University of Antwerp
Faculty of Science

# C++ Tracing Exercises

Advanced Programming
2022-2023

## Presentation Overview

1. What is the purpose of these exercises?
2. How to complete such an exercise?
3. How can you practice for the exam?
4. How will the exam exercise work?

## Exercise purpose

What:

- Given the following code, what is the output?
- Expected output corresponds to the execution trace
- Not allowed to actually execute it

Need to know:

- What is being executed?
- When is it executed?

## Simplified exercise example

Code:

```
int app_example()
{
        FUNCTION_TRACER;
        LOG_TRACER("statement: Engine e(10);");
         Engine e(10);
         return 0;
}
```

Output:

```
---> function body: int app_example()
     statement: Engine e(10);
     ---> member for object: 0x7fff15ad6840 : Vehicle::Engine::Engine(double)
     <--- member for object: 0x7fff15ad6840 : Vehicle::Engine::Engine(double)
     ---> member for object: 0x7fff15ad6840 : Vehicle::Engine::~Engine()
     <--- member for object: 0x7fff15ad6840 : Vehicle::Engine::~Engine()
<--- function body: int app_example()
```

## Possible Macro’s

FUNCTION_TRACER:

- Placed at the start of a function

BLOCK_TRACER:

- Placed at the start of a compound statement (Block)

LOG_TRACER:

- Provide a manual entry to the output (print statement)

MEMBER_TRACER:

- Placed at the start of a member function

## Important areas to practice

- Constructor types
- Assignment operators
- Inheritance
- Exception handling
- Conversions
- Member initialization
- Memory management
- Function and operator overloading
- Copy Elision & Return Value Optimization (RVO)

## Using the object tracer

```
git clone https://github.com/Advanced-Programming-UA/Object-Tracer
cd Object-Tracer
mkdir build
cd build
cmake ..
make install
./installed/bin/tracer
```

## Using the object tracer

```
The available exercises are:

copy
essentialops
exam_2021_1
exam_2021_2
exam_2022_1
exam_2022_2
exception
fcalls
hierarchies
poly1
poly2

Select a demo by name:
```

# Using the object tracer

```
Select a demo by name: copy

---> function body: int app_copy()
        statement: shared_ptr<Engine> e1Ptr(new Engine(100));
        ---> member for object: 0x55a3b01dde20 : Vehicles::Engine::Engine(double)
        <--- member for object: 0x55a3b01dde20 : Vehicles::Engine::Engine(double)
        statement: Person* p1Ptr = new Person("Owner 1");
        ---> member for object: 0x55a3b01dde40 : Vehicles::Person::Person(std::string)
        <--- member for object: 0x55a3b01dde40 : Vehicles::Person::Person(std::string)
        statement: Motorcycle* m1Ptr = new Motorcycle(e1Ptr, p1Ptr);
        ---> member for object: 0x55a3b01de100 : Vehicles::Body::Body(std::string)
        <--- member for object: 0x55a3b01de100 : Vehicles::Body::Body(std::string)
        ---> member for object: 0x55a3b01de0a0 : Vehicles::Wheel::Wheel()
        <--- member for object: 0x55a3b01de0a0 : Vehicles::Wheel::Wheel()
        ---> member for object: 0x55a3b01de0b0 : Vehicles::Wheel::Wheel()
        <--- member for object: 0x55a3b01de0b0 : Vehicles::Wheel::Wheel()
```

(…)

## Using the object tracer

- Example exercises are located at:

```
src/exercises/*
```

- Practice using these provided examples!
- Edit these examples or follow these instructions to add your own!
- Don’t underestimate these exercises, you’ll be surprised by some of the outcomes

# Using the object tracer

- You'll get the doxygen documentation of prog2, including some new files
- We'll give you the code that needs to be traced
- Your task is to produce the expected log as if the prog2 object tracer was run

## This Session

- Try to make the exercise from the January 2022 exam
  (src/exercises/app_exam_2022_1.cpp)
- Verify your own solution using the object tracer
  (choose exam_2022_1 demo)
- You likely won’t finish it today, but hopefully understand how it works
  - Simply make sure to practise it enough for the exam

## Questions

- Are there any questions?
- Send an e-mail to Thomas.Ave@uantwerpen.be