

Prog ::= let Bind in Exp end  
 FIRST(let Bind in Exp end) = {let}  
 Aggiungo nella cella [Prog,let] : Prog ::= let Bind in Exp end  
 Prog ::= letrec Bind in Exp end  
 FIRST(letrec Bind in Exp end) = {letrec}  
 Aggiungo nella cella [Prog,letrec] : Prog ::= letrec Bind in Exp end

Bind ::= var = Exp X  
 FIRST(var = Exp X) = {var}  
 Aggiungo nella cella [Bind,var] : Bind ::= var = Exp X

X ::= and Bind  
 FIRST(and Bind) = {and}  
 Aggiungo nella cella [X,and] : X ::= and Bind  
 X ::= epsilon  
 FIRST(epsilon) = FOLLOW(X) = {in}  
 Aggiungo nella cella [X,in] : X ::= epsilon

Exp ::= Prog  
 FIRST(Prog) = {let,letrec}  
 Aggiungo nella cella [Exp,let] e [Exp,letrec] : Exp ::= Prog  
 Exp ::= var Y  
 FIRST(var Y) = {var}  
 Aggiungo nella cella [Exp,var] : Exp ::= var Y  
 Exp ::= const  
 FIRST(const) = {const}  
 Aggiungo nella cella [Exp,const] : Exp ::= const  
 Exp ::= List  
 FIRST(List) = {[,nil]}  
 Aggiungo nella cella [Exp,[]] e [Exp,nil] : Exp ::= List  
 Exp ::= op ( Seq\_Exp )  
 FIRST(op ( Seq\_Exp )) = {op}  
 Aggiungo nella cella [Exp,op] : Exp ::= op ( Seq\_Exp )  
 Exp ::= lambda (Seq\_Var ) Exp  
 FIRST(lambda (Seq\_Var ) Exp) = {lambda}  
 Aggiungo nella cella [Exp,] : Exp ::= lambda (Seq\_Var ) Exp

Y ::= ( Seq\_Exp )  
 FIRST(( Seq\_Exp )) = {(}  
 Aggiungo nella cella [Y,(] : Y ::= ( Seq\_Exp )  
 Y ::= epsilon  
 FIRST(epsilon) = FOLLOW(Y) = {and, end, ,, in, let, letrec, var, const, op, lambda, [, nil]}  
 Aggiungo nella cella [Y,and] : Y ::= epsilon  
 Aggiungo nella cella [Y,end] : Y ::= epsilon  
 Aggiungo nella cella [Y,)] : Y ::= epsilon  
 Aggiungo nella cella [Y,in] : Y ::= epsilon  
 Aggiungo nella cella [Y,let] : Y ::= epsilon  
 Aggiungo nella cella [Y,letrec] : Y ::= epsilon  
 Aggiungo nella cella [Y,var] : Y ::= epsilon  
 Aggiungo nella cella [Y,const] : Y ::= epsilon  
 Aggiungo nella cella [Y,op] : Y ::= epsilon  
 Aggiungo nella cella [Y,labda] : Y ::= epsilon  
 Aggiungo nella cella [Y,[ ] : Y ::= epsilon  
 Aggiungo nella cella [Y,nil] : Y ::= epsilon

Seq\_Exp ::= Exp Z  
 FIRST(Exp Z) = FIRST(Exp) = {let, letrec, var, const, op, lambda, [, nil}  
 Aggiungo nelle celle [Seq\_Exp, let] : Seq\_Exp ::= Exp Z  
 Aggiungo nelle celle [Seq\_Exp, letrec] : Seq\_Exp ::= Exp Z  
 Aggiungo nelle celle [Seq\_Exp, var] : Seq\_Exp ::= Exp Z  
 Aggiungo nelle celle [Seq\_Exp, const] : Seq\_Exp ::= Exp Z  
 Aggiungo nelle celle [Seq\_Exp, op] : Seq\_Exp ::= Exp Z  
 Aggiungo nelle celle [Seq\_Exp, lambda] : Seq\_Exp ::= Exp Z  
 Aggiungo nelle celle [Seq\_Exp, []] : Seq\_Exp ::= Exp Z  
 Aggiungo nelle celle [Seq\_Exp, nil] : Seq\_Exp ::= Exp Z

Z ::= Seq\_Exp  
 FIRST(Seq\_Exp) = First(Exp) = {let, letrec, var, cst, op, lambda, [, nil}  
 Aggiungo nelle celle [Z, let] : Z ::= Seq\_Exp  
 Aggiungo nelle celle [Z, letrec] : Z ::= Seq\_Exp  
 Aggiungo nelle celle [Z, var] : Z ::= Seq\_Exp  
 Aggiungo nelle celle [Z, const] : Z ::= Seq\_Exp  
 Aggiungo nelle celle [Z, op] : Z ::= Seq\_Exp  
 Aggiungo nelle celle [Z, lambda] : Z ::= Seq\_Exp  
 Aggiungo nelle celle [Z, []] : Z ::= Seq\_Exp  
 Aggiungo nelle celle [Z, nil] : Z ::= Seq\_Exp  
 Z ::= epsilon  
 FIRST(epsilon) = FOLLOW(Z) = { }  
 Aggiungo nelle celle [Z, )] : Z ::= epsilon

Seq\_Var ::= var W  
 FIRST(var W) = { var }  
 Aggiungo nelle celle [Seq\_Var, var] : Seq\_Var ::= var W

W ::= Seq\_Var  
 FIRST(Seq\_Var) = { var }  
 Aggiungo nella cella [W, var] : W ::= Seq\_Var  
 W ::= epsilon  
 FIRST(epsilon) = FOLLOW(W) = { }  
 Aggiungo nella cella [W, )] : W ::= epsilon

List ::= [ Const\_List ]  
 FIRST([ Const\_List ]) = { [ }  
 Aggiungo nella cella [List, []] : List ::= [ Const\_List ]  
 List ::= nil  
 FIRST(nil) = { nil }  
 Aggiungo nella cella [List, nil] : List ::= nil

Const\_List ::= const V  
 FIRST(const V) = { const }  
 Aggiungo nella cella [Const\_List, const] : Const\_List ::= const V  
 Const\_List ::= List V  
 FIRST(List V) = FIRST(List) = { [, nil }  
 Aggiungo nella cella [Const\_List, []] : Const\_List ::= List V  
 Aggiungo nella cella [Const\_List, nil] : Const\_List ::= List V

V ::= ::Const\_list  
 FIRST(::Const\_list) = { :: }  
 Aggiungo nella cella [V, ::] : V ::= ::Const\_list  
 V ::= epsilon  
 FIRST(epsilon) = FOLLOW(V) = { ] }  
 Aggiungo nella cella [V, ] ] : V ::= epsilon