

# Code Checking Tools

Today we will talk about tools that will help you find bugs in your code.

- ▶ Valgrind's Memcheck Tool
- ▶ Clang: Not just a compiler

Valgrind is all-in-one, but Clang is (much) faster.

# Stack and Heap

# Stack and Heap

- ▶ The stack (on x86) starts at a high address and grows down
- ▶ The heap (on x86) starts at the bottom and grows up
- ▶ Destructors on stack-allocated class instances are called when the function returns
- ▶ Destructors on heap-allocated class instances are called when `delete` is called on the pointer

## Uninitialized Values (valgrind and memory-sanitizer)

- ▶ Reading a value that hasn't been initialized from the stack or the heap.
- ▶ Especially dangerous when program flow depends on that value.

## Uninitialized Values (valgrind and memory-sanitizer)

- ▶ Reading a value that hasn't been initialized from the stack or the heap.
- ▶ Especially dangerous when program flow depends on that value.
- ▶ `valgrind -track-origins=yes` Slower, but keeps track of where uninitialized values were allocated.
- ▶ `source symbolizer.sh`
- ▶ `clang++ -fsanitize=memory`  
`-fsanitize-memory-track-origins`

# Invalid Reads and Writes (valgrind and address-sanitizer)

- ▶ Reading or writing values from unallocated memory.
- ▶ Sometimes may result in a segfault, but not always.

# Invalid Reads and Writes (valgrind and address-sanitizer)

- ▶ Reading or writing values from unallocated memory.
- ▶ Sometimes may result in a segfault, but not always.
- ▶ Valgrind isn't perfect: you can read and write to things on the stack without complaint.
- ▶ `clang++ -fsanitize=address`

## Invalid and Mismatched deletes (valgrind and address-sanitizer)

- ▶ Mismatched delete: using `delete` with `new[]` or vice versa.
- ▶ Double delete: deleting the same memory twice



# Memory Leaks (valgrind)

- ▶ Valgrind runs leak checks after the program terminates.
- ▶ Directly lost: No pointer to that block anymore.
- ▶ Indirectly lost: A pointer to that block exists, but it's in a directly lost block.
- ▶ Still reachable: Still have a pointer to that block.
- ▶ Possibly lost: No pointer to the beginning of the block, but a pointer to somewhere inside the block.
- ▶ `valgrind -leak-check=full`
- ▶ Valgrind Memcheck Manual