National Undergraduate Programme in Mathematical Sciences

National Graduate Programme in Computer Science

Programming Language Concepts

Final Examination, II Semester, 2018–2019

Date : 23 April, 2019 | Marks : 30

Duration : Two hours | Weightage : 30%

- *Answer all questions.*
- *You can freely use the standard Church encoding for functions.*

$$
\begin{aligned}
f^0 x &= x \\
f^{n+1} x &= f(f^n x) \\
\boldsymbol{n} &= (\lambda fx.f^n x) \\
\boldsymbol{succ} &= (\lambda pfx.f(pfx)) \\
\boldsymbol{add} &= (\lambda pqfx.pf(qfx)) \\
\boldsymbol{mult} &= (\lambda pqf.p(qf)) \\
\boldsymbol{true} &= (\lambda xy.x) \\
\boldsymbol{false} &= (\lambda xy.y) \\
\boldsymbol{pair} &= (\lambda xyw.wxy) \\
\boldsymbol{fst} &= (\lambda p.p\ \boldsymbol{true}) \\
\boldsymbol{snd} &= (\lambda p.p\ \boldsymbol{false}) \\
\boldsymbol{ite} &= (\lambda bxy.bxy) \\
\boldsymbol{iszero} &= (\lambda x.(x(\lambda z.\boldsymbol{false}))\ \boldsymbol{true})
\end{aligned}
$$

---

1. (a) Define a λ-expression **pred** such that for all $m \in \mathbb{N}$:

$$
\mathbf{pred\ m} \xrightarrow{*}_{\beta} \begin{cases} \mathbf{0} & \text{if } m = 0 \\ \mathbf{k} & \text{if } m > 0 \text{ and } k = m - 1 \end{cases}
$$

(b) Use **pred** to define a λ-expression **subtract** such that for all $m, n \in \mathbb{N}$:

$$
\mathbf{subtract\ m\ n} \xrightarrow{*}_{\beta} \begin{cases} \mathbf{0} & \text{if } m \leq n \\ \mathbf{p} & \text{if } m > n \text{ and } p = m - n \end{cases}
$$

(c) Define a λ-expression **not** such that for all expressions M representing a boolean value b:

$$
(\mathbf{not}\ M) \xrightarrow{*}_{\beta} \begin{cases} \mathbf{false} & \text{if } M \xrightarrow{*}_{\beta} \mathbf{true} \\ \mathbf{true} & \text{if } M \xrightarrow{*}_{\beta} \mathbf{false} \end{cases}
$$

(d) Define a λ-expression **even** such that for all $m \in \mathbb{N}$:

$$
\mathbf{even\ m} \xrightarrow{*}_{\beta} \begin{cases} \mathbf{true} & \text{if } m \text{ is even} \\ \mathbf{false} & \text{if } m \text{ is odd} \end{cases}
$$

(e) Recall that the Fibonacci numbers are given by the following recurrence:

$$F_0 = 0$$
$$F_1 = 1$$
$$F_{n+2} = F_n + F_{n+1}$$

Define a λ-expression **fib** such that for all $m \in \mathbb{N}$:

$$\mathbf{fib\ m} \xrightarrow{*}_\beta \mathbf{k} \quad \text{iff} \quad k = F_m.$$

(f) Define a λ-expression **fac** such that for all $m \in \mathbb{N}$:

$$\mathbf{fac\ m} \xrightarrow{*}_\beta \mathbf{k} \quad \text{iff} \quad k = m!.$$

(18 marks)

2. Derive the most general type of the following expressions by following the typing algorithm presented in class. (Inductively define a type pattern and system of equations for each subexpression as presented in class, and solve the equations.)

(a) **let** $\{\mathbf{true} = \lambda xy.x;\ \mathbf{false} = \lambda xy.y\}$ **in** $(\lambda x.(x(\lambda z.\mathbf{false}))\ \mathbf{true})$

(b) **letrec** $\{t = \lambda f.f(tf);\ i = \lambda x.x\}$ **in** $t(ti)$

(8 marks)

3. Consider the following code:

```
public class C {
    private int first = 0;
    private int second = 0;

    public void a(){
        first++;
        second++;
        if (second == 2 && first != 2) {
            launch_missiles();
        }
    }

    public void b(){
        first++;
        second++;
    }
}
```

Let c be an object of type C. Suppose that one thread calls `c.a()` and, in parallel, another thread calls `c.b()`. Is it possible for the `launch_missiles()` call to be made? If so, show an execution trace that leads to the call. If not, give a supporting argument. *(4 marks)*