PONTIFICIA UNIVERSIDAD CATOLICA MADRE Y MAESTRA

Nombre:
Félix Alejandro Guzmán 2014 - 0565

Materia:
ST-ISC-314-T-001 Programación 3

Profesor:
Juan R. Núñez P.

Practica sobre:

Sintaxis Abstracta

Fecha de Entrega:

sábado, 24 de marzo de 2018

Sources.cm

```
Group is

absyn.sml
prabsyn.sml
errormsg.sml
table.sig
table.sml
symbol.sml
parse.sml
tiger.lex
tiger.grm
$/smlnj-lib.cm
$/ml-yacc-lib.cm
$/basis.cm
```

Test.tig

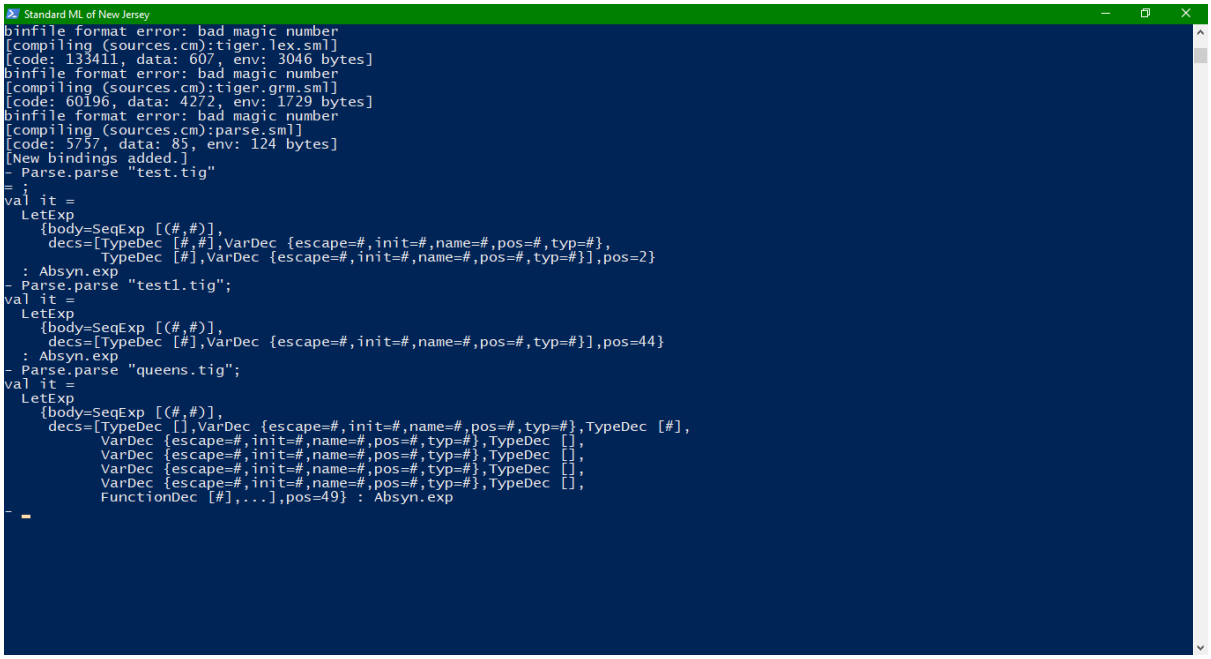
```
let
    type a = int
    type b = string
    var x := 5
    type c = a
    var y := 6
in
    row [ 5 ]
end
```

Test1.tig

```
/* an array type and an array variable */
let
    type  arrtype = array of int
    var arr1:arrtype := arrtype [10] of 0
in
    arr1
end
```

Queens.tig

```
/* A program to solve the 8-queens problem */

let
    var N := 8

    type intArray = array of int

    var row := intArray [ N ] of 0
    var col := intArray [ N ] of 0
    var diag1 := intArray [N+N-1] of 0
    var diag2 := intArray [N+N-1] of 0

    function printboard() =
       (for i := 0 to N-1
     do (for j := 0 to N-1
          do print(if col[i]=j then " O" else " .");
         print("\n"));
         print("\n"))

    function try(c:int) =
( /*  for i:= 0 to c do print("."); print("\n"); flush();*/
     if c=N
     then printboard()
     else for r := 0 to N-1
       do if row[r]=0 & diag1[r+c]=0 & diag2[r+7-c]=0
               then (row[r]:=1; diag1[r+c]:=1; diag2[r+7-c]:=1;
                 col[c]:=r;
                     try(c+1);
            row[r]:=0; diag1[r+c]:=0; diag2[r+7-c]:=0)

)
 in try(0)
end
```


Standard ML of New Jersey
binfile format error: bad magic number
[compiling (sources.cm):tiger.lex.sml]
[code: 133411, data: 607, env: 3046 bytes]
binfile format error: bad magic number
[compiling (sources.cm):tiger.grm.sml]
[code: 60196, data: 4272, env: 1729 bytes]
binfile format error: bad magic number
[compiling (sources.cm):parse.sml]
[code: 5757, data: 85, env: 124 bytes]
[New bindings added.]
- Parse.parse "test.tig"
= ;
val it =
LetExp
{body=SeqExp [(#,#)],
decs=[TypeDec [#,#],VarDec {escape=#,init=#,name=#,pos=#,typ=#},
TypeDec [#],VarDec {escape=#,init=#,name=#,pos=#,typ=#}],pos=2}
: Absyn.exp
- Parse.parse "test1.tig";
val it =
LetExp
{body=SeqExp [(#,#)],
decs=[TypeDec [#],VarDec {escape=#,init=#,name=#,pos=#,typ=#}],pos=44}
: Absyn.exp
- Parse.parse "queens.tig";
val it =
LetExp
{body=SeqExp [(#,#)],
decs=[TypeDec [],VarDec {escape=#,init=#,name=#,pos=#,typ=#},TypeDec [#],
VarDec {escape=#,init=#,name=#,pos=#,typ=#},TypeDec [],
VarDec {escape=#,init=#,name=#,pos=#,typ=#},TypeDec [],
VarDec {escape=#,init=#,name=#,pos=#,typ=#},TypeDec [],
VarDec {escape=#,init=#,name=#,pos=#,typ=#},TypeDec [],
FunctionDec [#],...],pos=49} : Absyn.exp
-