deterministic Program yields random results
- Invoking contained but uninvoked code
- Compiler/Linker refuse to build with unrelated errors
- Sudden call to `std::terminate`
- Finite Loops becoming infinite and vice versa
- Starting to play games Link

Definitions

Specification-Defined Behavior

> The programming language completely defines what happens when executing this program.

Ill Formed

> The code has syntax errors or diagnosable semantic errors.

- Compilation fails

Ill Formed, no Diagnostics

> The code has semantic errors which cannot be diagnosed.

- e.g. ODR violations
- This is UB

Implementation-Defined Behavior

> The exact behavior varies between compiler, operating system, or hardware and must be documented.

- E.g. exact number of bits in a `char`

Unspecified Behavior

> The exact behavior varies between compiler, operating system, or hardware (no documentation).

- E.g. order of evaluations

Undefined Behavior

> There are no restrictions on the behavior of the program.

> UB may be expected when the standard omits any explicit definition of behavior

How to Avoid UB

- compiler warnings
- clang-tidy
- cppcheck
- sanitizers

The Compiler's point of view

- UB -> Anything is allowed to happen
- The compiler is a big optimizer
- Aggressive optimizations, including "time travel"

`std::unreachable`

> Invokes undefined behavior at a given point.

- Clearly stating the intent of the code

Example from Our Code

```
assert(false); // assert logs
std::unreachable();
```

Behavior

- with MSVC 17.14.7, c++23

  Both Debug and Release

- Print assert log
- Terminate the program

End of Time-Travel with c++26

- With C++26: Output happening before undefined behavior is guaranteed to happen
- Earlier c++ versions still show the same behavior
- Effectively there is no time travel

References

- C++ programmer's guide to undefined behavior: part 1 of 11
- GCC Easter Egg: C++ Undefined Defined Behavior » Feross.org
- Undefined behavior - Wikipedia
- Dangerous optimizations - C++ meetup - YouTube