**Program:**

BEGIN

INTEGER a;

INTEGER b;

INTEGER c;

b := 5;

a := 2;

c := a\*b;

WRITE("a is : ");

WRITELN(a);

WRITE("b is : ");

WRITELN(b);

WRITELN("Multiplication Operators:");

WRITE("c = a\*b is : ");

WRITELN(c);

c := b/a;

WRITE("c = b/a is : ");

WRITELN(c);

c := b AND a;

WRITE("c = b AND a is : ");

WRITELN(c);

c := a REM b;

WRITE("c = b REM a is : ");

WRITELN(c);

WRITELN("Addition Operators:");

c := a+b;

WRITE("c = a+b is : ");

WRITELN(c);

c := b-a;

WRITE("c = b-a is : ");

WRITELN(c);

c := b OR a;

WRITE("c = b OR a is : ");

WRITELN(c);

b := 5;

a := 1;

WRITE("a is : ");

WRITELN(a);

WRITE("b is : ");

WRITELN(b);

WRITELN("While Loop Operation:");

WHILE b > 0 DO

BEGIN

b := b - 1;

WRITE("b inside while loop is : ");

WRITELN(b);

END;

WRITELN("If statement Operation and Relational Operator test:");

WRITE("a < b =   ");

IF a < b

THEN WRITELN(a);

WRITE("a > b =   ");

IF a > b

THEN WRITELN(a);

WRITE("a = b  =   ");

IF a = b

THEN WRITELN(a);

WRITE("a != b =   ");

IF a != b

THEN WRITELN(a);

END.

**Code Generation Output by Parser:**

.text

main:

la $a0 ProgBegin

li $v0 4

syscall

li $t0 5

sw $t0 4($sp)

lw $t0 4($sp)

sw $t0 -4($sp)

li $t0 2

sw $t0 8($sp)

lw $t0 8($sp)

sw $t0 0($sp)

lw $t1 0($sp)

lw $t2 -4($sp)

mult $t1, $t2

mflo $t3

sw $t3  12($sp)

lw $t0 12($sp)

sw $t0 -8($sp)

la $a0 lable0

li $v0 4

syscall

lw $t1 0($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

la $a0 lable1

li $v0 4

syscall

lw $t1 -4($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

la $a0 lable2

li $v0 4

syscall

la $a0 newline

li $v0 4

syscall

la $a0 lable3

li $v0 4

syscall

lw $t1 -8($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

lw $t1 -4($sp)

lw $t2 0($sp)

div $t1, $t2

mflo $t3

sw $t3  16($sp)

lw $t0 16($sp)

sw $t0 -8($sp)

la $a0 lable4

li $v0 4

syscall

lw $t1 -8($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

lw $t1 -4($sp)

lw $t2 0($sp)

and $t0, $t1, $t2

sw $t0  20($sp)

lw $t0 20($sp)

sw $t0 -8($sp)

la $a0 lable5

li $v0 4

syscall

lw $t1 -8($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

lw $t1 -4($sp)

lw $t2 0($sp)

div $t1, $t2

mfhi $t3

sw $t3  24($sp)

lw $t0 24($sp)

sw $t0 -8($sp)

la $a0 lable6

li $v0 4

syscall

lw $t1 -8($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

la $a0 lable7

li $v0 4

syscall

la $a0 newline

li $v0 4

syscall

lw $t1 0($sp)

lw $t2 -4($sp)

add $t0, $t1, $t2

sw $t0  28($sp)

lw $t0 28($sp)

sw $t0 -8($sp)

la $a0 lable8

li $v0 4

syscall

lw $t1 -8($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

lw $t1 -4($sp)

lw $t2 0($sp)

sub $t0, $t1, $t2

sw $t0  32($sp)

lw $t0 32($sp)

sw $t0 -8($sp)

la $a0 lable9

li $v0 4

syscall

lw $t1 -8($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

lw $t1 -4($sp)

lw $t2 0($sp)

or $t0, $t1, $t2

sw $t0  36($sp)

lw $t0 36($sp)

sw $t0 -8($sp)

la $a0 lable10

li $v0 4

syscall

lw $t1 -8($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

li $t0 5

sw $t0 40($sp)

lw $t0 40($sp)

sw $t0 -4($sp)

li $t0 1

sw $t0 44($sp)

lw $t0 44($sp)

sw $t0 0($sp)

la $a0 lable11

li $v0 4

syscall

lw $t1 0($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

la $a0 lable12

li $v0 4

syscall

lw $t1 -4($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

la $a0 lable13

li $v0 4

syscall

la $a0 newline

li $v0 4

syscall

loop0:

li $t0 0

sw $t0 48($sp)

lw $t1 48($sp)

lw $t2 -4($sp)

bgt $t1, $t2 ,ConditionLableW0

li $t0 1

sw $t0 52($sp)

lw $t1 -4($sp)

lw $t2 52($sp)

sub $t0, $t1, $t2

sw $t0  56($sp)

lw $t0 56($sp)

sw $t0 -4($sp)

la $a0 lable14

li $v0 4

syscall

lw $t1 -4($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

j loop0

ConditionLableW0:

la $a0 lable15

li $v0 4

syscall

la $a0 newline

li $v0 4

syscall

la $a0 lable16

li $v0 4

syscall

lw $t1 0($sp)

lw $t2 -4($sp)

bgt $t1, $t2 ,ConditionLable0

lw $t1 0($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

ConditionLable0:

la $a0 lable17

li $v0 4

syscall

lw $t1 0($sp)

lw $t2 -4($sp)

blt $t1, $t2 ,ConditionLable1

lw $t1 0($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

ConditionLable1:

la $a0 lable18

li $v0 4

syscall

lw $t1 0($sp)

lw $t2 -4($sp)

bne $t1, $t2 ,ConditionLable2

lw $t1 0($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

ConditionLable2:

la $a0 lable19

li $v0 4

syscall

lw $t1 0($sp)

lw $t2 -4($sp)

beq $t1, $t2 ,ConditionLable3

lw $t1 0($sp)

move $a0 $t1

li $v0 1

syscall

la $a0 newline

li $v0 4

syscall

ConditionLable3:

la $a0 ProgEnd

li $v0 4

syscall

li $v0 10

syscall

.data

ProgBegin: .asciiz  "Program Begin\n"

ProgEnd: .asciiz  "\nProgram End\n"

newline : .asciiz "\n"

lable0: .asciiz "a is : "

lable1: .asciiz "b is : "

lable2: .asciiz "Multiplication Operators:"

lable3: .asciiz "c = a\*b is : "

lable4: .asciiz "c = b/a is : "

lable5: .asciiz "c = b AND a is : "

lable6: .asciiz "c = b REM a is : "

lable7: .asciiz "Addition Operators:"

lable8: .asciiz "c = a+b is : "

lable9: .asciiz "c = b-a is : "

lable10: .asciiz "c = b OR a is : "

lable11: .asciiz "a is : "

lable12: .asciiz "b is : "

lable13: .asciiz "While Loop Operation:"

lable14: .asciiz "b inside while loop is : "

lable15: .asciiz "If statement Operation and Relational Operator test:"

lable16: .asciiz "a < b =   "

lable17: .asciiz "a > b =   "

lable18: .asciiz "a = b  =   "

lable19: .asciiz "a != b =   "

**Little Algol Output:**

Program Begin

a is : 2

b is : 5

Multiplication Operators:

c = a\*b is : 10

c = b/a is : 2

c = b AND a is : 0

c = b REM a is : 1

Addition Operators:

c = a+b is : 7

c = b-a is : 3

c = b OR a is : 7

a is : 1

b is : 5

While Loop Operation:

b inside while loop is : 4

b inside while loop is : 3

b inside while loop is : 2

b inside while loop is : 1

b inside while loop is : 0

b inside while loop is : -1

If statement Operation and Relational Operator test:

a < b =   a > b =   1

a = b  =   a != b =   1

Program End

-- program is finished running --

