CMPUT 325 LEC B1 - Winter 2021 - NON-PROCEDURAL PROG LANGUAGES

Orders of Reduction

Essentially, applicative order reduction requires that the arguments be evaluated before a function is applied. It applies the (leftmost) innermost function first.

Suppose P is

```
(
```

```
  (f (X Y) =  (+ X Y))
  (g (X) =   (+ 1 X))
)
```

Consider applicative order reduction of (f (g 2) (g 1)).

```
  (f (g 2) (g 1))
=> (f (+ 1 2) (g 1))
=> (f 3 (g 1))
=> (f 3 (+ 1 1))
=> (f 3 2)
=> (+ 2 3)
=> 5
```

In contrast, normal order reduction applies the outermost leftmost applicable function first.

```
  (f (g 2) (g 1))
=> (+ (g 2) (g 1)) ; f is applied even if its arguments have not been evaluated
=> (+ (+ 1 2) (g 1))
=> (+ 3 (g 1))
=> (+ 3 (+ 1 1))
=> (+ 3 2)
=> 5
```

Normal order reduction has better termination property.

Example.

Let program be

```
(
  (g (X) = (+ X (g (+ X 1)))
  (f (X Y) = (if (eq X 0) 0 Y))
)
```

Normal order:

```
(f 0 (g 1))
=> (if (eq 0 0) 0 (g 1))
=> (if T 0 (g 1))
=> 0
```

Applicative order:

```
(f 0 (g 1))
=> (f 0 (+ 1 (g (+ 1 1))))
=> (f 0 (+ 1 (g 2)))
=> (f 0 (+ 1 (+ 1 (g (+ 2 1)))))
=> ......
```

Last modified: Wednesday, 23 December 2020, 2:33 PM

◄ Examples of Primitive Functions

Jump to...

Sample Calls to Function fl-interp ►