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: PROCESSOR
    py: PROCESSOR
3
    a: separate X
5 b: separate <px> X
    c: separate <py> X
7 d: X
    e: detachable separate X
9 f: detachable separate <px> X
    g: 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 `twin` like `Current` and a procedure `do_something`. You can assume that the type of `c.twin` is attached and that its class type is `X`. You can also assume that the type of `c.twin` denotes that `c.twin` is on the same processor as `c`.

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.twin.do_something`
3. `e := c.twin; 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(element: G)
      -- Add the element to the list.
  end
end

class LIBRARY
  feature
  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
end

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

class LIBRARY
  feature
    books : LIST[STACK[separate BOOK]] -- Books.
  end
```
5.2 Task

What is the result type of $books.last.top$ from the perspective of the library? Justify your answer.