Programmer som Data - Assignment 4

Bastjan Rosgaard Sejberg, Søren Kastrup, Weihao Chen Nyholm-Andersen

September 2023

4.1

fsharp — fsharpi -r /Users/bastjansejberg/fsharp/FsLexYacc.Runtime.dll...

```
  val run : e:Absyn.expr -> int
  val ex1 : Absyn.expr
  val ex2 : Absyn.expr
  val ex3 : Absyn.expr
  val rundeep : n:int -> int
  val ex4 : Absyn.expr
  val ex5 : Absyn.expr

namespace FSI_0002
  val fromString : (string -> Absyn.expr)
  val eval : (Absyn.expr -> Fun.value Fun.env -> int)
  val run : e:Absyn.expr -> int

[> open ParseAndRun;;                                                    ]
[> run (fromString "5+7");;                                              ]
val it : int = 12

[> run (fromString "let y = 7 in y + 2 end");;                           ]
val it : int = 9

[> run (fromString "let f x = x + 7 in f 2 end");;                       ]
val it : int = 9

>
```

4.2

```
> let ex6 = fromString
-             @"let sum n = if n = 0 then n else n+sum(n-1)
-               in sum 1000 end";;
val ex6: expr =
  Letfun
    ("sum", "n",
     If
       (Prim ("=", Var "n", CstI 0), Var "n",
        Prim ("+", Var "n", Call (Var "sum", Prim ("-", Var "n", CstI 1)))),
     Call (Var "sum", CstI 1000))

> run ex6;;
val it: int = 500500
```

```
> let ex7 = fromString
-             @"let raised y = if y = 0 then 1 else 3*raised(y-1)
-               in raised 8 end";;
val ex7: expr =
  Letfun
    ("raised", "y",
     If
       (Prim ("=", Var "y", CstI 0), CstI 1,
        Prim ("*", CstI 3, Call (Var "raised", Prim ("-", Var "y", CstI 1)))),
     Call (Var "raised", CstI 8))

> run ex7;;
val it: int = 6561
```

```
- let ex8 = fromString
-             @"let raised y = if y = 0 then 1 else 3*raised(y-1)
-               in let addRaised z = if z = 0 then 1 else (raised z) + addRaised(z-1)
-                   in addRaised 11
-                   end
-               end";;
val ex8: expr =
  Letfun
    ("raised", "y",
     If
       (Prim ("=", Var "y", CstI 0), CstI 1,
        Prim ("*", CstI 3, Call (Var "raised", Prim ("-", Var "y", CstI 1)))),
     Letfun
       ("addRaised", "z",
        If
          (Prim ("=", Var "z", CstI 0), CstI 1,
           Prim
             ("+", Call (Var "raised", Var "z"),
              Call (Var "addRaised", Prim ("-", Var "z", CstI 1)))),
        Call (Var "addRaised", CstI 11)))

> run ex8;;
val it: int = 265720
```

```
> let ex9 = fromString
-             @"let raised y = y*y*y*y*y*y*y*y
-               in let addRaised z = if z = 0 then 0 else (raised z) + addRaised(z-1)
-                   in addRaised 10
-                   end
-               end";;
val ex9: expr =
  Letfun
    ("raised", "y",
     Prim
       ("*",
        Prim
          ("*",
           Prim
             ("*",
              Prim
                ("*",
                 Prim
                   ("*", Prim ("*", Prim ("*", Var "y", Var "y"), Var "y"),
                    Var "y"), Var "y"), Var "y"), Var "y"), Var "y"),
     Letfun
       ("addRaised", "z",
        If
          (Prim ("=", Var "z", CstI 0), CstI 0,
           Prim
             ("+", Call (Var "raised", Var "z"),
              Call (Var "addRaised", Prim ("-", Var "z", CstI 1)))),
        Call (Var "addRaised", CstI 10)))

> run ex9;;
val it: int = 167731333
```

Refer to the file ***Parse.fs*** *in the folder* ***Exercise_4.2-4.5****.*

4.3 & 4.4

```
> let pow = fromString @"let pow x n = if n=0 then 1 else x * pow x (n-1) in pow 3 8 end";;
val pow: Absyn.expr =
  Letfun
    ("pow", ["x"; "n"],
     If
       (Prim ("=", Var "n", CstI 0), CstI 1,
        Prim
          ("*", Var "x",
           Call (Var "pow", [Var "x"; Prim ("-", Var "n", CstI 1)]))),
     Call (Var "pow", [CstI 3; CstI 8]))

> let max2 = fromString @"let max2 a b = if a<b then b else a
- in let max3 a b c = max2 a (max2 b c)
- in max3 25 6 62 end
- end";;
val max2: Absyn.expr =
  Letfun
    ("max2", ["a"; "b"], If (Prim ("<", Var "a", Var "b"), Var "b", Var "a"),
     Letfun
       ("max3", ["a"; "b"; "c"],
        Call (Var "max2", [Var "a"; Call (Var "max2", [Var "b"; Var "c"])]),
        Call (Var "max3", [CstI 25; CstI 6; CstI 62])))

> run pow;;
val it: int = 6561

> run max2;;
val it: int = 62
```

For exercise 4.3 refer to the files ***Absyn.fs*** *and* ***Fun.fs*** *in the folder* ***Exercise_4.2-4.5****.*
For exercise 4.4 refer to the files ***FunLex.fsl*** *and* ***FunPar.fsy*** *in the folder* ***Exercise_4.2-4.5****.*

4.5

```
> let ex12 = fromString @"let inrange x y z = (x < y) && (y < z) in inrange 2 5 7 end";;
val ex12: Absyn.expr =
  Letfun
    ("inrange", ["x"; "y"; "z"],
     If
       (Prim ("<", Var "x", Var "y"), Prim ("<", Var "y", Var "z"), CstB false),
     Call (Var "inrange", [CstI 2; CstI 5; CstI 7]))

> let ex13 = fromString @"let outrange x y z = (y < x) || (z < y) in outrange 2 5 7 end";;
val ex13: Absyn.expr =
  Letfun
    ("outrange", ["x"; "y"; "z"],
     If
       (Prim ("<", Var "y", Var "x"), CstB true, Prim ("<", Var "z", Var "y")),
     Call (Var "outrange", [CstI 2; CstI 5; CstI 7]))

> run ex12;;
val it: int = 1

> run ex13;;
val it: int = 0
```

Refer to the files ***FunLex.fsl*** *and* ***FunPar.fsy*** *in the folder* ***Exercise_4.2-4.5****.*