**4.8 Bubble sort (冒泡法排序)**

**Algorithm** BubbleSort

**Variables:**

N: Integer

*tabnb*: array[1..1000] of Real

*k*, *step*, *i*: Integer

*inter*: Real

*permut*: Boolean

**Instructions:**

Write(“Enter the number N of elements (1<= N <= 1000) : ” !)

Read(KBD! N)

For *k* From 1 to N

Write(“Enter your ”, *k*, “-th element: ” !)

Read(KBD! tabnb[*k*])

EndFor

*permut* ← True

*step* ← 1

While(step <= N-1 AND *permut* = True) Do

*permut* ← False

For *i* From 1 To N-step

If(*tabnb*[*i*] > *tabnb*[*i*+1]) Then

*inter* ← *tabnb*[i]

*tabnb*[*i*] ← *tabnb*[*i*+1]

*tabnb*[*i*+1] ← *inter*

*permut* ← True

EndIf

EndFor

*step* ← *step* + 1

EndWhile

**End** BubbleSort

**4.9 Comparison of 2 arrays of characters**

DATA:

* Two arrays of characters, read from KBD

Result:

* An indicator, of Boolean type, to be printed on screen

*Set a list of require variables:*

* String1, string2: array[1..1001] of Characters
* *i*: Integer
* *N1,N2*: Integer
* *flag*: Boolean

Type of LOOP: undetermined loop: While

**Algorithm** ComparisonStrings

**Variables:**

stringA, stringB: array[1..1001] of Character

*i*: Integer

*N1,N2*: Integer

*flag*: Boolean

**Instructions:**

Write(“Enter the first string: ” !)

Read(KBD! stringA[1])

*i* ← 1

While(stringA[*i*]≠’\0’) Do

*i* ← *i* + 1

Read(KBD! stringA[*i*])

EndWhile

*N1* ← *i* – 1

Write(“Enter the second string: ” !)

Read(KBD! stringB[1])

*i* ← 1

While(stringB[*i*]≠’\0’) Do

*i* ← *i* + 1

Read(KBD! stringB[*i*])

EndWhile

*N2* ← *i* – 1

*flag* ← True

*i* ← 1

If(*N1* ≠ *N2*) Then

*flag* ← False

Else

While (*i*≤ *N1* AND *flag* = True) Do

If (stringA[*i*]≠stringB[*i*]) Then

*flag* ← False

EndIf

*i* ← *i* + 1

EndWhile

EndIf

If (flag = True) Then

Write(“The two arrays are the same.”)

Else

Write(“The two arrays are different.”)

EndIf

**End** ComparisonStrings

**4.14 Triangular linear system**

DATA:

* Coefficients of equations, real type, aii ≠ 0, read from KBD
* Constant numbers, real type, read from KBD

Result:

* The values of xi, 1≤i≤n, array[1..n] of reals, to be printed on screen.

Adequate structure to store the data: two-dimensional array: array[1..n,1..n+1] of real

Results: unidimensional array: array[1..n] of real

Recurrence relations:

Loop: determined loop: For

**Algorithm** TriangularLS

**Variables:**

Matrix: array [1..1000,1..1001] of Real

X: array[1..1000] of Real

*i, j*: Integer

*n*: Integer

*sum*: Real

**Instructions:**

Write(“Enter the number of variables, n<=1000: ” !)

Read(KBD! *n*)

Write(“Enter the coefficients and constants, line by line” !)

For *i* From 1 To *n*

For *j* From 1 to *i*+1

Read(KBD! Matrix[*i, j*])

EndFor

EndFor

X[1] ← Matrix[1,2]/Matrix[1,1]

For *i* From 2 To *n*

*sum* ← 0

For j From 1 To *i*-1

*sum* ← *sum* + Matrix[*i*,*j*]\*X[j]

EndFor

X[*i*] ← (Matrix[*i*, *i*+1] – *sum*)/ Matrix[*i*, *i*]

EndFor

Write(“The variables are: \n” !)

For *i* From 1 To *n*

Write(X[*i*], “ ” !)

EndFor

**End** TriangularLS

**4.15 Complex numbers**

**Solution1 : array type, no pre-definition is needed**

**Algorithm** Complex\_Cal

**Variables:**

Cpxnb1, Cpxnb2: array[1..2] of Real

**Instructions:**

Write(“Enter the real and imaginary parts of the first number : ” !)

Read(KBD! Cpxnb1[1], Cpxnb1[2])

Write(“Enter the real and imaginary parts of the second number : ” !)

Read(KBD! Cpxnb2[1], Cpxnb2[2])

Write(“The real part of the sum is: ”, Cpxnb1[1]+Cpxnb2[1], “ and the imaginary part is: ”, Cpxnb1[2]+Cpxnb2[2] !)

Write(“The real part of the product is: ”, Cpxnb1[1]\*Cpxnb2[1]- Cpxnb1[2]\*Cpxnb2[2], “ and the imaginary part is: ”, Cpxnb1[1]\*Cpxnb2[2]+ Cpxnb1[2]\*Cpxnb2[1] !)

**End** Complex\_Cal

**Solution2: define new type using article**

**Types:**

CompNb : article ( Re: Real

Im: Real

)

**Algorithm** Complex\_Cal

**Variables:**

Cpxnb1, Cpxnb2: CompNb

**Instructions:**

Write(“Enter the real and imaginary parts of the first number : ” !)

Read(KBD! Cpxnb1.Re, Cpxnb1.Im)

Write(“Enter the real and imaginary parts of the second number : ” !)

Read(KBD! Cpxnb2.Re, Cpxnb2.Im)

Write(“The real part of the sum is: ”, Cpxnb1.Re + Cpxnb2.Re, “ and the imaginary part is: ”, Cpxnb1.Im + Cpxnb2.Im !)

Write(“The real part of the product is: ”, Cpxnb1.Re\*Cpxnb2.Re  Cpxnb1.Im\*Cpxnb2.Im, “ and the imaginary part is: ”, Cpxnb1.Re\*Cpxnb2.Im + Cpxnb1.Im\*Cpxnb2.Re !)

**End** Complex\_Cal