Lec 8: Data Structures

CS220: Programming Principles

Sang Kil Cha
Set
Mathematical Set

A set is a collection of *distinct* objects.
Standard Set Operations

- Union \( A \cup B \).
- Intersection \( A \cap B \).
- Difference \( A \setminus B \).
Data Abstraction (w/ List)

We can use List to implement Set!

```
type Set<'T> = List<'T>
val union : Set<'T> -> Set<'T> -> Set<'T>
val inter : Set<'T> -> Set<'T> -> Set<'T>
val minus : Set<'T> -> Set<'T> -> Set<'T>
```

The expression “type A = <type name>” is called a “Type Abbreviation”. You can give an alternative name for a type using this expression\(^1\).

\(^1\)https://docs.microsoft.com/en-us/dotnet/fsharp/language-reference/type-abbreviations
Exercise: Implementing Set with List

What’s the problem?
Built-in Set

Implementing a set data structure is beyond the scope of this course, but we will learn how to use built-in data structures.

- Set.ofList
- Set.empty
- Set.add
- Set.remove
- Set.union
- Set.intersect
- Set.difference
Map and Fold

Built-in Set stores its elements in an order, and the mapping and folding will happen in the order of the elements. Elements in Set should have a comparable types\(^2\).

- Set.map: \((	ext{'T} -> 	ext{'U}) \rightarrow 	ext{Set}'	ext{T}\) \rightarrow 	ext{Set}'	ext{U}\)
- Set.fold: \((	ext{'State} -> 	ext{'T} -> 	ext{'State}) \rightarrow 	ext{'State} \rightarrow 	ext{Set}'	ext{T}\) \rightarrow 	ext{'State}\)

\(^2\)This is so-called type constraints in F# jargon.
Map
Map?

A map is a data structure composed of a collection of (key, value) pairs, such that each possible key appears at most once in the collection.

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>k1</td>
<td>&quot;Beer&quot;</td>
</tr>
<tr>
<td>k2</td>
<td>&quot;Juice&quot;</td>
</tr>
<tr>
<td>k3</td>
<td>&quot;Soda&quot;</td>
</tr>
</tbody>
</table>
Standard Map Operations

- **Add**: add a key-value pair to the map.
- **Find**: find a value associated with a particular key.
Data Abstraction (w/ List)

We can use List to implement Map!

```plaintext
type Map<'K, 'V> = List<'K * 'V>
val add: 'K -> 'V -> Map<'K, 'V> -> Map<'K, 'V>
val find: 'K -> Map<'K, 'V> -> 'V
```
Built-in Map

- Map.ofList
- Map.empty
- Map.add
- Map.find
- Map.tryFind
- Map.containsKey
Map and Fold

Built-in Map stores its elements in an order, and the mapping and folding will happen in the order of the elements.

- Map.map: \((’K \to ’V \to ’U) \to \text{Map}(’K, ’V) \to \text{Map}(’K, ’U)\)
- Map.fold: \((’\text{State} \to ’K \to ’V \to ’\text{State}) \to ’\text{State} \to \text{Map}(’K, ’V) \to ’\text{State}\)
In-Class Activity #9

Implement a function that takes in two integer lists and returns a new list of common elements. For example, given [1; 2; 3] and [3; 4; 5], the function should return [3]. Hint: Set has a built-in function for computing an intersection of two given sets.
Conclusion
• Set and Map are popular functional data structures in F#.
• There are many built-in functions for both data structures.
• Many built-in functions are higher-order functions. Notably `map`, `iter`, and `fold` functions are defined for most data structures.
Question?