CSE 341: Programming Languages

Winter 2005

Lecture 5— Type synonyms, more pattern-matching, accumulators

Goals

- Contrast type synonyms with new types
- See pattern-matching for built-in "one of" types (not really a concept, but important for ML programming) and "each of" types
- Investigate why accumulator-style recursion can be more efficient

Type synonyms

You can bind a *type name* to a type. Example:

```
type intpair = int * int
```

(We call something else a *type variable*.)

In ML, this creates a *synonym*, also known as a *transparent* type definition. Recursion not allowed.

So a type name is *equivalent* to its definition.

To contrast, the type a datatype binding introduces is not equivalent to any other type (until possibly a later type binding).