G. HUNAULT Angers, 2001

Mini-Projet en Algorithmique et Programmation :

Opérations sur les suites d'entiers

(Licence MASS)

1. Position du problème

Soit $(u_n)_{n \in N}$ une suite d'entiers dont les premières valeurs sont u_0, u_1, u_2...
On définit les opérations monoséquences suivantes

- si (a_n) = DECAG((u_n)) alors $a_n = u_{n-1}$ (pour $n > 1$, et a_0 =0),
- si (b_n) = DECAD((u_n)) alors $b_n = u_{n+1}$,
- si (c_n) = AJOUT((u_n)) alors $c_n = 1 + u_n$,
- si (d_n) = ALTER((u_n)) alors $d_n = (-1)^n u_n$.
- si (e_n) = SERIE((u_n)) alors $e_n = \displaystyle\sum_{k=0}^{k=n} u_k$.
- si (f_n) = BINOD((u_n)) alors $f_n = \displaystyle\sum_{k=0}^{k=n} C_n^k u_k$.
- si (g_n) = BINOI((u_n)) alors $g_n = \displaystyle\sum_{k=0}^{k=n} (-1)^{n-k} C_n^k u_k$.
- si (h_n) = ACONV((u_n)) alors $h_n = \displaystyle\sum_{k=0}^{k=n} u_k u_{n-k}$.
- si (i_n) = ECONV((u_n)) alors $i_n = \displaystyle\sum_{k=0}^{k=n} C_n^k u_k u_{n-k}$.

On définit aussi les opérations biséquences ADDIT, MULTC et MULTH par

- si (j_n) = ADDIT((u_n),(v_n)) alors $j_n = u_n + v_n$,
- si (k_n) = MULTC((u_n),(v_n)) alors $k_n = u_n * v_n$,
- si (l_n) = MULTH((u_n),(v_n)) alors $l_n = \sum_{k=0}^{k=n} u_k * v_{n-k}$.

Le but du miniprojet est de programmer en *Rexx* et en *Apl* ces opérations et de les appliquer à des suites d'entiers définies par des formules simples comme $u_n = n^2$.

On admettra que toutes les suites considérées sont implémentées par des tableaux de $n+1$ éléments, n étant fixé (par exemple $n = 50$) pour lesquels l'indice 0 est valide, de façon à ce que $T[0]$ corresponde à t_0.

2. Travail demandé

2.a) Ecrire les algorithmes complets (en-tête, paramètres, variables locales, variables globales et corps de l'algorithme) pour les douze opérations citées ; n sera une variable globale, il n'y aura qu'un seul paramètre (U) pour les opérations monoséquences et deux paramètres (U et V) pour les opérations biséquences.

Ecrire ensuite un algorithme principal qui affecte à n la valeur 10, qui construit deux tableaux TU et TV respectivement associés aux suites définies par u_n= n et $v_n = n^2$ puis qui affiche toutes les transformées monoséquences de TU et toutes les transformées biséquence de TU et TV.

On séparera bien l'appel des transformations et leur affichage, quitte à inventer des sous-programmes d'affichages ou à remplir un tableau complémentaire de résultats.

2.b) Traduire en *Rexx* tous les algorithmes précédents en un programme complet. Le programme utilisera un et un seul paramètre, à savoir la valeur de n. En cas d'absence de ce paramètre, le programme rappelera sa syntaxe.

2.c) Adapter la traduction des algorithmes en fonctions *Apl*. La fonction principale que l'on nommera OPESUIT n'utilisera aucun paramètre, n étant stocké comme seule variable globale dans la zone de travail (nommée $WORKSPACE$ en *Apl*).

2.d) Comment ont été choisis les noms DECAG, DECAD etc. ?

Ce miniprojet est à rédiger seul(e). Vous avez certainement à effectuer une recherche bibliographique simple, notamment pour la question *2.d* (indiquez vos sources). Vous m'enverrez en attaché par mail un fichier .zip ou un fichier .tar qui contient tout votre travail, à savoir : une zone de travail Apl (fichier .ws = "$Workspace$" ou .apl), un fichier .rex, texte du programme Rexx, ainsi qu'un document .doc au sens de $Word$ pour $Windows$ (éventuellement au format RTF) ou un fichier .tex pour LateX qui contiendra les algorithmes, la documentation et vos commentaires.

Ces fichiers auront pour nom vos initiales : Jomes Band fournira donc JB.WS, JB.REX et JB.DOC le tout dans JB.ZIP ou dans JB.TAR ; par contre Nona Higane rendra NH.WS, NH.REX et NH.DOC, le tout dans NH.ZIP ou dans NH.TAR. Lors de l'archivage, vous ne mettrez aucun sous-répertoire.

Ce travail de recherche, programmation et rédaction est à rendre pour le vendredi 7 décembre 2001 à 17h.

Aucun retard ne sera toléré.

Pour envoyer les fichiers, il faut impérativement utiliser la page Web

`http://www.info.univ-angers.fr/pub/gh/wals/envoiproj.php`

3. Remarques concernant la rédaction

Il vaut mieux rendre un travail incomplet que de ne rien rendre. La qualité des algorithmes, en particulier leur simplicité et leur lisibilité sera appréciée. Indiquez clairement vos choix : si un terme vous parait ambigu, précisez sa définition pour vous. Justifiez vos modes de programmation ; récursivité ou linéarité, appel d'opérateurs, de fonctions prédéfinies ou écriture de boucles explicites. Nommez distinctement vos variables (en particulier, vous utiliserez systématiquement au moins 4 lettres par identificateur). N'hésitez pas à commenter sans dépasser toutefois un commentaire toutes les 4 lignes en moyenne (sauf pour l'en-tête).

Vous n'êtes pas obligé(e) de tout tester dans tous les sens. Effectuez donc des tests élémentaires. L'affichage des résultats à l'écran devra être très propre, très soigné. Ainsi en cas d'affichage numérique multi-lignes, on mettra bien les unités sous les unités, les dizaines sous les dizaines etc. Soignez aussi la rédaction, l'orthographe (utilisez des correcteurs orthographiques et grammaticaux).

La note sera principalement fonction de la qualité de l'analyse du travail à effectuer, de l'exposé de la méthode et de la clarté de la rédaction. La valeur attribuée au programme ne représentera que 25 % de la note ; aucune interface n'est demandée.

Aucun langage autre que cité ne devra être utilisé, aucun traitement de textes autre que cité ne devra être utilisé. Tout fichier qui ne correspondra pas aux initiales de l'élève ne sera pas lu. En particulier, les fichiers nommés minip, algo, stat,projet, doc ne seront pas lus.

Un programme même incomplet mais commenté et bien indenté sera au moins autant apprécié qu'un programme qui "marche" mais sans commentaire et illisible quand aux structures de flot (boucles, tests etc.).

4. Références

Le logiciel *APL* pour *Dos/Windows* version spéciale Université d'Angers peut être récupéré à l'adresse

`http://www.info.univ-angers.fr/pub/gh/wals/getapl.htm`

on y trouvera aussi comment récupérer un *APL* gratuit pour *Linux*.

Le logiciel *REGINA (REXX)* pour *Dos/Windows* et *Linux* peut être récupéré à l'adresse

`http://www.info.univ-angers.fr/pub/gh/wals/getrex.htm`

Plus généralement, tous les documents pour le cours Algorithmique, Langages et Styles de Programmation dont un manuel élémentaire d'*Algorithmiques raisonnées*, les consignes pour le projet et ce texte sont disponibles à l'adresse

`http://www.info.univ-angers.fr/pub/gh/wals/`

Pour être sur(e) que les algorithmes sont correctement écrits, on utilisera le programme galg et on rendra le listage des instructions (fichier `.lst` plutot que l'algorithme initial `.alg`) ainsi que le fichier `.lvm` des variables, tableaux et modules. Le programme galg est disponible à l'adresse

`http://www.info.univ-angers.fr/pub/gh/Galg/`

On pourra profiter de la page

`http://www.research.att.com/~njas/sequences/transforms.html`

pour approfondir les transformations de suites d'entiers.