AG43
Semestre Automne 2005
Examen Final
Durée : 1h45

Exercice 1 : Cryptage d'une chaîne de caractères

Une procédure de cryptage de chaîne de caractères consiste à additionner n au code ASCII de chaque caractère de la chaîne en question.
Exemple : A (caractère 65) devient F (70) si n=5.
Si le caractère dépasse 255 alors on repasse à 0 (par exemple le caractère 254 sera devenu 3 avec n=5).

Ecrire la procédure **`decrypt(chaine,n)`** effectuant le décryptage.
Remarque : après le cryptage de la chaîne, le caractère de fin de chaîne est devenu n au lieu de 0.

Exercice 2 : Mise au format d'une chaîne de caractères

Ecrire la procédure **`format(chaine,n)`** permettant d'insérer en fin de chaîne autant de caractères blancs (espaces) que nécessaire pour que sa longueur devienne n.
Si la longueur initiale de la chaîne est supérieure à n alors la chaîne sera raccourcie en plaçant convenablement un caractère de fin de chaîne de sorte que sa longueur devienne n.

Exemple : La chaîne `ch` contenant "test", elle contiendra "test□□□□□□" ("test" suivi de 6 espaces) après appel de `format(ch,10)` et "tes" après appel de `format(ch,3)`.

Exercice 3 : Tableaux et matrices

Dans un programme d'interpolation polynomiale, on dispose d'un tableau X[n] d'abscisses et d'un tableau Y[n] d'ordonnées.
On souhaite assembler la matrice A[m,m] et le second membre B[m] suivants :

$$A = \begin{bmatrix} \sum_{i=1,n} x_i^0 & \sum_{i=1,n} x_i^1 & \cdots & \sum_{i=1,n} x_i^{m-1} \\ \sum_{i=1,n} x_i^1 & \sum_{i=1,n} x_i^2 & \cdots & \sum_{i=1,n} x_i^m \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1,n} x_i^{m-1} & \sum_{i=1,n} x_i^m & \cdots & \sum_{i=1,n} x_i^{2m-2} \end{bmatrix} \text{ et } B = \begin{bmatrix} \sum_{i=1,n} y_i x_i^0 \\ \sum_{i=1,n} y_i x_i^1 \\ \vdots \\ \sum_{i=1,n} y_i x_i^{m-1} \end{bmatrix}$$

où x^m désigne x à la puissance m.
Ecrire l'algorithme de la procédure d'assemblage nommée **`assemble(…)`** dont vous listerez les paramètres et leurs types.
Conseil : établir préalablement les relations permettant de calculer A[k,j] et B[k] où k et j désignent les indices de ligne et de colonne respectivement. (Vous constatez que les puissances des x croissent avec k et j).

Vous disposez de la fonction **`pow(a,b)`** retournant a^b.
L'indiçage des tableaux se fera par rapport à 0.
Remarque : On ne vous demande à aucun moment de résoudre le système par une quelconque méthode.

Exercice 4 : dates

Ecrire l'algorithme d'une fonction qui retourne le jour d'une date donnée (postérieure au 1er janvier 1900). La fonction retournera 1 pour lundi, 2 pour mardi … et 7 pour dimanche. Les dates seront entrées sous forme de tableaux de 3 entiers (jour, mois, annee).
Vous disposez de la fonction **`nbj(date1,date2)`** retournant le nombre de jours entre deux dates (date2 postérieure à date1) ainsi que de la fonction modulo (%).
Le 1er janvier 1900 est un dimanche.

Feuille à rendre signée avec votre copie

Nom, Prénom :	Signature

Exercice 5 : FORTRAN

Trouvez les cinq erreurs qui ont été glissées dans ce programme. Il est inutile de comprendre le fonctionnement du programme. Attention, une erreur indûment signalée peut enlever des points. Portez directement la correction sur cette feuille.

```
PROGRAM Interpolation
!  Programme de test pour subroutine SPLINE
REAL*4 PI
PARAMETER(N=20,PI=3.141593)
INTEGER*2 I,N
REAL*4 yp1,ypn,x(N),y(N),y2(N)

     write(*,*) 'Derivees Secondes pour sin(x) de 0 a PI'
! Construction du tableau pour interpolation
     do i=1,20
          x(i)=i*PI/N
          y(i)=sin(x(i))
     end do

! Calcul de la derivee seconde avec SPLINE
     yp1=cos(x(1))
     ypn=cos(x(N))
     call spline(N,x,y,yp1,ypn,y2)
! Controle des resultats
     write(*,'(t19,a,t35,a)') 'spline','actuel'
     write(*,'(t6,a,t17,a,t33,a)') 'angle',Derivee 2','Derivee 2'

     do i=1,N
          write(*,'(1x,f8.2,2f16.6)') x(i),y2(i),-sin(x(i))
     end do
END
```

```
SUBROUTINE spline(x,y,n,yp1,ypn,y2)
INTEGER*2 n,NMAX
REAL*4 yp1,ypn,x(n),y(n),y2(n)
PARAMETER (NMAX=500)
INTEGER*2 i,k
REAL*4 p,qn,sig,un,u(NMAX)

     if (yp1.gt..99e30) then
          y2(1)=0.
          u(1)=0.
     else
          y2(1)=-0.5
          u(1)=(3./(x(2)-x(1)))*((y(2)-y(1))/(x(2)-x(1))-yp1)
     endif

     do i=2,n-1
          sig=(x(i)-x(i-1))/(x(i+1)-x(i-1)
          p=sig*y2(i-1)+2.
          y2(i)=(sig-1.)/p
          u(i)=(6.*((y(i+1)-y(i))/(x(i+1)-x(i))-(y(i)-y(i-1))/ &
               (x(i)-x(i-1)))/(x(i+1)-x(i-1))-sig*u(i-1))/p
     end do

     if (ypn.gt..99e30) then
          qn=0.
          un=0.
     else
          qn=0.5
          un=(3./(x(n)-x(n-1)))*(ypn-(y(n)-y(n-1))/(x(n)-x(n-1)))
     endif

     y2(n)=(un-qn*u(n-1))/(qn*y2(n-1)+1.)
     do k=n-1,1,-1
          y2(k)=y2(k)*y2(k+1)+u(k)

END
```