Testes para o Compilador

Contents

1 Testes para o Compilador MP

1.1 Teste 1: programa sintaticamente correto

```
program
    integer : weight, group, charge, distance;
begin
   distance := 2300;
   read weight;
   if weight > 60 then group := 5
                  else group := (weight + 14) / 15
   endif;
   charge := 40 + 3 * (distance / 1000)
   write charge
end
```

1.2 Teste 2: programa sintaticamente correto

```
program
    char    : group;
    integer : weight, charge, distance;
begin
   distance := 2300;
   read weight;
   if weight > 60 then group := 5 + 97
                  else group := (weight + 14) / 15   + 97
   endif;
   charge := 36 + 2 * (distance / 1000)
   write(charge)
end
```

1.3 Teste 3: arranjos - Multiplicação de Matrizes - programa sintaticamente correto

```
program
    type matrix = (1:10,1:10) integer;
          matrix : a;
          matrix : b;
          matrix : ab;
procedure readmatrix(reference matrix: m):
    integer : i;
    integer : j;
begin
   i := 1;
   while i <= 2 do
       j := 1;
       while j <= 2 do
           read m(i,j);
           j := j + 1
       endwhile;
       i := i + 1
   endwhile
end;

procedure writematrix(value matrix:m):
    integer : i, j;
begin
   i := 1;
   while i <= 2 do
       j := 1;
       while j <= 2 do
           write(m(i,j));
           j := j + 1
       endwhile;
       i := i + 1
   endwhile
end;
```

1.4 ··· Teste 3: arranjos - Multiplicação de Matrizes - programa sintaticamente correto

```
procedure multiplymatrices(value matrix:m1, value matrix:m2, reference  matrix:product):
    integer : i, k, j;
    integer : cross;
begin
   i := 1;
   repeat
       i := i + 1;
       j := 1;
       while j <= 2 do
           cross := 0;
           k := 1;
           while k <= 2 do
              cross := cross + m1(i,k) * m2(k,j);
     k := k + 1
           endwhile;
           product(i, j) := cross;
           j := j + 1
       endwhile
   until i = 2
end;

begin
   readmatrix(a);
   readmatrix(b);
   multiplymatrices(a, b, ab);
   writematrix(ab)
end
```

1.5 Teste 4: arranjos - Multiplicação de Matrizes - programa com erros (redeclaração de parâmetro, e linha 13)

```
program
    type matrix = (1:10,1:10) integer;
         matrix : a;
         matrix : b;
         matrix : ab;
procedure readmatrix(reference matrix: m):
    integer : i;
    integer : j;
begin
   i := 1;
   while 1 <= 10 do
       j := 1;
       while do
           read m(i,j);
           j := j + 1;
       endwhile;
       i := i + 1;
   endwhile
end;

procedure writematrix(value matrix: m):
    integer : i, j;
begin
   i := 1;
   while i <= 10 do
       write(i);
       j := 1;
       while j <= 10 do
           write(m(i,j));
           j := j + 1;
       endwhile;
       write(i);
       i := i + 1
   endwhile
end;
```

1.6 ··· Teste 4: arranjos - Multiplicação de Matrizes - programa com erros

```
procedure multiplymatrices(value matrix:m1, value matrix:m1, reference matrix: product):
    integer : i, k, j;
    integer : cross;
begin
   i := 1;
   repeat
       j := 1;
       while j <= 10 do
           cross := 0;
           k := 1;
           while k <= 10 do
              cross := cross + m1(i,k) * m2(k,j);
      k := k + 1
           endwhile;
           product(i, j) := cross;
           j := j + 1
       endwhile
   until i = 2
end;

begin
   read a;
   read b;
   multiplymatrices(a, b, ab);
   write(ab)
end
```

1.7 Teste 5: comando WHILE - programa correto

```
program
    integer : i;
begin
   i := 20;
   while (1 > 10) do
       write(i+10);
       i := i - 1
   endwhile;
   write i
end
```

1.8 Teste 6: comando REPEAT - programa sintaticamente correto

```
program
    integer : i;
begin
   i := 20;
   repeat
      write(i+10);
      i := i - 1
   until (i<10);
   write i
end
```

1.9 Teste 7: comando REPEAT - ERRO DE TIPO (condição do repeat)

```
program
    integer : weight, group;
    integer : charge;
    integer : distance;
begin
   weight := 0;
   repeat
      weight := weight + 1;
      group := group * 2
   until weight + 10
end
```

1.10 Teste 8: MISCELANIA - programa sintaticamente correto

```
program

    /* Arranjos com varias dimensoes: */

    type matriz = (1:20, 0:30,10:50, 2:10) integer;
         integer: i, j;
         matriz: m, n, z;

    /* Arranjos como valor, referencia, e valor resultado */

integer procedure doit(value matriz: m, reference matriz: n, value result matriz: z):
    integer : i;
    integer : j;
begin
   m(7, 3, 15, 5) := i; /* arranjo como l-value */
   if i < j then
      i := j;
      i := 0
   else
      j := i;
      j := 0
   endif;

   while true do    /* loop com constante booleana */
       i := i - 1;
       i := j / i;
       j := n(3, 5, 8);
       i := doit(n, m, z)    /* chamada recursiva */
   endwhile

   return i+1
end;

begin
  repeat /* expressao complexa envolvendo arranjos: */

    i := i + 2 - j / i ** 2 + 5 - m(2, 3, 4) + n(3, 4, 5) **-m(10, 29, 7);
    j := i ** -(1/2)
  until i = 0;
```

```
  if i not= j then /* varios exits dentro de um laco */
     while i not= j do
        read i;
if (i = 0) then exit;
           i := i + 1;
           read j;
   if (j <= 0) then exit;
   j := j - 1;
     endwhile
  endif;

  i := doit(m, n, z)   /* chamada a procedimento retornado valor */
end
```