# Exercise 7

**Programming SS 2019 - Problem Set 3**

Author: *Elena Pfefferlé*, *Pascal Schenk*

## General:

In this code we have a class `Trafficlight`, this class features 3 constructors (1 default, 1 copy constructible and 1 copy assignable), 1 destructor and 3 attributes:

- `static int counter_` : This is a `static` variable, meaning that only one can exists no matter how many class objects are created.
- `const int id_` : This one stores the `id` of the class object.
- `light state` : This variable of type `light` (*enum*) stores the light *color* of the object.

**Line 1 - Trafficlight a(Trafficlight :: green);**

Here we create a new object `a` of class `Trafficlight`, we pass `Trafficlight::green` as argument, so `a.light == green` therefore `a.state_ = green`. `counter_` is passed to `a.id_` and increased by one.

- **guessed output** : “Constructing 0”
- `a.id_` = 0
- `a.state_` = green

**Line 2 - Trafficlight* b_ptr = new Trafficlight(Trafficlight::orange);**

Here we do two things, firstly we declare an `object pointer` with the syntax `Trafficight* b_ptr`. Then we create a `dynamic object` with the syntax `new` and store its address into the `object pointer`.

- **guessed output** : " Constructing 1"
- `b_ptr.id_` = 1
- `b_ptr.state_` = orange

**Line 3 - Trafficlight c(a);**

Here we create a new object `c` of class `Trafficlight` but based on `a`.

- **guessed output** : “Copy constructing 2 from 0”
- `c.id_` = 2
- `c.state_` = green

**Line 4 - Trafficlight d = `*b_ptr` ;**

Here we create a new object `d` based on `b_ptr`.

- **guessed output** : “Copy constructing 3 from 1”
- `d.id_` = 3
- `c.state_` = orange

**Line 5 / 6 - Trafficlight e; e = a;**

Here we firstly create an empty `Trafficlight` class object `e`.

- **guessed output** : “Constructing 4”
- `e.id_` = 4
- `e.state_` = red

The we reconstruct `e` based on `a`.

- **guessed output** : “Assigning 0 to 4”
- `e.id_` = 4
- `e.state_` = green

**Line 7 - Trafficlight& f = a;**

For this one, we are simply creating a reference `f` to `a`. We are not creating a new object and therefore not calling its constructor.

- **guessed output"** : `nothing`
- `f.id_` refers to `a.id_`
- `f.state_` refers to `f.state_`

**Line 8 - a.print_state();**

- **guessed output** : green

**Line 9 - b->print_state();**

Here we need to access the function behind the pointer with `->`

- **guessed output** : “orange”

**Line 10 - c.print_state();**

- **guessed output** : “green”

**Line 11 - d.print_state();**

- **guessed output** : “orange”

**Line 12 - e.print_state();**

- **guessed output** : “green”

**Line 13 - f.print_state();**

Here we are actually calling `a.print_state()`.

- **guessed output** : “green”

**Line 14 - delete b_ptr;**

Here we are calling `delete` on `b_ptr` , this mean that we call the class deconstructor `~Trafficight()` . However since `b_ptr` is an `object pointer` pointing on `class Trafficlight` it calls the deconstructor of all instances of that class.

- **guessed output** : “Deconstructing 1 Deconstructing 3 Deconstructing 4 Deconstructing 2 Deconstructing 0”
- **note** : I am unsure about the order.

# Guessed output

```
Constructing 0
Constructing 1
Copy constructing 2 from 0
Copy constructing 3 from 1
Constructing 4
Assigning 0 to 4
green
orange
green
orange
green
green
Deconstructing 1
Deconstructing 3
Deconstructing 4
Deconstructing 2
Deconstructing 0
```