Kiểu dữ liệu văn bản

Bởi:

PGS. TS. NGƯT Phạm Văn Huấn

Ngoài những dữ liệu số như các số nguyên, số thực, máy tính còn có thể lưu giữ và xử lý những dữ liệu văn bản như những chữ cái, những đoạn văn bản, những chữ số và một số ký hiệu khác. Trong Fortran gọi chung những dữ liệu này là dữ liệu ký tự. Trong chương này chúng ta xét thêm những đặc điểm khai báo những dữ liệu ký tự, một số thao tác với những dữ liệu ký tự và ứng dụng của chúng trong xử lý thông tin.

Tập các ký tự của Fortran

Tập ký tự của Fortran gồm 26 chữ cái tiếng Anh, mười chữ số từ 0 đến 9, dấu trống và 12 ký hiệu sau đây:

+ ? * / = ( ) , . ' $:

Ngoài ra còn một số ký tự khác tùy thuộc vào những hệ máy tính khác nhau.

Các hằng ký tự bao giờ cũng nằm trong cặp dấu nháy trên. Trong hằng ký tự dấu nháy trên ' được biểu thị bằng hai dấu nháy trên ‘’ (không phải dấu ngoặc kép). Thí dụ chữ LET'S của tiếng Anh sẽ được viết là 'LET''S'.

Thông thường người ta xử lý trong máy tính những từ, những dòng chữ gồm một số ký tự ghép lại với nhau. Trong trường hợp đó người ta gọi là *xâu ký tự*. *Độ dài của xâu ký tự* là số ký tự được ghép lại trong xâu ký tự đó. Một ký tự cũng có thể coi là một xâu ký tự với độ dài bằng 1. Do đó, ta gọi chung dữ liệu xâu ký tự là dữ liệu ký tự hay dữ liệu văn bản. Dưới đây là thí dụ về các hằng ký tự và độ dài tương ứng của chúng:

'CHU NHAT'	8 ký tự
'SENSOR 23'	9 ký tự
'08:40?13:25'	11 ký tự
'LE QUY DON'	10 ký tự
' '	2 ký tự

""""	2 ký tự

Các dạng khai báo biến ký tự

? Biến ký tự được khai báo bằng lệnh mô tả dạng tổng quát như sau:

CHARACTER * n Danh sách biến

trong đó *n* chỉ số ký tự (độ dài) trong mỗi xâu ký tự. Thí dụ lệnh

CHARACTER * 8 TEN, NGAY

chỉ rằng TEN và NGAY là những biến chứa 8 ký tự mỗi biến.

? Lệnh **CHARACTER** biến thể sau đây cho phép ta khai báo những biến ký tự với độ dài khác nhau trên cùng một dòng lệnh

CHARACTER TITLE * 10, NUOC * 2

? Một mảng chứa một số phần tử, mỗi phần tử có giá trị là một xâu ký tự được khai báo bằng một trong hai cách tương đương như sau:

CHARACTER * 4 NAME (50)

CHARACTER NAME (50) * 4

? Các xâu ký tự cũng có thể được dùng trong các chương trình con. Xâu ký tự phải được khai báo bằng lệnh CHARACTER trong cả chương trình chính và chương trình con. Cũng như các mảng dữ liệu số nguyên, số thực, trong chương trình con có thể khai báo biến ký tự mà không cần chỉ định rõ độ dài xâu và kích thước mảng. Thí dụ:

CHARACTER * (*) BCC

CHARACTER * (*) NAME (N)

Nhập, xuất dữ liệu ký tự

Khi xâu ký tự được dùng trong lệnh xuất toàn bộ xâu được in ra. Những dấu trống được tự động chèn vào xâu để tách riêng xâu ký tự với những mục in khác cùng dòng. Trong lệnh nhập, giá trị của biến ký tự phải được bao trong cặp dấu nháy trên. Nếu số ký tự trong cặp dấu nháy nhiều hơn so với độ dài đã mô tả của biến ký tự, thì những ký tự thừa ở bên phải sẽ bị bỏ qua (bị cắt bỏ); nếu số ký tự ít hơn ? những vị trí thừa ở bên phải được tự động điền bằng các dấu trống. Để in xâu ký tự trong lệnh xuất có định dạng, có

thể dùng đặc tả A (*chú ý*, có thể không cần chỉ rõ số vị trí dành cho mục in). Thí dụ: với đoạn chương trình:

```
CHARACTER * 20 THU

PRINT *, ' HAY NHAP MOT NGAY TRONG TUAN'

READ *, THU

PRINT 5, THU

5 FORMAT (1X, 'NGAY VUA NHAP LA ', A)

END
```

thì tương tác trên màn hình sẽ như sau:

```
HAY NHAP MOT NGAY TRONG TUAN
'CHU NHAT' ↵
NGAY VUA NHAP LA CHU NHAT
```

Thấy rằng số ký tự gõ vào biến THU chỉ bằng 8, không dài tới 20 như đã khai báo. Nhưng khi in ra màn hình ta không thấy rõ những dấu trống được tự động điền vào phía bên phải. Nếu lệnh FORMAT của lệnh in có dạng

```
FORMAT (1X, A, ' LA NGAY VUA NHAP')
```

thì trên màn hình sẽ thấy rõ những dấu trống như sau:

```
HAY NHAP MOT NGAY TRONG TUAN
'CHU NHAT'
CHU NHAT          LA NGAY VUA NHAP
```

Những thao tác với dữ liệu ký tự

Gán các giá trị ký tự

Những giá trị ký tự có thể được gán cho các biến ký tự bằng lệnh gán và một hằng ký tự. Nếu hằng có độ dài nhỏ hơn số ký tự đã khai báo của biến, thì các dấu trống sẽ tự

động được điền vào bên phải; nếu hằng có độ dài lớn hơn - các ký tự thừa sẽ bị bỏ qua. Thí dụ:

CHARACTER * 4 MONHOC (3)

MONHOC (1) = 'TOAN'

MONHOC (2) = 'LY'

MONHOC (3) = 'HOA HOC'

Trong những lệnh trên đây ta khai báo mảng MONHOC gồm 3 phần tử, mỗi phần tử là một xâu dài 4 ký tự. Vậy trong MONHOC (1) sẽ lưu 'TOAN', trong MONHOC (2) sẽ lưu 'LY**bb**', trong MONHOC (3) sẽ lưu 'HOA**b**' (chữ **b** chỉ dấu trống). Qua thí dụ này ta thấy tầm quan trọng của việc sử dụng các xâu có cùng độ dài mô tả của biến; nếu không các lệnh sẽ xử lý sai.

Một biến ký tự cũng có thể được gán giá trị của biến ký tự khác bằng lệnh gán, thí dụ

CHARACTER * 4 LOAI1, LOAI2

LOAI1 = 'GIOI'

LOAI2 = LOAI1

Sau lệnh gán này, cả hai biến LOAI1 và LOAI2 đều lưu xâu ký tự 'GIOI'.

Lệnh DATA cũng có thể dùng để khởi xướng giá trị của các biến ký tự. Thí dụ sau gán 12 tên tháng tiếng Anh vào mảng THANG:

	CHARACTER * 3 THANG (12)
	DATA THANG / 'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
*	'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec' /

So sánh các giá trị ký tự

Biểu thức lôgic trong lệnh IF lôgic cũng có thể là một phép so sánh các biến, hằng ký tự. Thí dụ, nếu các biến THANG, CH, TG là những biến ký tự, các lệnh sau đây là những lệnh đúng:

IF (THANG .EQ. 'FEB') NGAY = 28

```
IF (CH (I) .GT. CH (I+1)) THEN

TG = CH (I)

CH (I) = CH (I+1)

CH (I+1) = TG

END IF
```

Khi đánh giá một biểu thức lôgic với các xâu ký tự, trước hết chương trình xét độ dài của hai xâu. Nếu một xâu ngắn hơn xâu khác, thì xâu ngắn hơn được bổ sung thêm các dấu trống ở bên phải sao cho hai xâu trở thành có cùng độ dài. Việc so sánh hai xâu ký tự cùng độ dài thực hiện từ trái sang phải theo từng ký tự một. Hai xâu bằng nhau nếu chúng có cùng những ký tự trong cùng một thứ tự. Các ký tự được so sánh với nhau theo *chuỗi thứ tự so sánh* (collating sequence). Chuỗi này liệt kê các ký tự từ thấp đến cao. Thí dụ, một phần của chuỗi thứ tự so sánh đối với các ký tự ASCII liệt kê các ký tự dưới đây:

Chuỗi thứ tự so sánh của các ký tự:

```
b"#$%&()*+,-./

0123456789

:;=?@

ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Theo chuỗi này, những so sánh sau là đúng:

```
'A1' < 'A2'

'JOHN' < 'JOHNSTON'

'175' < '176'

'THREE' < 'TWO'

'$' < 'DOLLAR'
```

Nếu các xâu ký tự *chỉ chứa các chữ cái*, thì thứ tự từ thấp đến cao là thứ tự alphabê, được gọi là *thứ tự từ vựng* (lexicographic ordering).

Trích ra xâu con

Xâu con là một phần được trích ra từ xâu xuất phát và giữ nguyên thứ tự ban đầu. Trong Fortran xâu con được viết bằng *tên của xâu xuất phát, kèm theo hai biểu thức nguyên nằm trong cặp dấu ngoặc đơn, cách nhau bởi dấu hai chấm. Biểu thức thứ nhất chỉ vị trí đầu tiên ở xâu xuất phát mà từ đó xâu con được trích ra. Biểu thức thứ hai chỉ vị trí cuối cùng*. Thí dụ, nếu xâu 'FORTRAN' được lưu trong biến LANG, ta có thể có những xâu con như sau

Biến Xâu con

LANG (1 **:** 1) 'F'

LANG (1 **:** 7) 'FORTRAN'

LANG (2 **:** 3) 'OR'

LANG (7 **:** 7) 'N'

1) Ta có thể không viết biểu thức thứ nhất trong cặp dấu ngoặc đơn nếu giá trị của nó bằng 1 và có thể không viết biểu thức thứ hai nếu giá trị của nó bằng độ dài của xâu xuất phát. Ta cũng có thể không viết cả hai biểu thức. Nhưng trong cả ba trường hợp vẫn phải có dấu hai chấm (**:**) ở trong cặp dấu ngoặc. Thí dụ:

LANG (**:**4) là 'FORT'

LANG (5**:**) là 'RAN'

LANG (**:**) là 'FORTRAN'

2) Khi phép trích ra xâu con sử dụng cùng một tên biến, các biểu thức trong cặp dấu ngoặc đơn không được phủ lên nhau. Thí dụ, nếu biến LANG chứa xâu 'FORMATS', thì lệnh

LANG (7: 7) = LANG (6: 6)

sẽ biến giá trị của LANG thành 'FORMATT'. Nhưng lệnh sau đây sẽ sai không thể thực hiện được

LANG (3: 5) = LANG (2: 4)

3) Những trường hợp như: các vị trí đầu hoặc cuối không phải là số nguyên, là số âm, vị trí đầu lớn hơn vị trí cuối, vị trí đầu hoặc vị trí cuối có giá trị lớn hơn độ dài mô tả của xâu con, việc trích ra xâu con sẽ không thể thực hiện đúng đắn.

Thí dụ 32: Đếm số ký tự trong một văn bản. Giả sử một bức điện dài 50 ký tự. Hãy đếm số từ trong bức điện đó. Ta biết rằng trong một văn bản soạn đúng thì các từ cách nhau bằng một dấu trống, do đó ta chỉ cần đếm số dấu trống trong văn bản và số từ sẽ bằng số dấu trống cộng thêm một. Với trường hợp này chương trình sau sẽ đếm được đúng số từ:

```
      CHARACTER * 50 MESSGE
      INTEGER COUNT, I
      COUNT = 0
      DO 10 I = 1, 50
      IF (MESSGE (I: I) .EQ. ' ') COUNT = COUNT + 1
10    CONTINUE
      PRINT 5, COUNT + 1
5     FORMAT (1X, 'BUC DIEN GOM ', I2, ' TU')
      END
```

Kết hợp các xâu ký tự

Kết hợp hay *cộng* là thao tác ghép hai hoặc một số xâu ký tự vào thành một xâu ký tự. Thao tác này thực hiện bởi hai dấu gạch chéo //. Thí dụ muốn có từ WORKED ta có thể dùng phép kết hợp

'WORK' // 'ED'

Nhóm lệnh sau đây cho phép viết ra ngày tháng theo quy cách tiếng Việt, tức thêm các gạch chéo ngăn cách giữa các ký hiệu ngày, tháng và năm:

```
CHARACTER DAY*2,MONTH*2,YEAR*4,DATE*10

READ *, DAY, MONTH, YEAR

DATE = DAY//'/'//MONTH//'/'//YEAR

PRINT *, DATE
```

END

Theo nhóm lệnh này, nếu khi thực hiện lệnh READ ta gõ từ bàn phím '05' '10' '1999' ? thì trên màn hình sẽ in ra:

05/10/1999.

Những hàm chuẩn xử lý xâu ký tự

? **HàmINDEX**

Hàm này có hai đối số kiểu xâu ký tự, đưa ra một số nguyên chỉ vị trí của xâu thứ hai trong xâu thứ nhất. Thí dụ nếu ta có biến STR chứa mệnh đề 'TO BE OR NOT TO BE' và dùng lệnh

K = INDEX (STR, 'BE')

thì biến K sẽ có giá trị 4 vì xâu 'BE' xuất hiện lần đầu tiên trong xâu STR ở vị trí thứ 4.

? **HàmLEN**

Hàm LEN có một đối số kiểu xâu ký tự, nó đưa ra một số nguyên chỉ độ dài của xâu đó. Hàm này rất có ích trong những chương trình con chấp nhận các xâu ký tự độ dài bất kỳ nhưng cần biết độ dài thực tế ở trong chương trình con.

Thí dụ 33: Cấu tạo tên viết tắt của người. Viết chương trình đọc từ bàn phím họ tên đầy đủ (gồm họ, chữ đệm và tên) của một người và in lên màn hình dạng viết tắt. (Thí dụ, nếu nhập vào họ tên đầy đủ như sau:

TRAN CONG MINH,

thì dạng in ra sẽ là

T. C. MINH.

Chương trình NAMEED dưới đây cho phép ta gõ từ bàn phím một xâu ký tự gồm cả họ, chữ đệm và tên trên cùng một dòng nhưng cách nhau bởi một dấu trống. Thủ tục con EXTR cho phép tách riêng phần họ, chữ đệm và tên dựa vào vị trí các dấu trống trong họ tên đầy đủ. Sau đó thủ tục EDIT ghép các chữ cái đầu tiên của phần họ, chữ đệm kèm theo các dấu chấm và dấu trống với tên để cấu tạo thành tên viết tắt.

	PROGRAM NAMEED
	CHARACTER HO *10, DEM *10, TEN *20, HOTEN *25

```
      PRINT *, 'Nhap ho, chu dem, ten cach nhau 1 dau trong'
      READ 5, HOTEN
5     FORMAT (A)
      CALL EXTR (HOTEN, HO, DEM, TEN)
      CALL EDIT (HO, DEM, TEN, HOTEN)
      PRINT *, HOTEN
      END
      SUBROUTINE EXTR (XHOTEN, XHO, XDEM, XTEN)
      CHARACTER * (*) XHO, XTEN, XDEM, XHOTEN
      INTEGER B1, B2
      B1 = INDEX (XHOTEN, ' ')
      B2 = B1 + INDEX (XHOTEN (B1 + 1:) , ' ')
      XHO = XHOTEN (:B1-1)
      XDEM = XHOTEN (B1+1: B2-1)
      XTEN = XHOTEN (B2+1:)
      RETURN
      END
      SUBROUTINE EDIT (XHO, XDEM, XTEN, XHOTEN)
      INTEGER L
      CHARACTER *(*) XHO, XTEN, XDEM, XHOTEN
      XHOTEN = XHO(1: 1) // '.'
      L = INDEX (XHOTEN, ' ') + 1
      XHOTEN (L: L + 2) = XDEM (1: 1) // '. '
      XHOTEN (L + 3:) = XTEN
      RETURN
      END
```

? **Các hàmCHAR** và **ICHAR**

Các hàm này thao tác với các ký tự trong *chuỗi thứ tự so sánh* dùng trong máy tính. Nếu một máy tính có 256 ký tự trong chuỗi thứ tự so sánh của nó, thì các ký tự này được đánh số từ 0 đến 255. *Hàm***CHAR** *nhận một đối số nguyên và đưa ra một ký tự trong chuỗi thứ tự so sánh ở vị trí ứng với số nguyên đó*. *Hàm***ICHAR***là hàm ngược của hàm* **CHAR**. Nó nhận đối số là biến một ký tự và trả về một số nguyên ứng với vị trí của ký tự đó ở trong chuỗi thứ tự so sánh.

Vì các máy tính khác nhau có các chuỗi thứ tự so sánh khác nhau, nên các hàm này có thể dùng để xác định vị trí của những ký tự trong chuỗi thứ tự so sánh.

Thí dụ, nếu bạn muốn in ra màn hình tất cả các ký tự trong chuỗi thứ tự so sánh được dùng trong máy tính của mình từ vị trí 0 đến 255 có thể dùng chương trình sau:

```
PROGRAM CSCHAR

DO I = 0, 255

PRINT *, I, ' ', CHAR (I)

END DO

END
```

Chương trình sau đây cho phép in ra màn hình vị trí của các chữ cái in hoa tiếng Anh, những chữ cái thường và những chữ số từ 0 đến 9 trong chuỗi thứ tự so sánh trong máy tính bạn đang dùng:

```
PROGRAM COLSEQ

CHARACTER *70 SET

SET (1: 26) = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

SET (27: 52) = 'abcdefghijklmnopqrstuvwxyz'

SET (53: 62) = '0123456789'

DO I = 1, 62

PRINT *, SET (I : I), ICHAR (SET (I : I))

END DO

END
```

Với các máy tính thông dụng ngày nay, nếu chạy chương trình này, ta sẽ thấy tập các chữ số từ '**0**' đến '**9**' tuần tự có vị trí từ **48** đến **57**, tập các chữ cái hoa tiếng Anh từ '**A**' đến '**Z**' có vị trí từ **65** đến **90** và tập các chữ cái thường tiếng Anh từ '**a**' đến '**z**' có vị trí từ **97** đến **122** trong chuỗi thứ tự so sánh. Các vị trí còn lại trong chuỗi thứ tự so sánh sẽ ứng với những ký tự khác, trong đó có những ký tự chuyên dùng để biểu diễn các chữ cái Hy Lạp, các ký tự dùng để kẻ biểu bảng... Ta có thể khai thác những chi tiết này để viết những thủ tục rất có ích như in biểu bảng khá đẹp khi xuất dữ liệu lên màn hình, tự động tạo các tên file trong chương trình... khi cần thiết.

? **Các hàmLGE, LGT, LLE, LLT**

Những hàm này cho phép ta so sánh những xâu văn bản dựa trên *chuỗi thứ tự so sánh ASCII*. Những hàm này sẽ có ích nếu một chương trình có so sánh các xâu hay sắp xếp ký tự và được dùng trong các máy tính khác nhau. Những hàm này trả về một giá trị lôgic - TRUE hoặc FALSE tùy thuộc kết quả so sánh hai đối số kiểu xâu ký tự. Thí dụ, nếu ta có hai biến ký tự XAU1, XAU2 thì **LGE (XAU1, XAU2)** sẽ cho giá trị TRUE nếu XAU1 lớn hơn hoặc bằng XAU2 về phương diện từ vựng. Các hàm **LGT, LLE** và **LLT** thực hiện các phép so sánh "*lớn hơn về từ vựng*", "*nhỏ hơn hoặc bằng về từ vựng*" và "*nhỏ hơn về từ vựng*". Nhớ rằng *các hàm này dựa trên chuỗi thứ tự so sánh ASCII chứ không phải chuỗi thứ tự so sánh của máy tính*.

Trong Fortran 90 còn có **các hàmADJUSTL, ADJUSTR** dùng để dồn một xâu ký tự về trái hoặc về phải bằng cách cắt bỏ những dấu trống ở phía trái hoặc ở phía phải của xâu đó. **HàmTRIM** cắt bỏ những dấu trống ở đuôi một xâu văn bản và giảm độ dài xâu cho tương xứng

Trong thực tế hàm này và cả hàm **LEN** nữa không làm việc đúng như người ta mô tả nó trong tài liệu, độ dài xâu văn bản nhận được vẫn chỉ là độ dài mô tả chứ không phải độ dài thực tế.

.

Thí dụ 34: Sắp xếp danh sách theo thứ tự alphabê. Viết chương trình đọc từ bàn phím tên và số điện thoại của 20 người. In lên màn hình danh sách sắp xếp thứ tự alphabê theo tên người. Trong thí dụ này ta sử dụng các hàm so sánh đối với *bảng thứ tự so sánh ASCII*.

	PROGRAM NMSORT
	CHARACTER *8 TEN(20), TEL (20), TEMP
	DO I = 1, 20
	PRINT *, 'NHAP TEN NGUOI THU ', I
	READ 5, TEN(I)

```
      PRINT *, 'SO DIEN THOAI'
      READ 5, TEL (I)
      ENDDO
5     FORMAT (A)
      DO I = 1, 19
      K = I
      DO J = I+1, 20
      IF (LGT (TEN (K), TEN (J))) K = J
      END DO
      TEMP = TEN (K)
      TEN (K) = TEN (I)
      TEN (I) = TEMP
      TEMP = TEL (K)
      TEL (K) = TEL (I)
      TEL (I) = TEMP
      PRINT *, TEN (I), TEL (I)
      END DO
      PRINT *, TEN (20), TEL (20)
      END
```

Thí dụ 35:Mã hóa bức điện. Mã hóa bức điện là làm cho dòng văn bản bình thường của bức điện có một dạng khác thường chỉ có người mã hóa mới hiểu được nội dung của nó. Người ta có thể mã hóa một bức điện theo cách sau: Lấy một xâu gồm 62 chữ cái và chữ số làm *khóa*. Từng chữ cái bình thường trong bức điện được mã hóa bằng một chữ cái trong khóa sao cho chữ A bình thường được thay bằng chữ cái đầu tiên trong khóa, chữ B được thay bằng chữ cái thứ hai... Dưới đây là một chương trình nhận từ bàn phím một bức điện và in ra màn hình dạng mã hóa của bức điện đó. Trong chương trình này ta dùng một khóa là chuỗi các chữ cái và chữ số sắp xếp theo thứ tự khác thường như sau:

YXAZKLMBJOCFDVSWTREGHNIPUQ

yxazklmbjocfdvswtreghnipuq9087564312

Việc mã hóa các chữ cái trong bức điện được thực hiện trong thủ tục con ENCODE.

	PROGRAM MSGCOD
	CHARACTER DIEN*255, MADIEN*255,
	CHARACTER KHOA*62, ALPH*62
	ALPH = ‘ABCDEFGHIJKLMNOPQRSTUVWXYZ’ //
*	‘abcdefghijklmnopqrstuvwxyz0123456789'
	KHOA = 'YXAZKLMBJOCFDVSWTREGHNIPUQ’ //
*	‘yxazklmbjocfdvswtreghnipuq9087564312'
	PRINT*, 'ENTER A MESSEAGE’,
*	‘(MAXIMUM 255 LETTERS)'
	READ (5, '(A255)') DIEN
	CALL ENCODE (KHOA, ALPH, DIEN, MADIEN)
	PRINT 5, MADIEN
5	FORMAT (1X, /, 1X, 'THIS IS ENCODED AS' /, 1X, A /)
	END
	SUBROUTINE **ENCODE** (KEY, ALP, MESSGE, SECRET)
	CHARACTER MESSGE * (*), SECRET * (*)
	CHARACTER ALP * (*), KEY * (*), LETTER
	DO I = 1, LEN (MESSGE)
	LETTER = MESSGE (I : I)
	J = INDEX (ALP, LETTER)
	IF (J .EQ. 0) THEN
	SECRET (I : I) = LETTER
	ELSE
	SECRET (I : I) = KEY (J : J)
	END IF
	END DO

	RETURN
	END

Bài tập

1. Các biến K và J sẽ có giá trị bằng bao nhiêu sau khi thực hiện nhóm lệnh sau đây:

CHARACTER *18 STRG

INTEGER K, J

STR = 'TO BE OR NOT TO BE'

K = INDEX (STRG, 'BE')

J = INDEX (STR (K + 1:), 'BE') + K

2. Giả sử các bức điện được mã hóa bằng một khóa như trong thí dụ 31, tức dùng chuỗi các chữ cái và chữ số:

YXAZKLMBJOCFDVSWTREGHNIPUQ

yxazklmbjocfdvswtreghnipuq9087564312

Người ta giải mã như sau: Từng chữ cái trong mã điện sẽ được thay thế bởi một chữ cái trong bảng chữ cái alphabê theo quy tắc: nếu chữ cái trong mã điện trùng với chữ cái thứ nhất trong khóa thì chữ cái đó thay bằng chữ A trong bảng chữ cái alphabê, nếu trùng với chữ cái thứ hai thì thay nó bằng chữ B... Thí dụ, giả sử mã điện là dòng chữ

DKKG YG YJRWSRG EYGHRZYU

thì theo quy tắc trên, ta có bức điện được giải mã như sau:

MEET AT AIRPORT SATURDAY

Viết chương trình cho phép đọc từ bàn phím một bức điện dưới dạng mã hóa và in lên màn hình dạng đã giải mã của nó.

3. Giả sử danh mục số điện thoại của những người quen của bạn lưu trong file TELNUM dưới dạng những dòng gồm tên người đầy đủ và số điện thoại của mỗi người với format A30, A8. File không có dòng đầu báo thông tin về số dòng dữ liệu và cũng không có dòng ký hiệu cuối file báo hết dữ liệu. Hãy viết chương trình đọc vào từ bàn phím một tên người nào đó, sau đó kiểm tra xem người đó có trong danh mục điện thoại của bạn

không. Nếu không có thì đưa ra thông báo 'KHONG CO TRONG DANH MUC', nếu có thì in ra tên người cùng với số điện thoại tìm được sao cho số điện thoại được đặt trong cặp dấu ngoặc ngay sau tên.

4. File dữ liệu ADDR chứa khoảng 50 tên người và địa chỉ. Dòng thứ nhất của mỗi người chứa họ tên đầy đủ (30 ký tự) gồm họ, chữ đệm và tên. Dòng thứ hai chứa địa chỉ số nhà và đường phố (35 ký tự), tên thành phố (15 ký tự) và số điện thoại (15 ký tự). Mỗi xâu ký tự được ghi trong cặp dấu nháy trên. Hãy viết chương trình đọc dữ liệu và in ra thông tin về từng người theo mẫu nhãn sau đây (thí dụ):

HUY, N. Q.

91 NGUYEN THIEN THUAT

NHA TRANG, (058)832536

Mỗi nhãn cách nhau bốn dòng. Chú ý sau tên thành phố là dấu phảy, không nên để một dấu cách nào trước dấu phảy đó.

5. Giả sử bạn đã biết rằng ngày đầu năm của một năm là ngày thứ mấy trong tuần lễ. Hãy viết chương trình in tờ lịch tháng Giêng của năm đó dưới dạng dễ nhìn.

6. Giả sử bạn đã biết ngày đầu năm của một năm nào đó là thứ mấy trong tuần lễ. Hãy viết chương trình in tờ lịch của một tháng, năm bất kỳ trong tương lai dưới dạng dễ nhìn. Tháng và năm nhập từ bàn phím.

7. Viết chương trình in bảng các toán tử lôgic (bảng 4.2, chương 4, trang 56).

8. Viết thủ tục TDBANG (N, TENCOT) trong đó N là đối số nguyên, TENCOT là mảng một chiều gồm N phần tử văn bản chuyên dùng để in ra một tiêu đề cột của bảng. Thí dụ nếu chương trình gọi thủ tục này và chuyển đối số thực tế bằng 12 và một mảng 12 tên viết tắt tháng tiếng Anh ‘JAN’, ’FEB’, ‘MAR’, ‘APR’, ‘MAY’, ‘JUN’, ’JUL’, ‘AUG’, ‘SEP’, ‘OCT’, ‘NOV’, ‘DEC’ thì chương trình sẽ in ra tít đầu bảng có dạng như dưới đây:

JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC

9. Số liệu giá trị ngày của các yếu tố khí tượng thủy văn tại trạm quan trắc được lưu trong file ASCII có quy cách ghi như sau: Dòng trên cùng ghi tên trạm. Dòng thứ 2 có hai số nguyên viết cách nhau lần lượt chỉ tổng số ngày quan trắc và số yếu tố được quan trắc. Dòng thứ ba có 6 số nguyên viết cách nhau lần lượt chỉ ngày, tháng, năm đầu và ngày, tháng, năm cuối quan trắc. Dòng thứ 4 là tiêu đề cột liệt kê tên tất cả các yếu tố được quan trắc, mỗi tên được ghi với độ rộng 8 vị trí và căn bên phải. Các dòng tiếp

theo lần lượt ghi giá trị của các yếu tố, mỗi dòng một ngày. Tính giá trị trung bình tháng của tất cả các yếu tố trong tất cả các năm quan trắc. Kết quả ghi vào những file mới, mỗi yếu tố một file, sao cho tên file trùng với tên của yếu tố quan trắc.