HASKELL-declarations

$\underline{\mathsf{decl}} \quad \rightarrow \quad \underline{\mathsf{typedecl}} \mid \underline{\mathsf{fundecl}}$

$\underline{\mathsf{typedecl}} \quad \rightarrow \quad \underline{\mathsf{var}}_1, \ldots, \underline{\mathsf{var}}_n \; :: \; \underline{\mathsf{type}}, \qquad n \geq 1$

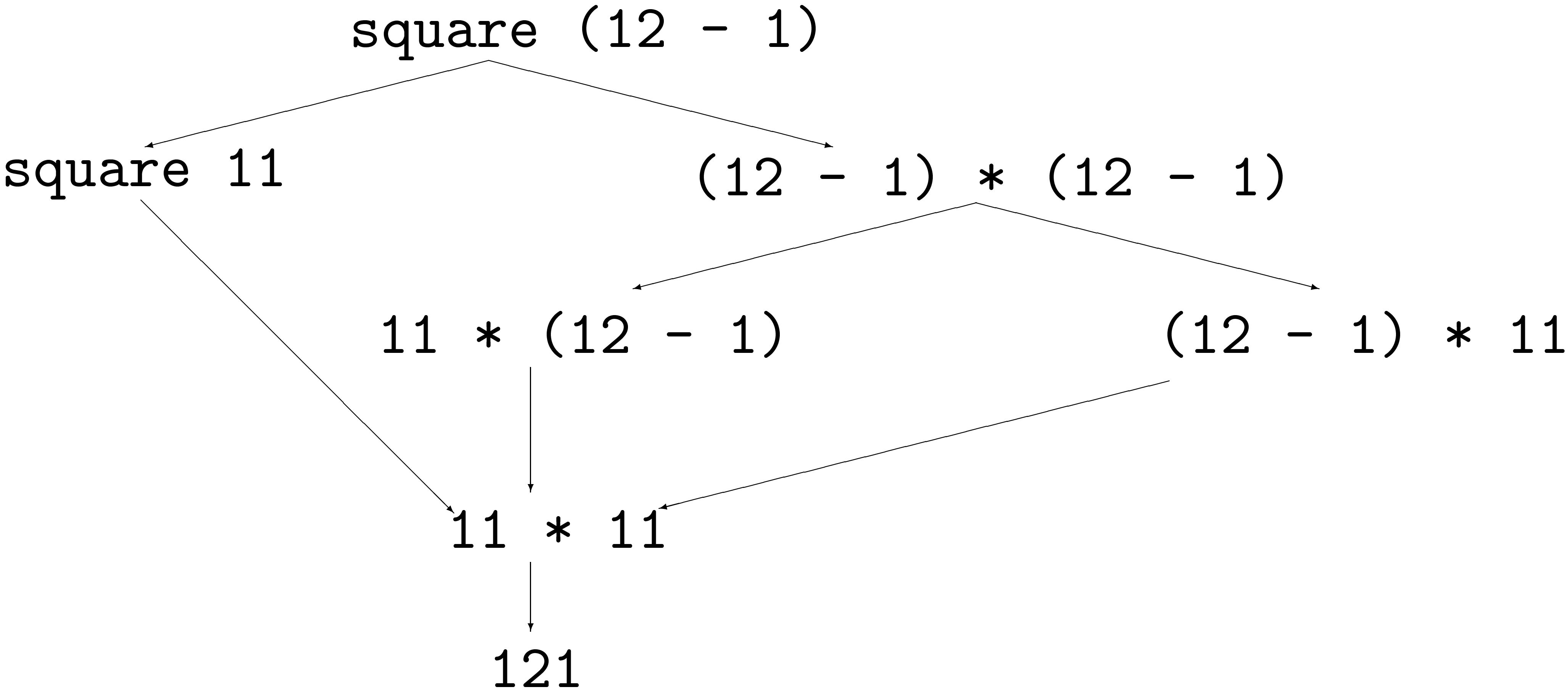
$\underline{\mathsf{var}} \quad \rightarrow \quad$ string starting with lower case symbol

$\underline{\mathsf{fundecl}} \quad \rightarrow \quad \underline{\mathsf{funlhs}} \; \underline{\mathsf{rhs}}$

$\underline{\mathsf{funlhs}} \quad \rightarrow \quad \underline{\mathsf{var}} \; \underline{\mathsf{pat}}$

$\underline{\mathsf{rhs}} \quad \rightarrow \quad = \; \underline{\mathsf{exp}}$

Evaluation in HASKELL



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
```