Assignment 6: SCOOP type system

ETH Zurich

1 Subtyping

1.1 Background

Have a look at the attributes shown in listing 1.

Listing 1: Attributes

1 \(px\): \textit{PROCESSOR}
2 \(py\): \textit{PROCESSOR}
3 \(a\): \textit{separate} \(X\)
4 \(b\): \textit{separate} \(<px>\ X\)
5 \(c\): \textit{separate} \(<py>\ X\)
6 \(d\): \(X\)
7 \(e\): \textit{detachable separate} \(X\)
8 \(f\): \textit{detachable separate} \(<px>\ X\)
9 \(g\): \textit{detachable} \(X\)

1.2 Task

Decide whether the following attachments are valid or not. Justify your answer.

1. \(a := b\)
2. \(a := d\)
3. \(b := a\)
4. \(b := c\)
5. \(b := d\)
6. \(d := a\)
7. \(d := b\)
8. \(a := e\)
9. \(e := a\)
2 Valid targets

2.1 Background

Have a look at listing 2.

Listing 2: Enclosing Feature

```plaintext
p: PROCESSOR
2
r (a: detachable separate X; b: separate <p> X; c: separate X)
4  local
d: separate <p> X
6  e: separate <c.handler> X
  f: separate X
8  do
10  end
```

Imagine that the class X has a function g: X and a procedure do_something.

2.2 Task

Decide for each of the following feature calls, whether the calls are valid or not when they appear in feature r of listing 2.

1. c.do_something
2. c.g.do_something
3. e := c; e.do_something
4. f := c; f.do_something
5. a.do_something
6. d := b; d.do_something

3 Separate generics or generic separate?

3.1 Background

The interplay between generics and separate types are important to understand, and enforce a good understanding of the type system.

3.2 Task

Consider the differences between:

- `separate LIST [BOOK]`
- `LIST [separate BOOK]`

Explain the distinction using the object/processor diagram.
4 Basic library: type combiner

4.1 Background

Consider the classes in listing 3. These classes belong to a basic library implementation.

Listing 3: Basic Library

```plaintext
class LIST[G]
  feature
    last : G
      -- Last element.
    put(a_element: G)
      -- Add the element to the list.
  end
end

class LIBRARY
  feature
  end
end
```

4.2 Task

What is the result type of `books.last` from the perspective of the library? What is the type of an actual argument in the call `books.put(...)` from the perspective of the library? Justify your answer.

5 Stack library: type combiner

5.1 Background

Consider the alternative stack based library implementation shown in listing 4.

Listing 4: Stack Library

```plaintext
class LIST[G]
  feature
    last : G -- Last element.
  end

class STACK[G]
  feature
    top : G -- Top element.
  end

class LIBRARY
  feature
    books : LIST[STACK[separate BOOK]] -- Books.
  end
end
```
5.2 Task
What is the result type of $books.last.top$ from the perspective of the library? Justify your answer.