Types & Programming Languages
Exercises 2

These exercises are based on the material in Lectures 1, 2, 3 and 4.

1. Prove by induction that for all $n \in N$, $0+1+4+\cdots+n^2 = \frac{n(n+1)(2n+1)}{6}$. Follow the structure of the proof in Lecture 3 Slide 16.

2. Fill in the details in the proof of Type Preservation (Exercise, Lecture 3 Slide 17).

3. *Call by name* means that the only reduction rule for function applications is the rule on Lecture 4 Slide 22. What are the advantages and disadvantages of call by name compared with call by value, from the point of view of practical programming? Hint: think about the function definitions

   ```
   f(x:int):int is 3
   ```

   ```
   g(x:int):int is x+x
   ```

   and the applications

   ```
   f(2+2)
   g(2+2)
   ```

   What would you want to do in a language implementation?

4. The operational semantics we have defined is sometimes called *small step* operational semantics. The alternative is *big step* operational semantics, which directly associates a final value with each expression by means of the following inductive rules. $e \Downarrow v$ means that the result of evaluating the expression e is the value v.

   - If v is a value then $v \Downarrow v$.
   - $\dfrac{e \Downarrow u \quad f \Downarrow v \quad w \text{ is the sum of } u \text{ and } v}{e + f \Downarrow w}$
   - Similar rules for $==$ and $\&$.
   - $\dfrac{c \Downarrow \mathsf{true} \quad e \Downarrow v}{\mathsf{if}\ c\ \mathsf{then}\ e\ \mathsf{else}\ e' \Downarrow v}$ and $\dfrac{c \Downarrow \mathsf{false} \quad e' \Downarrow v}{\mathsf{if}\ c\ \mathsf{then}\ e\ \mathsf{else}\ e' \Downarrow v}$

   (a) Prove by induction on e that if $e \to e'$ and $e' \Downarrow v$ then $e \Downarrow v$.

   (b) Prove that if $e \to^* v$ then $e \Downarrow v$. Use induction on the length of the reduction sequence $e \to^* v$, and the previous result.

   (c) Prove by induction on e that if $e \Downarrow v$ then $e \to^* v$.