<prog> ::= {statlist.next=newLabel()} <statlist> {emitlabel(statlist.next)} EOF

<statlist> ::= {stat.next=newLabel()} <stat> {emitlabel(stat.next)} {statlistp.next=statlist.next} <stalistp>

<statlistp> ::= ; {stat.next=newLabel()} <stat> {emitlabel(stat.next)} {statlistp1.next=statlistp.next} <stalistp1>

| ἐ

<stat> ::= = ID <expr> {emit(istore(id.addr))}

| print {exprlist.operazione=1} (<exprlist>)

| read(ID) {emit(invokestatic(read))} {emit(istore(id.addr))}

| cond {whenlist.next=stat.next} <whenlist> else {stat1.next=stat.next} <stat1>

| while {bexpr.true=newLabel(), bexpr.false=stat.next} {stat1.next=newLabel()} {emitlabel(stat1.next)} ( <bexpr>)

{emitlabel(bexpr.vero)} <stat> { emit(GOto, stat1.next)}

| {{statlist.next=stat.next} <statlist>}

<whenlist> ::= {whenitem.next=whenlist.next} <whenitem> {whenlistp.next=whenlist.next} <whenlistp>

<whenlistp> ::= {whenitem.next=whenlistp.next} <whenitem> {whenlistp1.next=whenlistp.next} <whenlstp1>

| ἐ

<whenitem> ::= when {bexpr.true=newLabel(), bexpr.false=stat.newLabel()} <bexpr> do {emitlabel(bexpr.vero)} {stat1.next=stat.next}

<stat> { emit(GOto, stat.next)} {emitlabel(bexpr.falso)}

<bexpr> ::= RELOP <expr> <expr> {emit(if\_xxxxxx, bexpr.vero)} {emit(GOto, bexpr.falso)}

<expr> ::= + {exprlist.operazione= plus.tag} (<exprlist>)

| \* {exprlist.operazione= mult.tag} (<exprlist>)

| - <expr> <expr> {emit(isub)}

| / <expr> <expr> {emit(idiv)}

| NUM {emit(ldc(num\_value))}

| ID {emit(iload(id.addr))}

<exprlist> ::= <expr> if(<exprlist>.operazione=1) emit(invokestatic(print))} {<exprlistp>.operazione=<exprlist>.operazione} <exprlistp>

<exprlistp> ::= <expr> {if(<exprlistp>.operazione =plus.tag) emit(iadd)

else if(<exprlistp>.operazione =mult.tag) emit(imul)

else if(<exprlistp>.operazione=1) emit(invokestatic(print))}

{<exprlistp1>.operazione=<exprlistp>.operazione} <exprlistp1>

| ἐ