Today’s Agenda

• Type synonyms
• Type generality
• Equality types
• Syntactic sugar
Type Synonyms

• What does `int * int * int` represent?
• In HW1 we called it a date
• Wouldn’t it be nice to reflect this representation in the source code itself?

```c
type date = int * int * int
```
**type vs datatype**

- **datatype** introduces a new type name, distinct from all existing types

  ```
  datatype suit = Club | Diamond | Heart | Spade
  datatype rank = Jack | Queen | King | Ace
  | Num of int
  ```

- **type** is just another name

  ```
  type card = suit * rank
  ```
Type Synonyms

Why?
• For now, just for convenience
• It doesn’t let us do anything new

Later in the course we will see another use related to modularity.
Type Generality

Write a function that appends two string lists...
Type Generality

• We would expect

\[
\text{string list } * \text{ string list } \rightarrow \text{ string list}
\]

• But the type checker found

\[
\text{`a list } * \text{ `a list } \rightarrow \text{ `a list}
\]

• `a are called Polymorphic Types
• Why is this OK?
More General Types

• The type

\[
\texttt{\textquoteleft \texttt{a list} * \texttt{\textquoteleft a list} -> \texttt{\textquoteleft a list}}
\]

is more general than the type

\[
\texttt{\text{string list} * \text{string list list} -> \text{string list list}}
\]

and “can be used” as any less general type, such as

\[
\texttt{\text{int list} * \text{int list} -> \text{int list}}
\]

• But it is not more general than the type

\[
\texttt{\text{int list} * \text{string list list} -> \text{int list}}
\]
The Type Generality Rule

The “more general” rule

A type $t1$ is more general than the type $t2$ if you can take $t1$, replace its type variables \textbf{consistently}, and get $t2$

What does \textbf{consistently} mean?
Equality Types

Write a list “contains” function...
Equality Types

- The double quoted variable arises from use of the \( = \) operator
  - We can use \( = \) on most types like \texttt{int}, \texttt{bool}, \texttt{string}, tuples (that contain only “equality types”)
  - Functions and \texttt{real} are not ”equality types”

- Generality rules work the same, except substitution must be some type which can be compared with \( = \)

- You can ignore warnings about “calling polyEqual”
Syntactic Sugar

• If-then-else is implemented as syntactic sugar for a case statement
If-then-else

• We’ve just covered case statements
• How could we implement if-then-else?

```plaintext
case x of
  true => "apple"
| false => "banana"
```

```plaintext
if x then "apple" else "banana"
```
Adventures in pattern matching

• Shape example
• Function-pattern syntax if we get to it