10.26 TD

2018年10月26日 8:00

4.17 Books

1. Types

TBook : article (title: array[1..100] of character,
author: array[1..50] of character,
edition: array[1..200] of character,
editionYear: integer
ISBN: array [1..5] integer
canBeBorrowed: boolean
isBorrowed: Boolean
returnDate: array [1..3] of integer

2. Algo :

Variables :
myBook : TBook index : integer

Instructions :

myBook . returnDate[1] ← 2018
myBook . returnDate[2] ← 10
myBook . returnDate[3] ← 26

As an example, we want to assign the return of myBook[2018. 10, 26]

---

write ("Give the name of author"!)
index ← 1
Read(KBD! myBook.author[index])
while (myBook. author[index] ≠ '\0')
| index ← index + 1
| read (KBD! myBook.author[index])
End While.

As a second example, we want to assign the author of this book.

We need to read the first character and while the character is ≠ '\0' we read the next one.

Remember special character for ending a string of character.

---

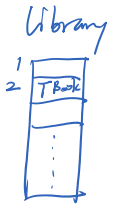
3. Algo :

Only 3 variables

Variables :
myBook: TBook
index: integer
library: array[1..300] of TBook

Library[2].returnDate[1] ← 2018
Library[2].returnDate[2] ← 12
Library[2].returnDate[3] ← 25

As an example, we want to assign the return date of the second book of the library.




```
For index from 1 to 300
    write ("give the year")
    read (KBD! library[index].returnDate[1])
    write ("give the month")
    read (KBD! library[index].returnDate[2])
    :
End For
```

As a third Example, to assign the return Date of each book of this library.

Chapter 5 Sub-algorithms

5.1 Fahrenheit

$$T_f = \frac{9}{5} \times T_c + 32$$

SA: FahrenheitToCelsius
IN: T1: real fahrenheit temp
OUT: T2: real celsius temp
Variables: \
Instructions:
$T_2 \leftarrow (T_1 - 32) \times \frac{5}{9}$
EndSA

SA: CelsiusToFahrenheit.
IN: T1: real celsius temp
OUT: T2: real fahrenheit temp
Variables: \
Instructions:
$T_2 \leftarrow \frac{9}{5} \times T_1 + 32$
EndSA

Remember!

All data in a SA are local to this SA. In other words, (T1, T2) of first SA have nothing in common with the 2nd one.

Algo: MainAlgo
Variables:

Variables:

givenTemp: real This is the temperature given by the user.

givenChoice: character 'C' or 'F'

transformTemp: real

Instructions:

```
Write("Give your temp"!)
Read (KBD! givenTemp)
Write ("Give your choice"!)
Read (KBD! givenChoice)
if (givenChoice = 'C')
| CelsiusTo Fahrenheit (givenTemp! transformtemp)
End if
if (givenChoice = 'F')
| FahrenheitTo Celsius (givenTemp! transformtemp)
End if
```

* Each SA has a single output parmter. In such a case, these SA's are functions. The second way for calling a function is:

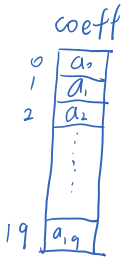
```
If (givenChoice = 'C')
| write (CelsiusTo Fahrenheit (givenTemp!)!)
End If
```

5.4 Polynomial

$a_3x^3+a_2x^2+a_1x+a_0$

1. Types:

TPoly: article (degree: integer,
coeff: array [0..10] of real)



coefficient: 系数

2. step1: Find the smallest degree between the two poly (minDegree: integer).

step2: In a For loop with index from 0 to minDegree, we assign the coefficient of the resulting poly with the sum of coefficient of both poly.

step3: We assign the remaining coefficients of the resulting poly with the remainly coefficient of the poly having the bigest degree.

SA: SumPoly

IN: P1, P2: TPoly We want $P_3 = P_1 + P_2$

OUT: P3: TPoly resulting Poly

Variable: minDegree : integer

Instructions:

Step 1

If P1.degree < P2.degree
    minDegree ← P1.degree
else
    minDegree ← P2.degree
EndIf

For index from 0 to minDegree
    P3.coeff[index] ← P1.coeff[index] + P2.coeff[index]
End For

If minDegree = P1.degree
    For index from (minDegree+1) to P2.degree
        P3.coeff[index] ← P2.coeff[index]
    End For
    P3.degree ← P2.degree
Else
    For index from (minDegree+1) to P1.degree
        P3.coeff[index] ← P1.coeff[index]
    End For
    P3.degree ← P1.degree
End If

Homework (with not?)

SA: Prod Poly
IN: P1, P2 : TPoly
OUT: P3 : TPoly
⋮ $P_3 = P_1 * P_2$

SA: Comp Val Poly
IN: P1: TPoly
x: real
OUT: value: real
⋮ $P_1(x)$

SA: InputValPoly
IN: \
OUT: P1: Poly
⋮

Algo:

Variables: P1, P2, P3, Sum, Proel: TPoly
x: real

Instructions:

Write ("Give the first Poly"!)
InputValPoly (! P1)
Write ("Give the second Poly"!)
InputValPoly (! P2)
Write ("Give the third Poly"!)
InputValPoly (! P3)
SumPoly (P1, P2 ! sum)
ProdPoly (sum, P3 ! prod)
CompValPoly (prod ! value)
Write (value!)