

## Solution 4: Object creation

ETH Zurich

### 1 Creating objects in Traffic

#### Solution

Listing 1: Class `OBJECT_CREATION`

```
1 indexing
   description: "Creation class (Assignment 4)"
3   date: "$Date$"
   revision: "$Revision$"
5
   class
7   OBJECT_CREATION
9 inherit
11  TOURISM
13 feature -- Explore Paris
15   explore is
      -- Create new objects for Paris.
17   do
      Paris.display
19
      create passenger.make_with_route (Route3, 1.5)
21     passenger.go
      passenger.set_reiterate (True)
23     Paris.put_passenger (passenger)
25
      create tram.make_with_line (Line1)
      tram.start
27     Paris.put_tram (tram)
29
      create coordinate.make (station_gare_de_lyon.location.x, station_gare_de_lyon.location
        .y)
      create landmark.make (coordinate, "Gare de lyon", "train_station.png")
31     Paris.put_landmark (landmark)
33
      create point_randomizer.make (Paris.center, Paris.radius)
      point_randomizer.generate_point_array (5)
35
      create free_moving.make_with_points (point_randomizer.last_array, 10.0)
```

```
37  free_moving.start
    Paris.put_free_moving (free_moving)
39
    point_randomizer.generate_point_array (10)
41
    create taxi.make_random (point_randomizer.last_array)
43    taxi.start
    Paris.put_taxi (taxi)
45
47  create line_type.make
    create tourist_line .make_with_terminal ("Tourist line", line_type,
        station_gare_de_lyon)
49  create color.make_with_rgb (255, 160, 0)
    tourist_line .set_color (color)
51    tourist_line .extend (station_st_michel_notre_dame)
    tourist_line .extend (station_champs_de_mars_tour_eiffel_bir_hakeim)
53    tourist_line .extend ( station_charles_de_gaulle_etoile )
    tourist_line .extend ( station_palais_royal_musee_du_lowvre )
55    Paris.put_line ( tourist_line )
57
    create bus.make_with_line ( tourist_line )
    bus.start
59    bus.set_reiterate (True)
    Paris.put_bus (bus)
61
    end
63
    passenger: TRAFFIC_PASSENGER
65    -- Passenger moving along Route3
67    tram: TRAFFIC_TRAM
    -- Tram
69
    landmark: TRAFFIC_LANDMARK
71    -- Landmark
73
    tourist_line : TRAFFIC_LINE
    -- Tourist bus line
75
    line_type : TRAFFIC_TYPE_BUS
77    -- Bus type
79
    bus: TRAFFIC_BUS
    -- Bus for tourist_line
81
    coordinate: TRAFFIC_POINT
83    -- Coordinate for landmarks
85
    taxi: TRAFFIC_TAXI
    -- Taxi
87
```

```
89  point_randomizer: TRAFFIC_POINT_RANDOMIZER
    -- Point list generator
91  color: TRAFFIC_COLOR
    -- Color for tourist_line
93
95  free_moving: TRAFFIC_FREE_MOVING
    -- Free moving
97 end
```

## 2 It's Logic!

Read Touch of class, paragraph 5.3. Here are some examples:

- **if** ( $x \geq 0$ ) **and** ( $x \leq 10$ ) **then** ... **end**
- **if** ( $x \geq 0$ ) **and then** ( $x.square\_root \geq 5$ ) **then** ... **end**
- **if** ( $x < 0$ ) **or** ( $x > 10$ ) **then** ... **end**
- **if** ( $x < 0$ ) **or else** ( $x.square\_root < 5$ ) **then** ... **end**

### Solution

1. Semi-strict boolean operators are non-commutative, that is, the order in which their operands appear is relevant. In the case of Eiffel, boolean expressions using the semi-strict operators are evaluated from left to right. Strict boolean operators are commutative. In Eiffel it is not specified which side of such an expression is evaluated first.
2. Semi-strict boolean operators need to be used if the evaluation of the right-hand side is depending on the evaluation of the left-hand side. Strict operators allow the compiler to optimize the evaluation of the expressions, so they should be preferred over semi-strict operators if the order of evaluation does not matter.

Examples:

- **if** ( $student\_number > 0$ ) **and** ( $teacher.is\_in\_class$ ) **then**  $teacher.give\_lecture$  **end**
- **if** ( $is\_file\_read\_ok$ ) **and then**  $num\_items\_read > 13$  **then**  $further\_compute$  **end**
- **if** ( $student\_number = 0$ ) **or** (**not**  $teacher.is\_in\_class$ ) **then**  $wait$  **end**
- **if** (**not**  $is\_file\_read\_ok$ ) **or else**  $num\_items\_read \leq 13$  **then**  $display\_error\_message$  **end**

## 3 Temperature application

### Solution

Listing 2: Class *TEMPERATURE*

**indexing**

```
2  description: "Objects that represent a temperature in Celsius or Kelvin."
```

```
4 class
    TEMPERATURE
6
    create
8    make_celsius, make_kelvin
10 feature -- Initialization
12    make_celsius (a_value: INTEGER) is
        -- Create with Celsius value.
14        require
            a_value_above_absolute_zero : a_value >= -273
16        do
            celsius_value := a_value
18        ensure
            value_set: celsius_value = a_value
20        end
22    make_kelvin (a_value: INTEGER) is
        -- Create with Kelvin value.
24        require
            a_value_above_absolute_zero : a_value >= 0
26        do
            celsius_value := a_value - 273
28        ensure
            value_set: kelvin_value = a_value
30        end
32 feature -- Access
34    celsius_value : INTEGER
        -- Temperature value in Celsius.
36
38    kelvin_value : INTEGER
        -- Temperature unit in Kelvin.
40    do
        Result := celsius_value + 273
        ensure
42        Result = celsius_value + 273
        end
44
46    feature -- Basic operations
48    sum_of_celsius_values (other: TEMPERATURE): INTEGER
        -- Return the sum of 'other' and 'Current' (in Celsius).
        require
50        celsius_value + other.celsius_value >= -273
        do
52        Result := celsius_value + other.celsius_value
        ensure
54        celsius_value_set : Result = celsius_value + other.celsius_value
        end
```

```
56 invariant  
58     value_above_absolute_zero : ( celsius_value >= -273) and (kelvin_value >= 0)  
60 end
```

Listing 3: Class `TEMPERATURE_APPLICATION`

```
1 indexing  
   description : "An application to convert temperatures between different units."  
3  
4 class  
5   TEMPERATURE_APPLICATION  
6  
7 create  
   make  
9  
10 feature -- Initialization  
11  
12   temperature_1: TEMPERATURE  
13     -- First temperature  
14  
15   temperature_2: TEMPERATURE  
16     -- Second temperature  
17  
18   temperature_3: TEMPERATURE  
19     -- Third temperature  
20  
21 make is  
   -- Creation procedure  
22 do  
   -- Print some welcome text.  
23   io.put_string("TEMPERATURE CONVERTER V1.0")  
   io.put_new_line  
24   io.put_string("=====  
25   io.put_new_line  
26   io.put_new_line  
27  
28   -- Input temperature in Celsius and show the converted value in Kelvin.  
29   io.put_string("Please enter first temperature in Celsius: ")  
30   io.read_integer  
31   if io.last_integer < -273 then  
32     io.put_string ("The input temperature in Celsius should be >= -273")  
33   else  
34     create temperature_1.make_celsius (io.last_integer)  
35     io.put_new_line  
36     io.put_string ("First temperature in Kelvin: ")  
37     io.put_integer (temperature_1.kelvin_value)  
38   end  
39  
40   io.put_new_line  
41   io.put_new_line
```

```
45      -- Input temperature in Kelvin and show the converted value in Celsius.
46      io.put_string("Please enter second temperature in Kelvin: ")
47      io.read_integer
48      if io.last_integer < 0 then
49          io.put_string ("The input temperature in Kelvin should be >= 0")
50      else
51          create temperature_2.make_kelvin (io.last_integer)
52          io.put_new_line
53          io.put_string ("Second temperature in Celsius: ")
54          io.put_integer (temperature_2.celsius_value)
55
56      end
57
58      -- Add both temperatures in Celsius and show the result in both Celsius and Kelvin
59      if temperature_1 /= Void and temperature_2 /= Void then
60          io.put_new_line
61          io.put_new_line
62          if temperature_1.celsius_value + temperature_2.celsius_value < -273 then
63              io.put_string ("The two temperatures cannot be summed up!")
64          else
65              io.put_string ("Adding second temperature to first...")
66              create temperature_3.make_celsius (temperature_1.sum_of_celsius_values (
67                  temperature_2))
68              io.put_new_line
69              io.put_new_line
70              io.put_string ("Resulting temperature in Celsius: ")
71              io.put_integer (temperature_3.celsius_value)
72              io.put_new_line
73              io.put_new_line
74              io.put_string ("Resulting temperature in Kelvin: ")
75              io.put_integer (temperature_3.kelvin_value)
76              io.put_new_line
77          end
78      end
79
80      end
81 end
```