Recursion and Efficiency

1 Recursive Functions

A *recursive* function is one which calls itself. This is an incredibly powerful concept in functional languages. The syntax for this is: `let rec` ***`name arguement1 arguement2`*** `... =` ***`expression`***. When definining a recursive function, it is important that you set a base case such that the function doesn't continue to call itself with no end. For example:

```
factorial : int -> int

let rec factorial x =
  if x < 0 then 0 else
    if x = 0 then 1 else
      x * factorial (x - 1)
```