Type Synonyms

What if I want to call `int * int * int` a `date`?

```
type date = int * int * int
```

Type Synonyms

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 Synonym

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

We expected
```
string list * string list -> string list
```

But the type checker says
```
'a list * 'a list -> 'a list
```

Why is this okay?
<table>
<thead>
<tr>
<th>Type Generality</th>
<th>Type Generality</th>
</tr>
</thead>
<tbody>
<tr>
<td>The type ‘a is more general</td>
<td>The “more general” rule</td>
</tr>
<tr>
<td>More general types “can be used” as any less general type.</td>
<td>A type $t_1$ is more general than the type $t_2$ if you can take $t_1$, replace its type variables consistently, and get $t_2$</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Equality Types</td>
<td>Equality Types</td>
</tr>
<tr>
<td>Write a contains function...</td>
<td>Double quotes arise from use of the ‘=’ operator</td>
</tr>
<tr>
<td></td>
<td>We can only use ‘=’ on types that can be compared</td>
</tr>
<tr>
<td></td>
<td>Generality rules work the same, except substitution must be some type which can be compared with ‘=’</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Syntactic Sugar</td>
<td>Syntactic Sugar</td>
</tr>
<tr>
<td>If-then-else is just a case statement in disguise...</td>
<td>Pattern matching...</td>
</tr>
</tbody>
</table>