Département STPI Jeudi 12 avril 2012

I3 - Algorithmique

Durée : *1h30*
Documents autorisés : **AUCUN** (calculatrice comprise) [1]

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 (3 points)

1. Quelle est la différence entre itération déterministe et itération indéterministe ?
2. Qu'est ce que l'encapsulation ?
3. Quel est le rôle de la conception préliminaire ?

2 Le jeu Sogo (8 points)

≪ Le Sogo est le nom donné en 1978 à la première édition française par la société Ravensburger d'un jeu qui existait depuis 1968 aux États-Unis d'Amérique sous le nom de Score four. Les premières versions du jeu datent de 1968 alors que le jeu Puissance 4 n'est apparu qu'en 1974.

Le Sogo est un jeu d'alignement en trois dimensions, avec un effet de gravité : une pièce ne peut être jouée à un niveau supérieur que si des pièces ont déjà été jouées à tous les niveaux inférieur au même endroit.

Le but du jeu est d'aligner 4 pièces de sa couleur le premier. L'alignement peut être vertical, horizontal ou suivre n'importe quelle diagonale. ≫ (Wikipédia).

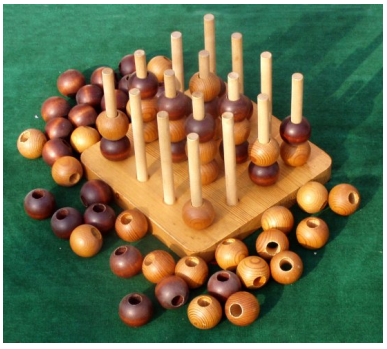

FIGURE 1 – Le jeu Sogo (source Wikipédia)

Comme nous l'avons vu en cours, on peut modéliser ce jeu de la façon suivante :
Constante LARGEUR = 4
Constante PROFONDEUR = 4
Constante HAUTEUR = 4
Type Contenu = {vide,pionNoir,pionBlanc}
Type PlateauSogo = **tableau[***1..LARGEUR* **][** *1..PROFONDEUR***][***1..HAUTEUR***] de** Contenu

1. Proposez la procédure `vider`, qui permet de vider entièrement un plateau du jeu Sogo.
2. Proposez la fonction `hauteur`, qui permet de calculer le nombre de pions verticaux présents sur une colonne (identifiée par son abscisse et son ordonnée).

1. Sauf les dictionnaires pour les étudiants non francophones

3. Proposez la procédure `jouerUnPion`, qui permet de jouer un pion sur une colonne (identifiée par son abscisse et son ordonnée). Cette colonne ne devant pas être totalement remplie.

3 Une analyse descendante (9 points)

Soit l'analyse descendante présentée par la figure 2 qui permet de rechercher la position d'une chaîne de caractères dans une autre chaîne indépendemment de la casse (d'où le suffixe IC à l'opération `positionSousChaineIC`), c'est-à-dire que l'on ne fait pas de distinction entre majuscule et minuscule.

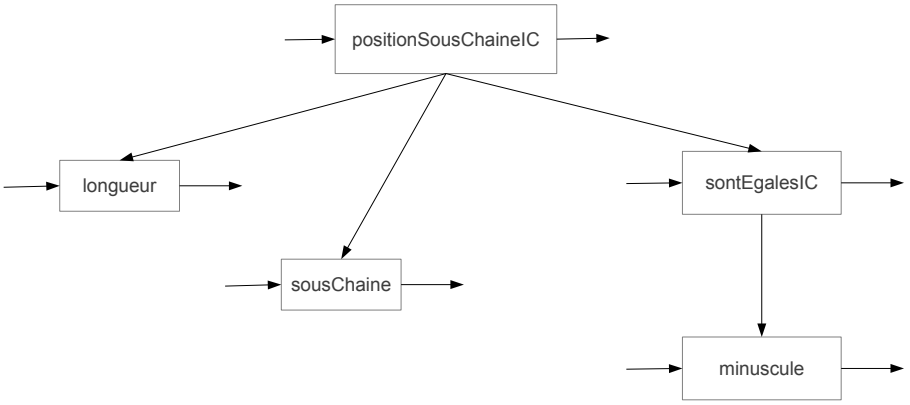

FIGURE 2 – Une analyse descendante

Pour résoudre ce problème il faut pouvoir :
- obtenir la longueur d'une chaîne de caractères ;
- obtenir la sous-chaîne d'une chaîne en précisant l'indice de départ de cette sous-chaîne et sa longueur (le premier caractère d'une sous-chaîne à l'indice 1) ;
- savoir si deux chaînes de caractères sont égales indépendemment de la casse.

L'opération `positionSousChaineIC` retournera la première position de la chaîne recherchée dans la chaîne si cette première est présente, 0 sinon.

Par exemple :
- `positionSousChaine("AbCdEfGh","cDE")` retournera la valeur 3 ;
- `positionSousChaine("AbCdEfGh","abc")` retournera la valeur 1 ;
- `positionSousChaine("AbCdEfGh","xyz")` retournera la valeur 0.

1. Complétez l'analyse descendante en précisant les types de données en entrée et en sortie.
2. Donnez les signatures complètes (avec préconditions si nécessaire) des sous-programmes (fonctions ou procédures) correspondant aux opérations de l'analyse descendante.
3. Donnez l'algorithme du sous-programme correspondant à l'opération `positionSousChaineIC` (vous ne donnerez pas les algorithmes des autres opérations).
4. Donnez le code Pascal de ce sous programme en ayant au préalable déclarer les autres sous-programmes.