I3 - Algorithmique

Durée : *3h00*
Documents autorisés : **AUCUN** (calculatrice comprise)

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.
- **N'utilisez pas de crayon à papier**.

1 Question de cours (4 points)

1.1 Compréhension

Nous avons vu en cours que le tri rapide avait l'algorithme suivant :

procédure triRapide **(E/S** t : **Tableau[**1..MAX**] d'Entier** ,**E** nb :**Naturel)**
debut
  triRapideRecursif(t,1,nb)
fin
procédure triRapideRecursif **(E/S** t : **Tableau[**1..MAX**] d'Entier** ,**E** d,f :**Naturel)**
  Déclaration indicePivot : **Naturel**
debut
  si d<f **alors**
    partionner(t,d,f,indicePivot)
    triRapideRecursif(t,d,indicePivot-1)
    triRapideRecursif(t,indicePivot+1,f)
  finsi
fin

La figure donnée en dernière page (à rendre avec votre copie) représente à l'aide de flèches les différents appels de procédures. Numérotez chronologiquement l'ordre de ces appels.

1.2 Complexité

Sachant que la complexité dans le pire et le meilleur des cas de la procédure `partitionner` est respectivement en $O(n)$ et $\Omega(n)$, rappelez et justifiez les complexités dans le pire et dans le meilleur des cas de la procédure `triRapide`.

2 Palindrome (4 points)

Une chaîne de caractères est un palindrome si la lecture de gauche à droite et de droite à gauche est identique. Par exemple "radar", "été", "rotor", etc. sont des palindromes.

On suppose posséder les fonctions suivantes :
- **fonction** longueur (uneChaine : **Chaine de caracteres) : Naturel**
  avec $longueur("") = 0$
- **fonction** iemeCaractere (uneChaine : **Chaine de caracteres**, iemePlace : **Naturel) : Caractere**
  avec $0 < iemePlace \leq longueur(uneChaine)$
- **fonction** sousChaine (uneChaine : **Chaine de caracteres**, debut, longueur : **Naturel) : Chaine de caracteres**
  avec :

- $0 < debut \leq longueur(uneChaine)$,
- $debut + longueur \leq longueur(uneChaine)$,
- $sousChaine(s, d, 0) =$ ""

1. Proposez une fonction **itérative** qui permet de savoir si une chaîne est un palindrome.
2. Proposez une fonction **récursive** qui permet de savoir si une chaîne est un palindrome.

3 Résolution d'une équation par dichotomie (4 points)

Soit le type `FonctionDeRDansR` défini de la façon suivante :
- **Type** FonctionDeRDansR **= fonction**(x : **Reel**) : **Reel**

1. Proposez le corps de la fonction suivante qui résoud de manière **dichotomique** et **récursive** l'équation $f(x) = 0$ sur l'intervalle $[a, b]$ à ϵ près (sachant que la solution est unique et appartient bien à l'intervalle $[a, b]$) :
   - **fonction** resoudre **(**f : FonctionDeRDansR, a,b : **Reel**, epsilon : **Reel) : Reel**
2. Votre algorithme récursif est terminal ou non terminal ? Justifiez.

4 Jeu de la vie (8 points)

Règles

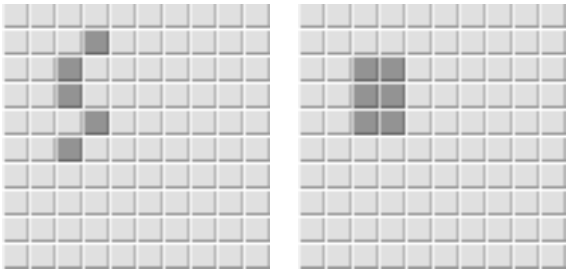
≪ En préambule, il faut préciser que le jeu de la vie n'est pas vraiment un jeu au sens ludique, puisqu'il ne nécessite aucun joueur ; il s'agit d'un automate cellulaire, un modèle où chaque état conduit mécaniquement à l'état suivant à partir de règles pré-établies.

Le jeu se déroule sur une grille à deux dimensions, théoriquement infinie (mais de longueur et de largeur finies et plus ou moins grandes dans la pratique), dont les cases, qu'on appelle des ≪ cellules ≫, par analogie avec les cellules vivantes, peuvent prendre deux états distincts : ≪ vivantes ≫ ou ≪ mortes ≫.

À chaque étape, l'évolution d'une cellule est entièrement déterminée par l'état de ses huit voisines de la façon suivante :
- une cellule morte possédant exactement trois voisines vivantes devient vivante (elle naît) ;
- une cellule vivante possédant deux ou trois voisines vivantes le reste, sinon elle meurt.

≫ [1]

La figure 1 présente un exemple de calcul d'une nouvelle génération (en gris clair les cellules mortes et en gris foncé les cellules vivantes).



(a) ancienne génération (b) nouvelle génération

FIGURE 1 – Calcul d'une nouvelle génération du jeu de la vie

1. Wikipédia

4.1 Analyse

On suppose que l'on possède un type *Grille* avec les fonctions et procédures suivantes :
- **fonction** grille (largeur, hauteur : **Naturel) :** Grille
- **fonction** obtenirLargeur **(**g : Grille**) : Naturel**
- **fonction** obtenirHauteur **(**g : Grille**) : Naturel**
- **fonction** estUneCelluleVivante **(**g : Grille, x,y : **Naturel) : Booleen**
  avec $0 < x \leq obtenirLargeur(g)$ et $0 < y \leq obtenirHauteur(g)$
- **procédure** faireMourirUneCellule **(E/S** g : Grille,**E** x,y : **Naturel)**
  avec $0 < x \leq obtenirLargeur(g)$ et $0 < y \leq obtenirHauteur(g)$
- **procédure** faireNaitreUneCellule **(E/S** g : Grille,**E** x,y : **Naturel)**
  avec $0 < x \leq obtenirLargeur(g)$ et $0 < y \leq obtenirHauteur(g)$

Pour résoudre le problème ***simulerJeuDeLaVie*** qui affiche les n premières générations d'une grille d'une certaine $largeur$ et d'une certaine $hauteur$, initialement remplie aléatoirement d'un certain $pourcentage$ de cellules vivantes, il faut savoir résoudre les sous-problèmes suivants :
- initialiser une grille avec le certain pourcentage de cellules vivantes (sous-programme *initialiser*)
- calculer la prochaine génération d'une grille (sous-programme *nouvelleGeneration*), sachant que pour résoudre ce problème il faut savoir résoudre le sous-problème suivant :
  - savoir si une cellule va naître ou continuer à vivre (sous-programme *vaNaitreOuContinuerAVivre*), sachant que pour résoudre ce problème il faut savoir résoudre le sous-problème suivant :
    - déterminer le nombre de cellules voisines vivantes (sous-programme *nbVoisinsVivants*)
- afficher une grille (sous-programme *afficher*)

Questions

1. Proposez une analyse descendante du (sous-)programme *simulerJeuDeLaVie*.
2. Pour chaque (sous-)programme de votre analyse descendante, déterminez si c'est un (sous-)programme de la logique métier ou de l'interface homme-machine (IHM).
3. Comment pourriez vous n'avoir aucun (sous-)programme gérant les aspects IHM.

4.2 Conception préliminaire

Donnez les signatures des procédures ou fonctions issues de votre analyse descendante.

4.3 Conception détaillée

1. Donnez le corps des procédures ou fonctions associées aux (sous-)programme :
   - *simulerJeuDeLaVie* ;
   - *nouvelleGeneration*.
2. Proposez une conception du type *Grille*.