Programming Languages

Pattern Matching

WashU CSE 425s
Prof. Dennis Cosgrove
#03: Mon, Jan 31, 2022

Challenge Problems

The challenge problems are extra credit. I have a high opinion of WashU students and therefore expect you all to do them. They are some of the more interesting problems in the whole course and highly recommend them to you.

S&Q: Is this not object oriented programming? Why is functional programming considered entirely different? Or are functional languages a subset/superset of OOP languages?

No. It is not object oriented programming.

- SML supports encapsulation
- Missing critical elements: classes, inheritance, override, dynamic dispatch

S&Q: Is 'option' syntactic sugar for datatypes? It seems like the behavior could be replicated with something like: datatype option = SOME of (type) | NONE.

https://smlfamily.github.io/Basis/option.html

```
datatype 'a option = NONE | SOME of 'a
```

S&Q: Is list regarded as a data type?

https://smlfamily.github.io/Basis/list.html

```
datatype 'a list = nil | :: of 'a * 'a list
```

note: we can also use [] in patterns for nil

S&Q: I don't really understand the specifics of the 'Each of', 'One of', and 'Self reference' types that were mentioned. Could you please re-explain them?

each of:

```
val xyz = (1,2,3)
int * int * int
```

one of:

```
datatype suit = Clubs | Diamonds | Hearts | Spades
datatype rank = Jack | Queen | King | Ace | Num of int
```

self reference:

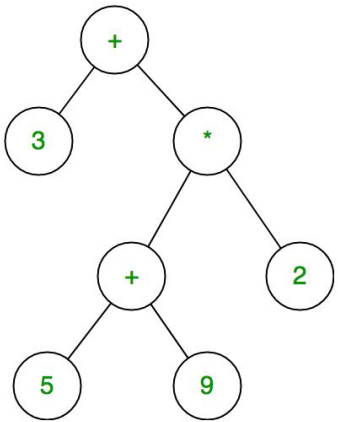
```
datatype 'a list = nil | :: of 'a * 'a list
```

S&Q: I don't quite understand how recursive datatype binding works. Could you give one more concrete example?

S&Q: Could we go over another example of datatype bindings that have self-references? I don't quite understand when Prof. Grossman describes them as trees

```
datatype exp = Constant of int
             | Negate of exp
             | Add of exp * exp
             | Multiply of exp * exp
```

```
(3+((5+9)*2))
```



S&Q: Could we go over another example of datatype bindings that have self-references? I don't quite understand when Prof. Grossman describes them as trees

```
datatype 'a tree = leaf
                 | node of { value : 'a, left : 'a tree, right : 'a tree }
```

S&Q: Is pattern matching in SML tied to using a case expression or is it an idea that can be used in other contexts? (Say, a function accepts only a Multiply expression as a parameter)

- function signatures are just patterns

S&Q: function signatures are just patterns

First, SML functions only take one argument

But, wait… UW 0

```
fun f(x,y) =
    x * y
```

https://www.coursera.org/learn/programming-languages/lecture/FCVA9/each-of-pattern-matching-truth-about-functions

S&Q: function signatures are just patterns

```
fun my_length(values) =
   case values of
     [] => 0
   | _ :: values' => 1 + my_length(values')
```

```
fun my_length [] = 0
  | my_length (_::values') = 1 + my_length(values')
```

S&Q: Still confused on records vs. tuples

Tuples Are Just Records (coming soon)

https://www.coursera.org/learn/programming-languages/lecture/e2qVG/tuples-as-syntactic-sugar

S&Q: In some of the example functions, he omitted the type of the parameter. Is this allowed? How does the type checker determine the type?

Hindley–Milner type system

SML is really strict, but very helpful.

SML Knows What The Types Are At Compile Time Can Reject The Program Before You Even Run It

```
val x = 2
val y = x DIV "fred"
```

```
x = 2
y = x / "fred"
```

S&Q: What would the type of a case expression be?

every single branch must return the same type of expression,
so it is that common type

(akin to the `then` branch and the `else` branch of an `if` expression must return expressions of the same type)

S&Q: Why would a missing case not throw an exception?
Is there a 'default' or catchall option like in other languages' switch statements?

every expression must evaluate to a value

not like statements

You can be missing a case in a switch in Java. Doing nothing is a fine path.

S&Q: Are datatypes allowed to have multiple constructors with the same name as long as they accept different types?

just need different names

S&Q: Is it actually useful to represent an expression tree when natural math operations already exist in SML? What are some more unique examples of expression trees?

when you are building another language (which programming languages researchers like Prof. Grossman tend to do)

S&Q: Is pattern matching used in the compilation of other languages?

MLs descendants (for example, Haskell) and Rust

S&Q: Pattern matching seems difficult but I am interested in it. I understand the videos but what will we be using this for?

Pattern matching is soooooooooooooooooooooooooo sweet.

super charged if elseif elseif elseif with automatic variable extraction

Q&S: Are case expressions from SML "borrowed" in other languages (like C++ and Java) where case statements are used, rather than several if-else statments?

No. Case statements in C++, Java are totally different. More like what can be implemented as a single jump instruction in C, as opposed to pattern matching.

Rust adopted pattern matching, though.

S&Q: What does it mean by naming variables introduced in a pattern?

while you are declaring the pattern you tell the system what names you would would like to use for the variables which will expand the environment

S&Q: So far, all of the pieces of SML we've seen have had parallels in Java, but pattern matching seems to break this rule. Is this true?

Pattern matching the big difference.

S&Q: Dan said case statements will become more complex, but he said that the matching is patterns and that expressions are not evaluated, so are SML case expressions ever written similarly to Java case statements? I think I just need to wait and see more case patterns demonstrated.

Sum Scan

https://classes.engineering.wustl.edu/cse425s/index.php?title=Sum_Scan_Assignment

Is Strictly Ascending

https://classes.engineering.wustl.edu/cse425s/index.php?title=Is_Strictly_Ascending_Assignment

Sum Scan With Pattern Matching

https://classes.engineering.wustl.edu/cse425s/index.php?title=Sum_Scan_With_Pattern_Matching_Assignment

Practice Problems B

https://classes.engineering.wustl.edu/cse425s/index.php?title=SML_Practice_Problems_B

credit: Pavel Lepin and Charilaos Skiadas

note: tons of tests

Demo How To Run The Tests

Plan

loop

- answer questions until there is a pause
- implement Practice Problems without and then with pattern matching

S&Q: What are some other examples of syntactic sugar in ML or Java?

S&Q: Could we have a case expression with recursion that doesn't make up the entirety of the function body?

S&Q: Are there similar examples of syntactic sugar in languages more familiar to us, like Java C, C++, Python?

S&Q: What is the minimum set of features required in a programming language? Is there some equivalent to being Turing complete? Are there any languages that are even more minimal than ML that don't construct any of the data types besides the essentials?

S&Q: What happens if you use the same field name twice in a record? Does it throw an error or does the first value get replaced?

S&Q: At the risk of getting out ahead of our skis, how does Java use OOP to do the 'opposite' of what SML is doing in creating each of and one of types? I'm trying to learn the SML but still take a step back and understand the language constructs and how this concept applies in different ways. It initially strikes me as pretty specific to SML, though perhaps this is due to unfamiliarity with functional programming?

S&Q: Are there similar examples of syntactic sugar in languages more familiar to us, like Java C, C++, Python?

S&Q: I was a bit confused on the question in useful datatypes "In this lecture, we introduced a simple arithmetic expression language. When we evaluate any expression in that language, every subexpression contained in that expression is always evaluated. Which of the following could we extend our language with, such that every subexpression is not necessarily evaluated?" The answer to this was if/then/else conditionals but I don't really understand how this answer is achieved.

S&Q: How similar is datatype to Class or enum in java, and what are the key differences?

S&Q: In the last video, Dan had the same evaluation for the Add case and the Multiply case. Is there any way to combine these? Perhaps "default" as in java?

S&Q: Since SML seems particularly convenient for compiler development, why do you think it is overshadowed by C and C++?

S&Q: Over the course so far we've seen just how powerful SML can be in terms of correctness, safety, and now parsing. What do you think is preventing it or another functional language from gaining mainstream appeal? Functional features are being added bit by bit to mainstream languages, however the experience really isn't the same.