**HASKELL-declarations**

\[ \text{decl} \rightarrow \text{typedecl} \mid \text{fundecl} \]

\[ \text{typedecl} \rightarrow \text{var}_1, \ldots, \text{var}_n :: \text{type}, \quad n \geq 1 \]

\[ \text{var} \rightarrow \text{string starting with lower case symbol} \]

\[ \text{fundecl} \rightarrow \text{funlhs} \; \text{rhs} \]

\[ \text{funlhs} \rightarrow \text{var} \; \text{pat} \]

\[ \text{rhs} \rightarrow = \; \text{exp} \]
Evaluation in HASKELL

```
square 11

square (12 - 1)

11 * (12 - 1)  (12 - 1) * 11

11 * 11

121
```
Function Declarations with Pattern Matching

len :: [data] -> Int
len [] = 0
len (x : xs) = 1 + len xs

fac :: Int -> Int
fac 0 = 1
fac (x+1) = (x+1) * fac x

second :: [Int] -> Int
second [] = 0
second (x : []) = 0
second (x : y : xs) = y

half :: Int -> Int
half 0 = 0
half 1 = 0
half (x+2) = 1 + half x