INFSEN02-1 Sample exam

The INFDEV@HR Team

1 Exam

1.1 Question 1

Given the following lambda program, and a series of relevant delta rules, show the beta reductions for this program.

```
if FALSE then 1 else 0
```

1.2 Relevant delta rules

If-then-else

```
(λp th el→((p th) el))
```

Boolean false

```
(λt f→f)
```

Integer one (0)

```
(λs z→z)
```

Integer one (1)

```
(λs z→(s z))
```

1.3 Answer 1 (note: you do not need to write all this detail yourself, it is only included for completeness)

```
if FALSE then 1 else 0
```

```
(((if-then-else FALSE) 1) 0)
```

```
((( (λp th el→((p th) el)) FALSE) 1) 0)
```

```
((((λp th el→((p th) el)) FALSE) 1) 0)
```

```
((((λp th el→((p th) el)) (λt f→f) ) 1) 0)
```

```
((((λp th el→((p th) el)) (λt f→f)) 1) 0)
```

```
((((λp th el→((p th) el)) (λt f→f)) (λs z→(s z)) ) 0)
```

```
((((λp th el→((p th) el)) (λt f→f)) (λs z→(s z))) 0)
```

```
((((λp th el→((p th) el)) (λt f→f)) (λs z→(s z)))
    (λs z→z) )
```

```
((((λp th el→((p th) el)) (λt f→f)) (λs z→(s z))) (λs z→z))
```

```
(((λth el→(( (λt f→f)  th) el)) (λs z→(s z))) (λs z→z))
```

```
(((λth el→(((λt f→f) th) el)) (λs z→(s z))) (λs z→z))
```

```
((λel→(((λt f→f) (λs z→(s z)) ) el)) (λs z→z))
```

```
((λel→(((λt f→f) (λs z→(s z))) el)) (λs z→z))
```

```
(((λt f→f) (λs z→(s z))) (λs z→z) )
```

```
(((λt f→f) (λs z→(s z))) (λs z→z))
```

```
((λf→f) (λs z→z))
```

```
((λf→f) (λs z→z))
```

```
(λs z→z)
```

```
(λs z→z)
```

```
0
```

1.4 Question 2

Given the following lambda calculus program, and a series of relevant delta rules, give the full typing derivation for the program.

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((p β) th) el))
```

1.5 Relevant delta rules

Boolean type:

```
(∀α ⇒((α→α)→α→α))
```

1.6 Answer 2 (note: you do not need to write all this detail yourself, it is only included for completeness)

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((p β) th) el))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((p β) th) el))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((Boolean β) th) el))
```

```
Λβ⇒(λ(p:Boolean)(th:β)(el:β)→(((Boolean β) th) el))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((Boolean β) β) el))
```

```
Λβ⇒(λ(p:Boolean) (th:β)(el:β)→(((Boolean β) β) el))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((Boolean β) β) β))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((Boolean β) β) β))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→((((∀α ⇒(α→α→α)) β) β) β))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→((((∀α ⇒(α→α→α)) β) β) β))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((β→β→β) β) β))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→(((β→β→β) β) β))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→((β→β) β))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→((β→β) β))
```

```
Λβ⇒(λ(p:Boolean) (th:β) (el:β)→β)
```

```
Λβ⇒(λ(p:Boolean) (th:β)(el:β)→β)
```

$\Lambda\beta\Rightarrow$(λ(p:Boolean) (th:β)→ ($\beta\rightarrow\beta$))

$\Lambda\beta\Rightarrow$(λ(p:Boolean)$\underline{\text{(th:}\beta\text{)}}$→$\underline{(\beta\rightarrow\beta)}$))

$\Lambda\beta\Rightarrow$(λ(p:Boolean)→ ($\beta\rightarrow\beta\rightarrow\beta$))

$\Lambda\beta\Rightarrow\underline{\text{(λ(p:Boolean)→(}\beta\rightarrow\beta\rightarrow\beta\text{))}}$

$\Lambda\beta\Rightarrow$ (Boolean→$\beta\rightarrow\beta\rightarrow\beta$)

$\underline{\Lambda\beta\Rightarrow\text{(Boolean}\rightarrow\beta\rightarrow\beta\rightarrow\beta\text{)}}$

($\forall\beta$ $\Rightarrow$(Boolean→$\beta\rightarrow\beta\rightarrow\beta$))