Syed Taseer Haider
22L-6696
BSCS 6G

Stmt → WhileStmt
| IfStmt
| CompStmt
| Expr

WhileStmt → while (Rvalue) Stmt
{ S.begin = newlabel(); }
{ Rvalue.true = newlabel(); Rvalue.false = newlabel(); }
{ gen('label', S.begin) }
{ gen('if', Rvalue.addr, 'goto', Rvalue.true) }
{ gen('goto', Rvalue.false) }
{ gen('label', Rvalue.true) }
{ Stmt.next = S.begin; Stmt }
{ gen('goto', S.begin) }
{ gen('label', Rvalue.false) }

IfStmt → Agar (Rvalue) Stmt ElsePart
{ Rvalue.true = newlabel(); Rvalue.false = newlabel(); S.next =

{ gen('goto', Rvalue.false) }

{ gen('label', Rvalue.true) }

{ Stmt.next = S.next; Stmt } { gen('goto', S.next) }

{ gen('label', Rvalue.false) }

{ ElsePart }

{ gen('label', S.next) }

ElsePart → Wagarna Stmt

{ Stmt }

ElsePart → ε

CompStmt → { StmtList }

StmtList → Stmt StmtList'

StmtList' → Stmt StmtList' | ε

Expr → identifier := Mag

{ gen(identifier.lexeme, ':=', Mag.addr) }

Expr → ε

Rvalue → Mag Rvalue'

{ Rvalue'.inh = Mag.addr; Rvalue.addr = Rvalue'.addr }

Rvalue' → RelOp Mag Rvalue'

{ t = newtemp(); gen(t, ':=', Rvalue'.inh, RelOp.op, Mag.addr);

Rvalue'1.inh = t; Rvalue'.addr = Rvalue'1.addr }

Rvalue' → ε

{ Rvalue'.addr = Rvalue'.inh }

RelOp → ==

{ RelOp.op = '==' }

RelOp → < RelOp1

RelOp1 → =

{ RelOp.op = '<=' }

RelOp1 → >

{ RelOp.op = '<>' }

RelOp1 → ε

{ RelOp.op = '<' }

RelOp → > RelOp2

RelOp2 → =

{ RelOp.op = '>=' }

RelOp2 → ε

{ RelOp.op = '>' }

Mag → Term Mag'

{ Mag'.inh = Term.addr; Mag.addr = Mag'.addr }

Mag' → + Term Mag'

{ t = newtemp(); gen(t, ':=', Mag'.inh, '+', Term.addr); Mag'1.inh = t;

Mag'.addr = Mag'1.addr }

Mag' → - Term Mag'

{ t = newtemp(); gen(t, ':=', Mag'.inh, '-', Term.addr); Mag'1.inh = t;

Mag'.addr = Mag'1.addr }

Mag' → ε

{ Mag'.addr = Mag'.inh }

Term → Factor Term'

{ Term'.inh = Factor.addr; Term.addr = Term'.addr }

Term' → * Factor Term'

{ t = newtemp(); gen(t, ':=', Term'.inh, '*', Factor.addr); Term'1.inh =

t; Term'.addr = Term'1.addr } Term' → / Factor Term'

{ t = newtemp(); gen(t, ':=', Term'.inh, '/', Factor.addr); Term'1.inh = t;

Term'.addr = Term'1.addr }

Term' → ε

{ Term'.addr = Term'.inh }

Factor → (Mag)

{ Factor.addr = Mag.addr }

Factor → identifier

{ Factor.addr = identifier.lexeme }

Factor → number

{ Factor.addr = number.lexeme }