ASI3 Mercredi 26 Novembre 2008

Algorithmique et Base de la programmation

Durée : *1h30*
Documents autorisés : **AUCUN**

Remarques :
- Veuillez lire attentivement les questions avant de répondre.
- Le barème donné est un barème indicatif qui pourra évoluer lors de la correction.
- Rendez une copie propre.

1 Lissage de courbe (7 points)

L'objectif de cet exercice est de développer un « filtre non causal », c'est-à-dire une fonction qui lisse un signal en utilisant une fenêtre pour moyenner les valeurs (Cf. figure 1). Pour les premières et dernière valeurs, seules les valeurs dans la fenêtre sont prises en compte.

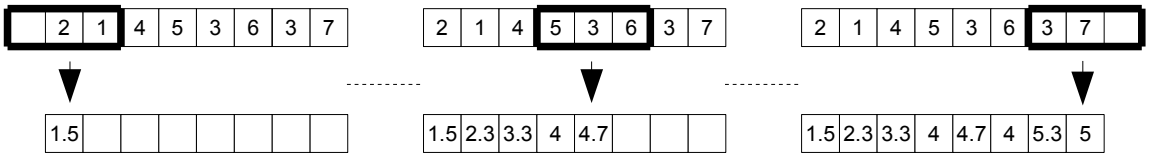

Fig. 1 – Lissage d'un signal avec une fenêtre de taille 3

Soit le type `Signal` :
Type Signal = **Structure**
donnees : **Tableau**[1..MAX] **de Reel**
nbDonnees : **Naturel**
finstructure

Après avoir fait une analyse descendante du problème, proposez l'algorithme de la fonction `filtreNonCausal` avec la signature suivante :
- **fonction** filtreNonCausal (signalNonLisse : Signal, tailleFenetre : **Naturel) :** Signal

  ⌊**précondition(s)** $tailleFenetre \geq 1$
  $impaire(tailleFenetre)$

Solution proposée :
Analyse descencante :

- $filtreNonCausale$: $Signal \times$ **Naturel** $\nrightarrow Signal$
  - min : **Naturel** $\times$ **Naturel** $\rightarrow$ **Naturel**
  - max : **Naturel** $\times$ **Naturel** $\rightarrow$ **Naturel**
  - $moyenne$: $Signal \times$ **Naturel** $\times$ **Naturel** $\nrightarrow$ **Reel**
    - $somme$: $Signal \times$ **Naturel** $\times$ **Naturel** $\nrightarrow$ **Reel**

Algorithmes :

fonction somme (unSignal : Signal, debut, fin : **Naturel) : Reel**

⌊**précondition(s)** $debut \leq fin$
$debut > 0$
$fin \leq unSignal.nbDonnees$

Déclaration resultat : **Reel**
i : **Naturel**

debut
resultat ← 0
pour i ←debut **à** fin **faire**
resultat ← resultat+ unSignal.donnes[i]
finpour
retourner resultat
fin

fonction moyenne (unSignal : Signal, debut, fin : **Naturel) : Reel**

⌊**précondition(s)** $debut \leq fin$
$debut > 0$
$fin \leq unSignal.nbDonnees$

debut
retourner somme(unSignal,debut,fin)/(fin-debut+1)
fin

fonction filtreNonCausal (unSignal : Signal, tailleFenetre : **Naturel) :** Signal

⌊**précondition(s)** $tailleFenetre \geq 1$
$impaire(tailleFenetre)$

Déclaration resultat : Signal
i : **Naturel**

debut
resultat.nbDonnees ← unSignal.nbDonnees
pour i ←1 **à** resultat.nbDonnees **faire**
resultat.donnes[i] ← moyenne(unSignal,max(1,i-tailleFenetre div 2), min(unSignal.nbDonnees ,i+tailleFenetre div 2))
finpour
retourner resultat
fin

2 Manipulation de chaînes de caractères (9 points)[1]

Dans cet exercice on considère que l'on possède les fonctions suivantes :

- **fonction** longueur (uneChaine : **Chaine de caracteres) : Naturel**
  ...avec $longueur("") = 0$
- **fonction** iemeCaractere (uneChaine : **Chaine de caracteres**, iemePlace : **Naturel) : Caractere**
  ⌊**précondition(s)** $0 < iemePlace \leq longueur(uneChaine)$
- **fonction** sousChaine (uneChaine : **Chaine de caracteres**, debutSousChaine : **NaturelNonNul**, longueurSousChaine : **Naturel) : Chaine de caracteres**
  ⌊**précondition(s)** $0 < debutSousChaine \leq longueur(uneChaine)$
  $debutSousChaine + longueurSousChaine \leq longueur(uneChaine) + 1$

2.1 Fonction `estPresent`

Proposez l'algorithme (itératif ou récursif) de la fonction `estPresent` qui permet de savoir si un caractère est présent dans une chaîne. La signature de cette fonction est :

- **fonction** estPresent (ch : **Chaine de caracteres**, c : **Caractere) : Booleen**

Par exemple :

- `estPresent("TRALALA",'A')` retournera *VRAI*
- `estPresent("TRALALA",'C')` retournera *FAUX*

Solution proposée :

fonction estPresent (ch : **Chaine de caracteres**, c : **Caractere) : Booleen**
debut
  si ch="" ou (longueur(ch)=1 et iemeCaractere(ch,1)≠c) **alors**
    retourner FAUX
  sinon
    si iemeCaractere(ch,1)=c **alors**
      retourner VRAI
    sinon
      retourner estPresent(sousChaine(ch,2,longueur(ch-1)),c)
    finsi
  finsi
fin

2.2 Fonction `enleverCaractere`

Proposez l'algorithme récursif de la fonction `enleverCaractere` qui permet de supprimer la première occurence d'un caractère d'une chaîne. La signature de cette fonction est :

- **fonction** enleverCaractere (ch : **Chaine de caracteres**, c : **Caractere) : Chaine de caracteres**

[1] Inspiré d'un exercice de l'examen d'algorithmique de 2005 du Master ICA de l'Université Pierre Mendès France

Par exemple :
- `enleverCaractere("TRALALA",'A')` retournera *"TRLALA"*
- `enleverCaractere("TRALALA",'B')` retournera *"TRALALA"*

Solution proposée :
fonction enleverCaractere (ch : **Chaine de caracteres**, c : **Caractere**) **: Chaine de caracteres**
debut
  cas où longueur(ch) **vaut**
    0:
      retourner ""
    1:
      si iemeCaractere(ch,1)=c **alors**
        retourner ""
      sinon
        retourner ch
      finsi
    autre :
      si iemeCaractere(ch,1)=c **alors**
        retourner sousChaine(ch,2,longueur(ch-1))
      sinon
        retourner sousChaine(ch,1,1)+enleverCaractere(sousChaine(ch,2,longueur(ch-1)),c)

      finsi
  fincas
fin

2.3 Fonction `estConstructible`

Proposez l'algorithme récursif de la fonction `estConstructible` qui permet de savoir si un mot est constructible à partir d'un ensemble de lettres. Le mot et l'ensemble de lettres sont représentés à l'aide d'une chaîne de caractères. Si une lettre apparaît deux fois dans le mot, il faut qu'elle apparaissent au moins deux fois dans l'ensemble de lettres. La signature de cette fonction est :
- **fonction** estConstructible (mot : **Chaine de caracteres**, lettres : **Chaine de caracteres**) **: Booleen**

Par exemple :
- `estConstructible("BONJOUR","BJNORUYZ")` retournera *faux* (le 'O' n'apparaissant qu'une seule fois dans "BJNORUYZ")
- `estConstructible("TRALALA","LLAAAAART")` retournera *vrai*
- `estConstructible("","ABC")` retournera *vrai*

Solution proposée :
fonction estConstructible (mot : **Chaine de caracteres**, lettres : **Caractere**) **: Chaine de caracteres**

```
debut
   si ch="" alors
      retourner VRAI
   sinon
      si estPresent(lettres,iemeCaractere(mot,1)) alors
         si longueur(mot)=1 alors
            retourner VRAI
         sinon
            retourner estConstructible(sousChaine(mot,2,longueur(mot)-1),
            enleverCaractere(lettres,iemeCaractere(mot,1))  )
         finsi
      sinon
         retourner VRAI
      finsi
   finsi
fin
```

3 Un petit dessin (4 points)[2]

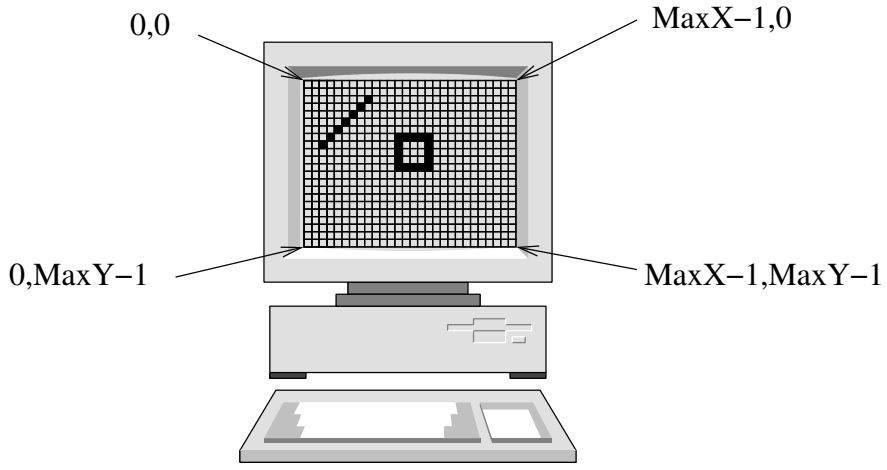

FIG. 2 – Écran graphique

On peut utiliser l'écran d'un ordinateur graphiquement (pour dessiner des points, segments, cercles, etc.) via une variable de type `EcranGraphique`.

On munit le type `EcranGraphique` des procédures :

- **procédure** deplacer **(E/S** e : EcranGraphique **E** x,y : **Naturel)**
  ...qui permet de déplacer (sans rien afficher) le curseur graphique aux coordonnées (x, y) ;
- **procédure** tracerLigneJusquA **(E/S** e : EcranGraphique **E** x,y : **Naturel)**
  ...qui permet de tracer une ligne de la position courante du curseur aux coordonnées (x, y). (x, y) deviennent les nouvelles coordonnées du curseur.

[2]Exercice inspiré d'un programme de M. Hugo ETIEVANT `http ://cyberzoide.developpez.com/info/turbo/abonnes.php3`

Comme l'indique la figure 2, on considère que les coordonnées $(0, 0)$ du type `EcranGraphique` se trouvent en haut à gauche de l'écran. On ne se soucie pas du fait que des coordonnées (x, y) dans les deux procédures précédentes puissent « sortir de l'écran ».

Soit la procédure `p1` suivante :

procédure p1 **(E/S** e : EcranGraphique , **E** x,y,d,n : **Naturel)**
debut
  si n>0 **alors**
    deplacer(e,x,y)
    tracerLigneJusquA(e,x+(d div 2),y)
    tracerLigneJusquA(e,x+d,y-(d div 2))
    tracerLigneJusquA(e,x+(d div 2),y-d)
    tracerLigneJusquA(e,x,y-(d div 2))
    tracerLigneJusquA(e,x+(d div 4),y-(d div 4))
    p1(e,x+(d div 4),y-(d div 4),(d div 2),n-1)
    tracerLigneJusquA(e,x+(d div 2),y)
    tracerLigneJusquA(e,x+d,y)
    tracerLigneJusquA(e,x+d,y-d)
    tracerLigneJusquA(e,x,y-d)
    tracerLigneJusquA(e,x,y)
  finsi
fin

En numérotant chronologiquement les différents segments tracés, dessinez la figure qui sera produite par l'instruction `p1(e,0,0,10,3)`.

Solution proposée :

Un dessin du style :