Bill Wadge

University of Victoria
Victoria BC Canada
Professor Emeritus

Seven Candles!

С днем рождения Виктор Львович

Wadge Degrees

Wadge Degrees

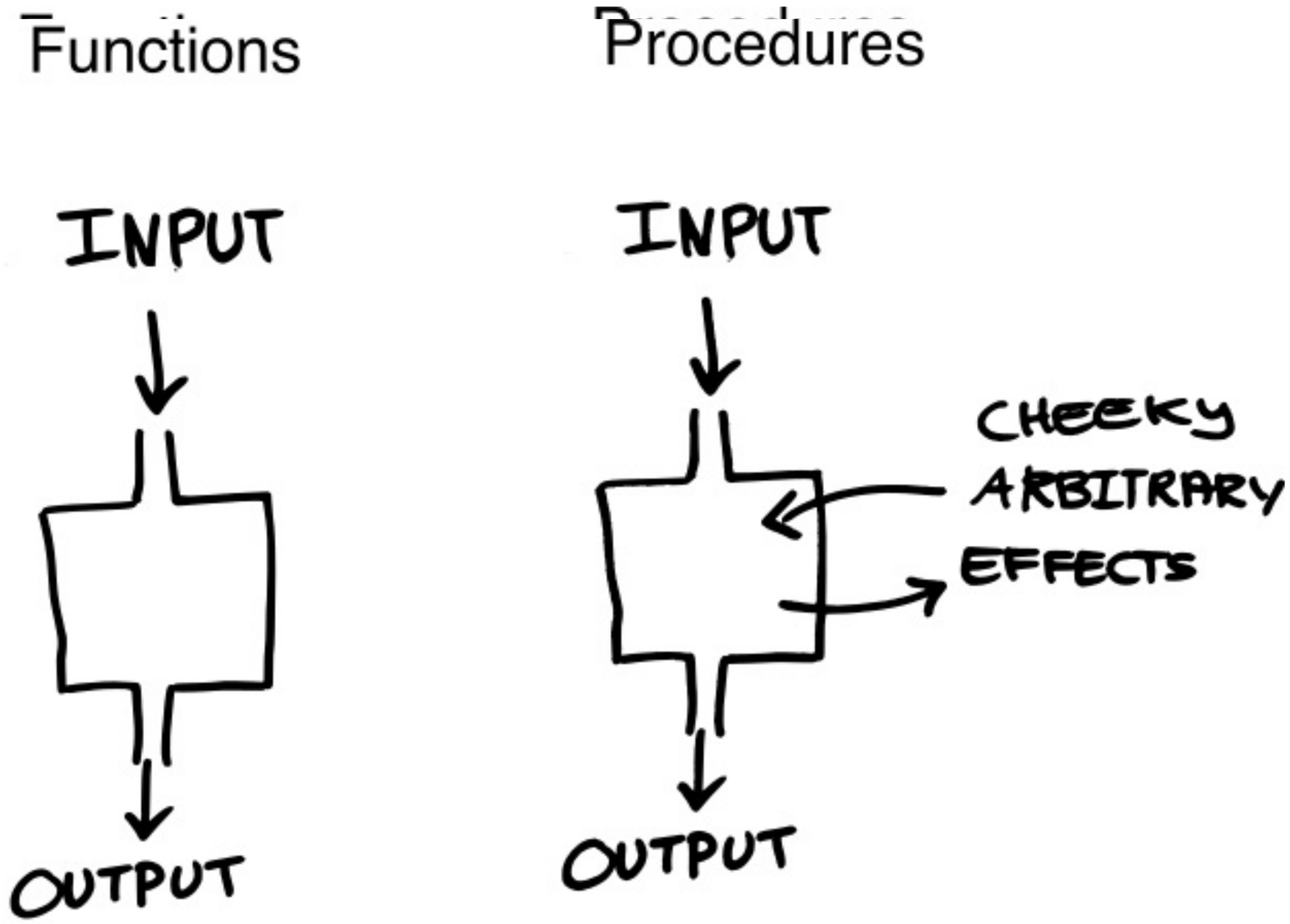
nothing

Putting the Fun Back in Functional Programming

PyFL is Fun!

Bill Wadge

Functional vs Procedural Programming



Ideas from Functional Programming that improve software design:

AVOID HIDDEN SIDE-EFFECTS

ENSURE REFERENTIAL TRANSPARENCY

USE LIST-PROCESSING FUNCTIONS

Kleene: Recursion Equations

f(n) = if n<2 then 1 else n*f(n-1) fi

Church: the λ Calculus

λf: λx: f(f(x))

Peter Landin: where clause

```
f(7)
        where
            f(n) = if
                       n<2 then 1
                       else f(n-1)+f(n-2)
                   fi
         end
```

David Turner: SASL, KRC, Miranda

```
subsets [] = [[]]
subsets(x:xs) =
    [[x] ++ y | y <- ys] ++ ys
            where ys = subsets xs
```

Simon Peyton-Jones, Phil Wadler et al.: Haskell

```haskell
quickSort []     = []
quickSort (x:xs) = quickSort [a | a <- xs, a < x]
                   ++ [x] ++
                   quickSort [a | a <- xs, a >= x]
```


```
eu([3,4])
  where
    eu = lambda(l) sqrt(ssq(l));
    ssq = lambda(l)
                  if  l ==[] then 0 else hd(l)**2+ssq(tl(l))  fi;
  end
```

Basic PyFL

- infix data expressions

  a*hd(b)+x

- conventional function application

  f(x+y,j<>k)

- where clauses

  p+q where p = 3; q = m*5; end

- lambda expressions

  lambda (x,y) x**2+y**2 end

Extension: ISWIM Function Definitions

```
ssq(x,y) = x**2 +y**2;
```

becomes

```
ssq = lambda (x,y) x**2 +y**2 end;
```

Non-Recursive Factorial

```
fac(7)
 where
  fac = Y(A);
  A(f) = lambda (n)
              if n<2 then 1 else n*f(n-1) fi;
  Y(a) = g(g) where g(x) = a(x(x)); end;
end
```

Extension: Multiple Returns

```
a, b = E;
```

becomes

```
a = cp0(E);
b = cp1(E);
```

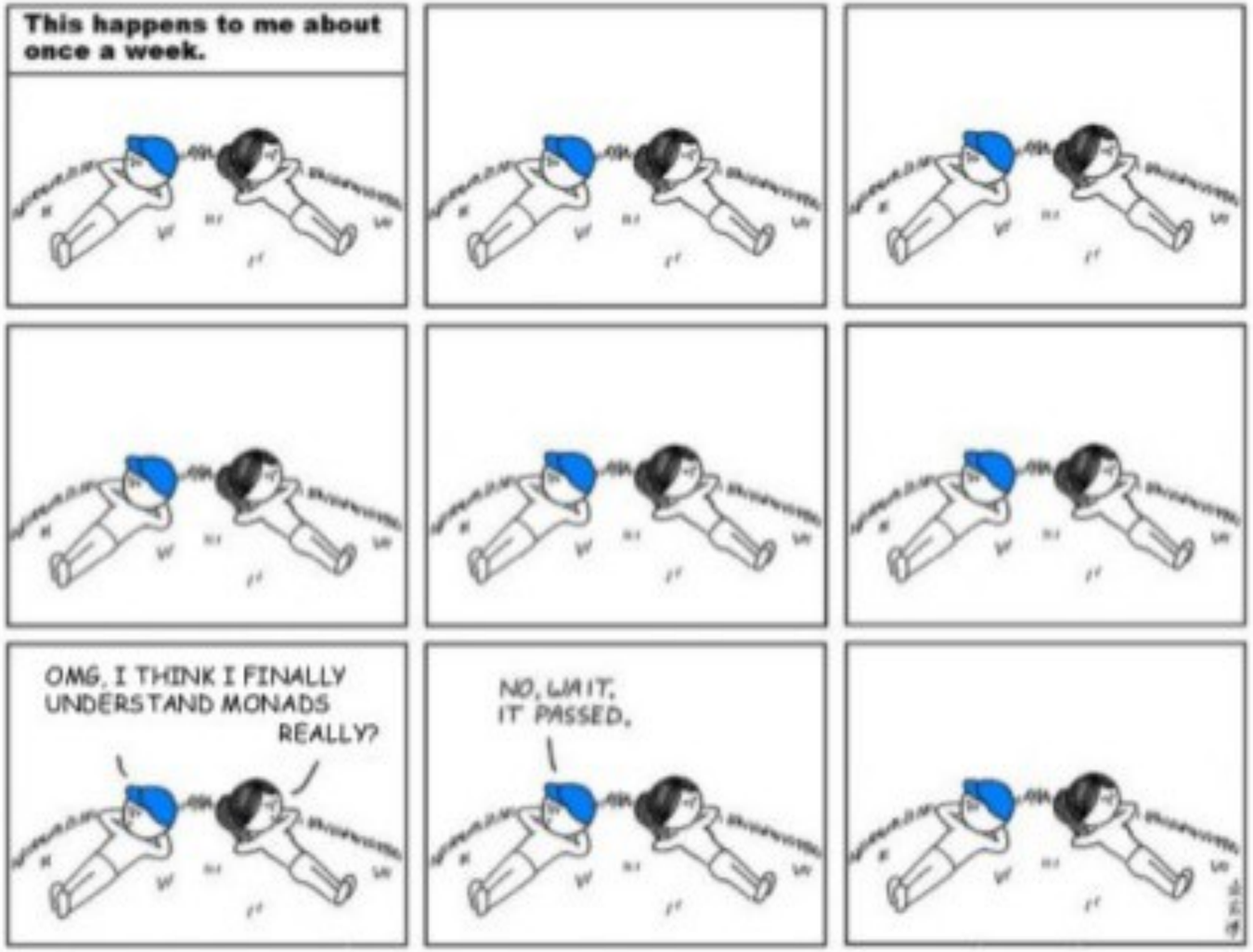
Extension: Input/Output

input(“N: ”)

output(“Root N ”,R)

No Monads!

What is a monad?



Extension: While Loops

```
Root10 = while abs( 10-A*A) < 0.0001;
                A = 1 fby (A+10/A)/2;
                result = A;
               end;
```

Extension: Variable Binding Operators

ssq(l) = sumfor i in l all i*i end

$$\sum_{i \in l} i*i$$

Extensions: Other VBOs

prodfor
maxfor
minfor
consfor
appendfor
concatfor

collectfor
unionfor
intersectfor
those
exist
forall

Example: Primes

```
primes(100)
  where
    primes(N)=
      those p in rg(2,N):
        not exist d in rg(2,sqrt(N)):
          d != p and p mod d == 0
        end
      end;
    rg(a,b) = if a>b then [] else a :: rg(a+1,b) fi
  end
```

Example: Subsets

```
allsubsets({a b c d})
where
allsubsets(s) =
  unionfor x in s all {%s%} + wix + sr + {{}}
    where
      sr = allsubsets(s - {% x %});
      wix = collectfor u in sr all u + {%x%} end;
   end
  end;
end
```

Example: Tictactoe

```
while outputs(DisplayBoard) &&
   not(HasWon(X) or HasWon(O) or IsTie) and
      ix   != 10
   board = startboard fby MakeMove(player,move);
   O = "O";
   X = "X";
   ix = 1 fby ix+1;
   player = "X" fby opponent(player);
```

Example: Tictactoe (Continued)

```
HasWon(p) =    // p has already won in board b
                exist l in lines:
                 forall c in l:
                  board(c) == p
                 end
                end;
```


DEMO