Codificação de Programas em Português Estruturado

```
Programa ALG1
Início
  Escreva "Bom dia"
Fim
```

```
Programa ALG2
Var
  X : Inteiro
Início
  Leia X
  Escreva X
Fim
```

```
Programa ALG3
Var
  X : Inteiro
  Y : Inteiro
Início
  Leia X
  Y ← X ↑ 2
  Escreva Y
Fim
```

```
Programa ALG4
Var
  X : Inteiro
  Y : Inteiro
  Z : Inteiro
Início
  Leia X
  Leia Y
  Z ← X + Y
  Escreva Z
Fim
```

```
Programa ALG5
Var
  X : Inteiro
  Y : Inteiro
  Z : Inteiro
Início
  Leia X
  Leia Y
  Z ← X ↑ 2 + Y ↑ 2
  Escreva Z
Fim
```

```
Programa ALG6
Var
  X : Inteiro
Início
  Leia X
  Se (X > 100) Então
    Escreva X
  Fim_se
Fim
```

```
Programa ALG7
Var
  X : Inteiro
  Y : Inteiro
  Z : Inteiro
Início
  Leia X
  Leia Y
  Se (X > 100) Então
    Z ← X + Y
    Escreva Z
  Fim_se
Fim
```

```
Programa ALG8
Var
  X : Inteiro
  Y : Inteiro
Início
  Leia X
  Leia Y
  Se (X <= Y) Então
    Escreva X
  Senão
    Escreva Y
  Fim_se
Fim
```

```
Programa ALG9
Var
  X : Inteiro
  Y : Inteiro
Início
  Leia X
  Se (X >= 10) Então
    Y ← X ↑ 2
  Senão
    Y ← X ↑ 3
  Fim_se
  Escreva Y
Fim
```

```
Programa ALG10
Var
  X  : Inteiro
  Y  : Inteiro
  N1 : Inteiro
  N2 : Inteiro
Início
  Leia X
  Leia Y
  Se (X > Y) Então
    N1 ← Y
    N2 ← X
  Senão
    N1 ← X
    N2 ← Y
  Fim_se
  Escreva N1
  Escreva N2
Fim
```

```
Programa ALG11
Var
  X : Inteiro
  I : Inteiro
Início
  X ← 0
  I ← 1
  Enquanto (I <= 10) Faça
    Escreva X
    X ← X + 2
    I ← I + 1
  Fim_enquanto
Fim
```

```
Programa ALG12
Var
  X : Inteiro
  I : Inteiro
Início
  X ← 1
  I ← 1
  Enquanto (I <= 10) Faça
    Escreva X
    X ← X * 2
    I ← I + 1
  Fim_enquanto
Fim
```

Programas em Linguagem PASCAL

```
Program ALG01;
Begin
  WriteLn('Bom dia');
End.
```

```
Program ALG02;
Var
  X : Integer;
Begin
  ReadLn(X);
  WriteLn(X);
End.
```

```
Program ALG03;
Var
  X : Integer;
  Y : Integer;
Begin
  ReadLn(X);
  Y := X * X;
  WriteLn(Y);
End.
```

```
Program ALG04;
Var
  X : Integer;
  Y : Integer;
  Z : Integer;
Begin
  ReadLn(X);
  ReadLn(Y);
  Z := X + Y;
  WriteLn(Z);
End.
```

```
Program ALG05;
Var
  X : Integer;
  Y : Integer;
  Z : Integer;
Begin
  ReadLn(X);
  ReadLn(Y);
  Z := X * X + Y * Y;
  WriteLn(Z);
End.
```

```
Program ALG06;
Var
  X : Integer;
Begin
  ReadLn(X);
  If (X > 100) Then
    WriteLn(X);
End.
```

```
Program ALG07;
Var
  X : Integer;
  Y : Integer;
  Z : Integer;
Begin
  ReadLn(X);
  ReadLn(Y);
  If (X > 100) Then
    Begin
      Z := X + Y;
      WriteLn(Z);
    End;
End.
```

```
Program ALG08;
Var
  X : Integer;
  Y : Integer;
Begin
  ReadLn(X);
  ReadLn(Y);
  If (X <= Y) Then
    WriteLn(X)
  Else
    WriteLn(Y);
End.
```

```
Program ALG09;
Var
  X : Integer;
  Y : Integer;
Begin
  ReadLn(X);
  If (X >= 10) Then
    Y := X * X
  Else
    Y := X * X * X;
  WriteLn(Y);
End.
```

```
Program ALG10;
Var
  X  : Integer;
  Y  : Integer;
  N1 : Integer;
  N2 : Integer;
Begin
  ReadLn(X);
  ReadLn(Y);
  If (X > Y) Then
    Begin
      N1 := Y;
      N2 := X;
    End
  Else
    Begin
      N1 := X;
      N2 := Y;
    End;
  WriteLn(N1);
  WriteLn(N2);
End.
```

```
Program ALG11;
Var
  X : Integer;
  I : Integer;
Begin
  X := 0;
  I := 1;
  While (I <= 10) Do
    Begin
      WriteLn(X);
      X := X + 2;
      I := I + 1;
    End;
End.
```

```
Program ALG12;
Var
  X : Integer;
  I : Integer;
Begin
  X := 1;
  I := 1;
  While (I <= 10) Do
    Begin
      WriteLn(X);
      X := X * 2;
      I := I + 1;
    End;
End.
```

Codificação de Programas em Linguagem Structured BASIC

```
REM ALG01
PRINT "Bom dia"
END
```

```
REM ALG02
DIM x AS INTEGER
INPUT x
PRINT x
END
```

```
REM ALG03
DIM x AS INTEGER
DIM y AS INTEGER
INPUT x
y = x ^ 2
PRINT y
END
```

```
REM ALG04
DIM x AS INTEGER
DIM y AS INTEGER
DIM z AS INTEGER
INPUT x
INPUT y
z = x + y
PRINT z
END
```

```
REM ALG05
DIM x AS INTEGER
DIM y AS INTEGER
DIM z AS INTEGER
INPUT x
INPUT y
z = x ^ 2 + y ^ 2
PRINT z
END
```

```
REM ALG06
DIM x AS INTEGER
INPUT x
IF (x > 100) THEN
    PRINT x
END IF
END
```

```
REM ALG07
DIM x AS INTEGER
DIM y AS INTEGER
DIM z AS INTEGER
INPUT x
INPUT y
IF (x > 100) THEN
    z = x + y
    PRINT z
END IF
END
```

```
REM ALG08
DIM x AS INTEGER
DIM y AS INTEGER
INPUT x
INPUT y
IF (x <= y) THEN
    PRINT x
ELSE
    PRINT y
END IF
END
```

```
REM ALG09
DIM x AS INTEGER
DIM y AS INTEGER
INPUT x
IF (x >= 10) THEN
    y = x ^ 2
ELSE
    y = x ^ 3
END IF
PRINT y
END
```

```
REM ALG10
DIM x AS INTEGER
DIM y AS INTEGER
DIM n1 AS INTEGER
DIM n2 AS INTEGER
INPUT x
INPUT y
IF (x > y) THEN
    n1 = y
    n2 = x
ELSE
    n1 = x
    n2 = y
END IF
PRINT n1
PRINT n2
END
```

```
REM ALG11
DIM x AS INTEGER
DIM i AS INTEGER
x = 0
i = 1
WHILE (i <= 10)
    PRINT x
    x = x + 2
    i = i + 1
WEND
END
```

```
REM ALG12
DIM x AS INTEGER
DIM i AS INTEGER
x = 1
i = 1
WHILE (i <= 10)
    PRINT x
    x = x * 2
    i = i + 1
WEND
END
```

Codificação de Programas em Linguagem C

```c
/* ALG01 */
#include <stdio.h>
int main()
{
  printf("Bom dia\n");
  return 0;
}
```

```c
/* ALG02 */
#include <stdio.h>
int X;
int main()
{
  scanf("%i", &X);
  printf("%i\n", X);
  return 0;
}
```

```c
/* ALG03 */
#include <stdio.h>
#include <math.h>
int X;
int Y;
int main()
{
  scanf("%i", &X);
  Y = pow(X, 2);
  printf("%i\n", Y);
  return 0;
}
```

```c
/* ALG04 */
#include <stdio.h>
int X;
int Y;
int Z;
int main()
{
  scanf("%i", &X);
  scanf("%i", &Y);
  Z = X + Y;
  printf("%i\n", Z);
  return 0;
}
```

```c
/* ALG05 */
#include <stdio.h>
#include <math.h>
int X;
int Y;
int Z;
int main()
{
  scanf("%i", &X);
  scanf("%i", &Y);
  Z = pow(X, 2) + pow(Y, 2);
  printf("%i\n", Z);
  return 0;
}
```

```c
/* ALG06 */
#include <stdio.h>
int X;
int main()
{
  scanf("%i", &X);
  if (X > 100)
    printf("%i\n", X);
  return 0;
}
```

```c
/* ALG07 */
#include <stdio.h>
int X;
int Y;
int Z;
int main()
{
  scanf("%i", &X);
  scanf("%i", &Y);
  if (X > 100)
    {
      Z = X + Y;
      printf("%i\n", Z);
    }
  return 0;
}
```

```c
/* ALG08 */
#include <stdio.h>
int X;
int Y;
int main()
{
  scanf("%i", &X);
  scanf("%i", &Y);
  if (X <= Y)
    printf("%i\n", X);
  else
    printf("%i\n", Y);
  return 0;
}
```

```c
/* ALG09 */
#include <stdio.h>
#include <math.h>
int X;
int Y;
int main()
{
  scanf("%i", &X);
  if (X >= 10)
    Y = pow(X, 2);
  else
    Y = pow(X, 3);
  printf("%i\n", Y);
  return 0;
}
```

```c
/* ALG10 */
#include <stdio.h>
int X;
int Y;
int N1;
int N2;
int main()
{
  scanf("%i", &X);
  scanf("%i", &Y);
  if (X > Y)
    {
      N1 = Y;
      N2 = X;
    }
  else
    {
      N1 = X;
      N2 = Y;
    }
  printf("%i\n", N1);
  printf("%i\n", N2);
  return 0;
}
```

```c
/* ALG11 */
#include <stdio.h>
int X;
int I;
int main()
{
  X = 0;
  I = 1;
  while (I <= 10)
    {
      printf("%i\n", X);
      X = X + 2;
      I = I + 1;
    }
  return 0;
}
```

```c
/* ALG12 */
#include <stdio.h>
int X;
int I;
int main()
{
  X = 1;
  I = 1;
  while (I <= 10)
    {
      printf("%i\n", X);
      X = X * 2;
      I = I + 1;
    }
  return 0;
}
```

Codificação de Programas em Linguagem C++

```
// ALG01
#include <iostream>
using namespace std;
int main(void)
{
  cout << "Bom dia" << endl;
  return 0;
}
```

```
// ALG02
#include <iostream>
using namespace std;
int main(void)
{
  int X;
  cin >> X;
  cout << X << endl;
  return 0;
}
```

```
// ALG03
#include <iostream>
#include <cmath>
using namespace std;
int main(void)
{
  int X;
  int Y;
  cin >> X;
  Y = pow(X, 2);
  cout << Y << endl;
  return 0;
}
```

```
// ALG04
#include <iostream>
using namespace std;
int main(void)
{
  int X;
  int Y;
  int Z;
  cin >> X;
  cin >> Y;
  Z = X + Y;
  cout << Z << endl;
  return 0;
}
```

```
// ALG05
#include <iostream>
#include <cmath>
using namespace std;
int main(void)
{
  int X;
  int Y;
  int Z;
  cin >> X;
  cin >> Y;
  Z = pow(X, 2) + pow(Y, 2);
  cout << Z << endl;
  return 0;
}
```

```
// ALG06
#include <iostream>
using namespace std;
int main(void)
{
  int X;
  cin >> X;
  if (X > 100)
    cout << X << endl;
  return 0;
}
```

```
// ALG07
#include <iostream>
using namespace std;
int main(void)
{
  int X;
  int Y;
  int Z;
  cin >> X;
  cin >> Y;
  if (X > 100)
    {
      Z = X + Y;
      cout << Z << endl;
    }
  return 0;
}
```

```
// ALG08
#include <iostream>
using namespace std;
int main(void)
{
  int X;
  int Y;
  cin >> X;
  cin >> Y;
  if (X <= Y)
    cout << X << endl;
  else
    cout << Y << endl;
  return 0;
}
```

```
// ALG09
#include <iostream>
#include <cmath>
using namespace std;
int main(void)
{
  int X;
  int Y;
  cin >> X;
  if (X >= 10)
    Y = pow(X, 2);
  else
    Y = pow(X, 3);
  cout << Y << endl;
  return 0;
}
```

```
// ALG10
#include <iostream>
using namespace std;
int main(void)
{
  int X;
  int Y;
  int N1;
  int N2;
  cin >> X;
  cin >> Y;
  if (X > Y)
    {
      N1 = Y;
      N2 = X;
    }
  else
    {
      N1 = X;
      N2 = Y;
    }
  cout << N1 << endl;
  cout << N2 << endl;
  return 0;
}
```

```
// ALG11
#include <iostream>
using namespace std;
int main(void)
{
  int X = 0;
  int I = 1;
  while (I <= 10)
    {
      cout << X << endl;
      X = X + 2;
      I = I + 1;
    }
  return 0;
}
```

```
// ALG12
#include <iostream>
using namespace std;
int main(void)
{
  int X = 1;
  int I = 1;
  while (I <= 10)
    {
      cout << X << endl;
      X = X * 2;
      I = I + 1;
    }
  return 0;
}
```

Codificação de Programas em Linguagem D

```
// ALG01
import std.stdio;
void main()
{
  writeln("Bom dia!");
}
```

```
// ALG02
import std.stdio;
void main()
{
  int X;
  readf(" %s", &X);
  writeln(X);
}
```

```
// ALG03
import std.stdio;
void main()
{
  int X;
  int Y;
  readf(" %s", &X);
  Y = X ^^ 2;
  writeln(Y);
}
```

```
// ALG04
import std.stdio;
void main()
{
  int X;
  int Y;
  int Z;
  readf(" %s", &X);
  readf(" %s", &Y);
  Z = X + Y;
  writeln(Z);
}
```

```
// ALG05
import std.stdio;
void main()
{
  int X;
  int Y;
  int Z;
  readf(" %s", &X);
  readf(" %s", &Y);
  Z = X ^^ 2 + Y ^^ 2;
  writeln(Z);
}
```

```
// ALG06
import std.stdio;
void main()
{
  int X;
  readf(" %s", &X);
  if (X > 100)
    writeln(X);
}
```

```
// ALG07
import std.stdio;
void main()
{
  int X;
  int Y;
  int Z;
  readf(" %s", &X);
  readf(" %s", &Y);
  if (X > 100)
    {
      Z = X + Y;
      writeln(Z);
    }
}
```

```
// ALG08
import std.stdio;
void main()
{
  int X;
  int Y;
  readf(" %s", &X);
  readf(" %s", &Y);
  if (X <= Y)
    writeln(X);
  else
    writeln(Y);
}
```

```
// ALG09
import std.stdio;
void main()
{
  int X;
  int Y;
  readf(" %s", &X);
  if (X >= 10)
    Y = X ^^ 2;
  else
    Y = X ^^ 3;
  writeln(Y);
}
```

```
// ALG10
import std.stdio;
void main()
{
  int X;
  int Y;
  int N1;
  int N2;
  readf(" %s", &X);
  readf(" %s", &Y);
  if (X > Y)
    {
      N1 = Y;
      N2 = X;
    }
  else
    {
      N1 = X;
      N2 = Y;
    }
  writeln(N1);
  writeln(N2);
}
```

```
// ALG11
import std.stdio;
void main()
{
  int X = 0;
  int Y = 1;
  while (I <= 10)
    {
      writeln(X);
      X = X + 2;
      I = I + 1;
    }
}
```

```
// ALG12
import std.stdio;
void main()
{
  int X = 1;
  int Y = 1;
  while (I <= 10)
    {
      writeln(X);
      X = X * 2;
      I = I + 1;
    }
}
```

Codificação em Linguagem Ada

```
-- ALG01
WITH Ada.Text_IO;
USE Ada;
PROCEDURE ALG01 IS
BEGIN
  Text_IO.Put("Bom dia");
END ALG01;
```

```
-- ALG02
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG02 IS
  X : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  Integer_Text_IO.Put(X);
END ALG02;
```

```
-- ALG03
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG03 IS
  X : Integer;
  Y : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  Y := X ** 2;
  Integer_Text_IO.Put(Y);
END ALG03;
```

```
-- ALG04
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG04 IS
  X : Integer;
  Y : Integer;
  Z : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  Integer_Text_IO.Get(Y);
  Z := X + Y;
  Integer_Text_IO.Put(Z);
END ALG04;
```

```
-- ALG05
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG05 IS
  X : Integer;
  Y : Integer;
  Z : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  Integer_Text_IO.Get(Y);
  Z := X ** 2 + Y ** 2;
  Integer_Text_IO.Put(Z);
END ALG05;
```

```
-- ALG06
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG06 IS
  X : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  IF (X > 100) THEN
    Integer_Text_IO.Put(X);
  END IF;
END ALG06;
```

```
-- ALG07
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG07 IS
  X : Integer;
  Y : Integer;
  Z : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  Integer_Text_IO.Get(Y);
  IF (X > 100) THEN
    Z := X + Y;
    Integer_Text_IO.Put(Z);
  END IF;
END ALG07;
```

```
-- ALG08
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG08 IS
  X : Integer;
  Y : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  Integer_Text_IO.Get(Y);
  IF (X <= Y) THEN
    Integer_Text_IO.Put(X);
  ELSE
    Integer_Text_IO.Put(Y);
  END IF;
END ALG08;
```

```
-- ALG09
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG09 IS
  X : Integer;
  Y : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  IF (X >= 10) THEN
    Y := X ** 2;
  ELSE
    Y := X ** 3;
  END IF;
  Integer_Text_IO.Put(Y);
END ALG09;
```

```
-- ALG10
WITH Ada.Integer_Text_IO;
USE Ada;
PROCEDURE ALG10 IS
  X  : Integer;
  Y  : Integer;
  N1 : Integer;
  N2 : Integer;
BEGIN
  Integer_Text_IO.Get(X);
  Integer_Text_IO.Get(Y);
  IF (X > Y) THEN
    N1 := Y;
    N2 := X;
  ELSE
    N1 := X;
    N2 := Y;
  END IF;
  Integer_Text_IO.Put(N1);
  Integer_Text_IO.Put(N2);
END ALG10;
```

```
-- ALG11
WITH Ada.Integer_Text_IO;
WITH Ada.Text_IO;
USE Ada;
PROCEDURE ALG11 IS
  X : Integer;
  I : Integer;
BEGIN
  X := 0;
  I := 1;
  WHILE (I <= 10) LOOP
    Integer_Text_IO.Put(X);
    Text_IO.New_Line;
    X := X + 2;
    I := I + 1;
  END LOOP;
END ALG11;
```

```
-- ALG12
WITH Ada.Integer_Text_IO;
WITH Ada.Text_IO;
USE Ada;
PROCEDURE ALG12 IS
  X : Integer;
  I : Integer;
BEGIN
  X := 1;
  I := 1;
  WHILE (I <= 10) LOOP
    Integer_Text_IO.Put(X);
    Text_IO.New_Line;
    X := X * 2;
    I := I + 1;
  END LOOP;
END ALG12;
```

Codificação em Linguagem Lua

```
-- ALG01
io.write("Bom dia\n")
```

```
-- ALG02
X = io.read("*number")
io.write(X, "\n")
```

```
-- ALG03
X = io.read("*number")
Y = X ^ 2
io.write(Y, "\n")
```

```
-- ALG04
X = io.read("*number")
Y = io.read("*number")
Z = X + Y
io.write(Z, "\n")
```

```
-- ALG05
X = io.read("*number")
Y = io.read("*number")
Z = X ^ 2 + Y ^ 2
io.write(Z, "\n")
```

```
-- ALG06
X = io.read("*number")
if (X > 100) then
  io.write(X, "\n")
end
```

```
-- ALG07
X = io.read("*number")
Y = io.read("*number")
if (X > 100) then
  Z = X + Y
  io.write(Z, "\n")
end
```

```
-- ALG08
X = io.read("*number")
Y = io.read("*number")
if (X <= Y) then
  io.write(X, "\n")
else
  io.write(Y, "\n")
end
```

```
-- ALG09
X = io.read("*number")
if (X >= 10) then
  Y = X ^ 2
else
  Y = X ^ 3
end
io.write(Y)
```

```
-- ALG10
X = io.read("*number")
Y = io.read("*number")
if (X > Y) then
  N1 = Y
  N2 = X
else
  N1 = X
  N2 = Y
end
io.write(N1, "\n")
io.write(N2, "\n")
```

```
-- ALG11
X = 0
I = 1
while (I <= 10) do
  io.write(X, "\n")
  X = X + 2
  I = I + 1
end
```

```
-- ALG12
X = 1
I = 1
while (I <= 10) do
  io.write(X, "\n")
  X = X * 2
  I = I + 1
end
```

Codificação em Linguagem Python

```
# ALG01
print("Bom dia")
```

```
# ALG02
X = int(input())
print(X)
```

```
# ALG03
X = int(input())
Y = X ** 2
print(Y)
```

```
# ALG04
X = int(input())
Y = int(input())
Z = X + Y
print(Z)
```

```
# ALG05
X = int(input())
Y = int(input())
Z = X ** 2 + Y ** 2
print(Z)
```

```
# ALG06
X = int(input())
if (X > 100):
  print(X)
```

```
# ALG07
X = int(input())
Y = int(input())
if (X > 100):
  Z = X + Y
  print(Z)
```

```
# ALG08
X = int(input())
Y = int(input())
if (X <= Y):
  print(X)
else:
  print(Y)
```

```
# ALG09
X = int(input())
if (X >= 10):
  Y = X ** 2
else:
  Y = X ** 3
print(Y)
```

```
# ALG10
X = int(input())
Y = int(input())
if (X > Y):
  N1 = Y
  N2 = X
else:
  N1 = X
  N2 = Y
print(N1)
print(N2)
```

```
# ALG11
X = 0
I = 1
while (I <= 10):
  print(X)
  X = X + 2
  I = I + 1
```

```
# ALG12
X = 1
I = 1
while (I <= 10):
  print(X)
  X = X * 2
  I = I + 1
```

Codificação de Programas em Linguagem Classic BASIC

```
10 REM ALG01
20 PRINT "Bom dia"
30 END
```

```
10 REM ALG02
20 INPUT X
30 PRINT X
40 END
```

```
10 REM ALG03
20 INPUT X
30 LET Y = X ^ 2
40 PRINT Y
50 END
```

```
10 REM ALG04
20 INPUT X
30 INPUT Y
40 LET Z = X + Y
50 PRINT Z
60 END
```

```
10 REM ALG05
20 INPUT X
30 INPUT Y
40 LET Z = X ^ 2 + Y ^ 3
50 PRINT Z
60 END
```

```
10 REM ALG06
20 INPUT X
30 IF (X > 100) THEN GOTO 50
40 GOTO 60
50 PRINT X
60 END
```

```
10 REM ALG07
20 INPUT X
30 INPUT Y
40 IF (X > 100) THEN GOTO 60
50 GOTO 80
60 LET Z = X + Y
70 PRINT Z
80 END
```

```
10 REM ALG08
20 INPUT X
30 INPUT Y
40 IF (X <= Y) THEN GOTO 50 ELSE GOTO 70
50 PRINT X
60 GOTO 80
70 PRINT Y
80 END
```

```
10 REM ALG09
20 INPUT X
30 IF (X >= 10) THEN GOTO 40 ELSE GOTO 60
40 LET Y = X ^ 2
50 GOTO 70
60 LET Y = X ^ 3
70 PRINT Y
80 END
```

```
 10 REM ALG10
 20 INPUT X
 30 INPUT Y
 40 IF (X > Y) THEN GOTO 50 ELSE GOTO 80
 50 LET N1 = Y
 60 LET N2 = X
 70 GOTO 100
 80 LET N1 = X
 90 LET N2 = Y
100 PRINT N1
110 PRINT N2
120 END
```

```
10 REM ALG11
20 LET X = 0
30 LET I = 1
40 IF (I <= 10) THEN GOTO 50 ELSE GOTO 90
50 PRINT X
60 X = X + 2
70 I = I + 1
80 GOTO 40
90 END
```

```
10 REM ALG12
20 LET X = 1
30 LET I = 1
40 IF (I <= 10) THEN GOTO 50 ELSE GOTO 90
50 PRINT X
60 X = X * 2
70 I = I + 1
80 GOTO 40
90 END
```

Codificação de Programas em Linguagem COMAL

```
10 // ALG01
20 PRINT "Bom dia"
30 END
```

```
10 // ALG02
20 INPUT x#
30 PRINT x#
40 END
```

```
10 // ALG03
20 INPUT x#
30 y# := x# ^ 2
40 PRINT y#
50 END
```

```
10 // ALG04
20 INPUT x#
30 INPUT y#
40 z# := x# + y#
50 PRINT z#
60 END
```

```
10 // ALG05
20 INPUT x#
30 INPUT y#
40 z# := x# ^ 2 + y# ^ 2
50 PRINT z#
60 END
```

```
10 // ALG06
20 INPUT x#
30 IF (x# > 100) THEN
40   PRINT x#
50 ENDIF
60 END
```

```
10 // ALG07
20 INPUT x#
30 INPUT y#
40 IF (x# > 100) THEN
50   z# := x# + y#
60   PRINT z#
70 ENDIF
80 END
```

```
10 // ALG08
20 INPUT x#
30 INPUT y#
40 IF (x# <= y#) THEN
50   PRINT x#
60 ELSE
70   PRINT y#
80 ENDIF
90 END
```

```
10 // ALG09
20 INPUT x#
30 IF (x# >= 10) THEN
40   y# := x# ^ 2
50 ELSE
60   y# := x# ^ 3
70 ENDIF
80 PRINT y#
90 END
```

```
10 // ALG10
20 INPUT x#
30 INPUT y#
40 IF (x# > y#) THEN
50   n1# := y#
60   n2# := x#
70 ELSE
80   n1# := x#
90   n2# := y#
100 ENDIF
110 PRINT n1#
120 PRINT n2#
130 END
```

```
10 // ALG11
20 x# := 0
30 i# := 1
40 WHILE (i# <= 10) DO
50   PRINT x#
60   x# := x# + 2
70   i# := i# + 1
80 ENDWHILE
90 END
```

```
10 // ALG12
20 x# := 1
30 i# := 1
40 WHILE (i# <= 10) DO
50   PRINT x#
60   x# := x# * 2
70   i# := i# + 1
80 ENDWHILE
90 END
```