Controle Continu IF122

Les documents sont interdits. La durée de l'épreuve est de 1h50. Si vraiment vous etes bloqués appelez moi, j'echangerais éventuellement une indication contre une pénalité. Le barème est donné à titre indicatif. Bonne chance.

[Question de cours]

(1.5 points) Définissez les types suivant : `ArbreBinaire, ArbreBinaireDeRecherche, ListeDoublementChainee`.

Objet du Problème : On s'intéresse à la construction d'ensembles d'entiers formant une partition de $\{0..n\}$, par exemple $E_0 = \{0,2,4\}$, $E_1 = \{3,5\}$, $E_2 = \{1\}$ est une partition de $\{0,1,2,3,4,5\}$ en 3 ensembles. On peut aussi considérer que l'on a associé à chaque entier le numero de l'ensemble auquel il appartient, par une application que l'on a qu'à appeler `est_dans`, telle qu'ici on ait : $est\_dans(0) = 0, est\_dans(1) = 2, est\_dans(2) = 0, est\_dans(3) = 1, est\_dans(4) = 0, est\_dans(5) = 1$.

Exercice 1.

(0.5 points) Ecrire un programme `main` qui demande à l'utilisateur une valeur limite pour l'intervalle $\{0..n\}$.

(1 point) Définissez un type `Tableau` de taille variable.

(1.5 points) Ecrire une fonction `init` qui prenne en argument un entier n et retourne un `Tableau` d'entiers indexés par $\{0..n\}$, de sorte qu'à l'indice i corresponde la valeur i.

(1 point) Ecrire une procedure qui affiche ce `Tableau`.

(1 point) Compléter votre programme principal pour enchainer et tester vos fonctions.

Intérêt pour le Problème : Ces fonctions vous permettent de créer la partition la plus fine possible, puisque le `Tableau` retourné par `init` attribue à chaque entier i un ensemble numeroté i. Cet ensemble est dans un premier temps réduit à un seul element. On peut remarquer que puisque une partition quelconque sera toujours plus grossière que celle là, on pourra toujours attribuer des numéros aux ensembles en prenant des valeur dans $\{0..n\}$.

(1 point) Ecrire une fonction `TrouverEnsemble` qui ait pour argument un entier i, et un `Tableau`, qui retourne le numéro de l'ensemble auquel i appartient.

(2 points) Ecrire une procédure `AfficheEnsemble` qui prend en argument un entier i et un `Tableau` d'entier T, et qui affiche les entiers qui sont dans l'ensemble i. (Dans l'exemple du debut si i vaut 1, on affichera $3, 5$.)

(0.5 points) Quelle est la bonne mesure de complexité pour `AfficheEnsemble` (Répondez sous forme d'un commentaire dans votre programme)

(0.5 points) Compléter votre programme principal en verifiant que si $T = init(n)$ alors la sequence `AfficheEnsemble(1,T) ; AfficheEnsemble(3,T)` affiche bien 1 puis 3.

Suivit du problème : On ne peut pas avoir une meilleurs complexité pour `AfficheEnsemble` sans mieux structurer les données, c'est ce que nous allons faire à présent : nous allons représenter les ensembles par une liste chainée de leurs elements.

Exercice 2.

(2 points) Créer un type `Ensemble` qui ait 3 champs, un qui contienne une liste chainée de ses elements, l'autre qui indiquera sa taille, et un troisième qui soit un pointeur vers la dernière cellule de la liste de ses éléments.

(2 points) Ecrivez une procedure `AfficheEnsemble` qui prend en argument un `Ensemble` et qui affiche ses caractéristiques.

(1 point) Créér un type `TableauEnsemble` de taille variable et qui représente un tableau indexés par $\{0..n\}$ d'`Ensemble` (vous vous inspirerez du tableau d'entier à taille variable). L'intention étant que l'ensemble en index i portera le numéro i.

(1 point) Ecrivez une procedure `AfficheTableauEnsemble` qui affiche tout les ensembles d'un `TableauEnsemble`.

(0.5 points) Ecrire une procedure `AfficheEnsemble` qui prenne en argument un entier i et un `TableauEnsemble`, qui affiche les éléments de l'ensemble numéro i.

(0.5 points) Discutez sa complexité par rapport à la première question.

(2 points) Créer une fonction `initEnsemble(int n)` qui retourne un `TableauEnsemble` où l'ensemble numero i de ce tableau ne soit constitué que du seul element i.

(0.5 points) Compléter votre programme principal pour vérifier rapidement que vos fonctions sont correctes.

On va à présent articuler et tester tout ca, avec une question bonus plus difficile :

Exercice 3.

(4 points) Ecrire une procédure `UnionEnsemble(int i, int j, Tableau &T, TableauEnsemble &TE)` qui fasse l'union des ensembles qui contiennent les entiers i et j. J'impose que l'union de deux ensemble de taille différentes prenne pour numéro celui du plus grand (en taille) des deux. L'ensemble correspondant à l'autre numéro se retrouvera donc vide. Pensez à modifier le tableau T pour propager les changements que vous avez effectuez.