January 9th 2016 - Discussion Notes

Ryan Sharif

January 8, 2016

Contents

1 Introduction

1.1 Contact information

Seunghyun Yoo shyoo1st@cs.ucla.edu

Office hours will be held on Mondays and Wednesdays from 5:30 - 6:30 pm @ BH2432

1.2 Homework

The first homework for this class is due January 14th.

2 OCaml Discussion

2.1 Basics

2.1.1 add function

let add x y = x + y ;;

2.1.2 factiorial function

let rec fact n = if n <= 0 then 1 else n * fact (n - 1) ;;

2.1.3 operands

There are separate operands for ints and reals:

int	float
+	+.
-	-.
*	*.

2.1.4 square function

```
let square x = x * x ;;
(* another way to define the same function*)
let square = fun x -> x * x ;;

(* one more way to define the same function  *)
let square x =
  match x with
  | x -> x * x ;;
```

2.1.5 how many parameters does a function take

Every function just takes one parameter by default; however, you can have a function take a tuple:

```
(* tuple way *)
let add(x,y) = x + y ;;

(* currying style *)
let add x y = x + y ;;
val add int -> int -> int = <fun>
```

2.1.6 conditions

```
let max a b =
  if a > b then a
  else b
;;

let eval op v1 v2 =
  match op with
| "+" -> v1 + v2
| "-" -> v1 -v2
| _ -> failwith "undefined"
```

2.1.7 lists

Lists have several properties:

- immutable
- homogeneous

We may want to add items to the front of a list: use the :: operator
We may also want to append items to the list: use the @ operator

2.1.8 type definition

To define a type:

```
type name = typedef;;
type ' name = typedef -polymorphic;;
```

For example, we can provide a binary-tree example:

```
type binary-tree = Leaf of int
  | Tree
  | Tree of binary tree * binary-tree;;
```

2.2 Homework