e3a MP 2004

1 Chercher-remplacer

Voir les listings caml et Maple

2 Racine carée

Voir les listings caml et Maple

3 Nombre de 1

Si n est pair c'est le nombre de 1 de $\frac{n}{2}$, s'il est est impair c'est 1+ le nombre de 1 de $\frac{n}{2}$. Il reste à dire cela en fonction du langage choisi Voir les listings caml et Maple

4 Qu'affichera ce programme ?

On l'exécute "à la main" : x vaut 1, y vaut 1, donc on affiche 1, x devient 2, y devient 11
x vaut 2, y vaut 11, donc on affiche 2, x devient 4, y devient 21
x vaut 4, y vaut 21, donc on affiche 4, x devient 8, y devient 31
x vaut 8, y vaut 31, donc on affiche 8, x devient 16, y devient 41
x vaut 16, y vaut 41, donc on affiche 16, x devient 32, y devient 51
x vaut 32, y vaut 51, donc on affiche 32, x devient 64, y devient 61
x vaut 64, y vaut 61, donc on ne fait plus rien
Finalement on a affiché 1 2 4 8 16 32
Voir les listings caml et Maple

5 Le triangle de Pascal

on déduit chaque terme du précédent par une multiplication et une division
Voir les listings caml et Maple

6 Tri d'une liste à petit ensemble de valeurs

Je n'ai pas réussi à trouver de circonstances dans lesquelles cet algorithme serait meilleur que celui consistant a lire une fois tout le vecteur pour compter les 0,1,2. Si les 0,1,2 sont en fait des clés et qu'il faut avec eux déplacer leurs contenus, il suffit de parcourir de même le vecteur en enregistant les indices des 0,1,2

Cette question étant célèbre sous le nom de "drapeau tricolore" (je l'ai lue associée au nom de Dijkstra, internet dit le contraire), il y a peut être un intérêt à faire ce mini-tri en un seul parcours ... bien qu'il nécessite plus d'échanges

C'est un classique de la programmation en tant que "machine à se tromper". Pour éviter les erreurs les méthodes usuelles sont la programmation fonctionelle (mais l'énoncé exige de l'itératif) et ici l'usage de sentinelles : on met 0,1,2 au début ce qui élimine les cas particuliers, et à la fin on enlève ces 0,1,2, ... ce n'est vraiment pas l'idée de cet énoncé

- Si x=2, "il est en place", il suffit donc de remplacer i par i+1
- Si x=1: on échange t[k+1] et t[i+1], on remplace k par k+1 et i par i+1
- Si x=0 : on échange t[k+1] et t[i+1], puis t[j+1] et t[k+1], on remplace j par j+1, k par k+1 et i par i+1

En début d'algorithme on a j=0, k=0, i=0. Si i=n, on a fini. Sinon, i<n et on lit x=t[i+1]

- Si x=2, "il est en place", il suffit donc de remplacer i par i+1
- Si x=1:
  - si $k+1 \leq i$ on échange t[k+1] et t[i+1], on remplace k par k+1 et i par i+1
  - sinon (on n'a pas encore vu de 2) on remplace k par k+1 et i par i+1
- Si x=0 :
  - si $k+1 \leq i$ et $j+1 \leq k$: on échange t[k+1] et t[i+1], puis t[j+1] et t[k+1], on remplace j par j+1, k par k+1 et i par i+1

- si k=i et j+1 $\leq$ k : on échange t[j+1] et t[i+1] , on remplace j par j+1, k par k+1 et i par i+1
- si k<i et j=k (il y a eu des 2, pas de 1)
  * si j=0, on échange t[j+1] et t[i+1] , on remplace j par j+1, k par k+1 et i par i+1
  * sinon , on échange t[j+1] et t[i+1] , on remplace j par j+1, k par k+1 et i par i+1
- si k=i et j=k (il n'y a eu ni 2, ni 1)
  * si j=0 : j=j+1, k=k+1, i=i+1
  * sinon on remplace j par j+1, k par k+1 et i par i+1

La quasi-absence de modifications dues aux cas particuliers est étonnante

Remarque sur la programmation Maple : avec des listes ça ne marche pas ce qui est 1/2-normal, avec des arrays cela m'a amené à chercher le nombre de termes d'un array ce qui est carrément folklorique

Voir les listings caml et Maple

7 Génération de nombres pseudo-aléatoires

L'énoncé disait bien (à la fin, dommage) de ne pas se préoccuper des dépassements de valeurs, le programme caml joint est écrit avec des num et non des int

Il y a une erreur dans l'énoncé : le "r" doit être x (sinon ce n'est pas la peine d'éxécuter a+1 fois la même fonction sur la même donnée)

Voir les listings caml et Maple

8 Programmation d'expressions logiques

Dans le cas d'une expression logique commençant par "quelquesoit" on va faire un ET des valeurs logiques des différents termes de la formule (en partant de vrai)

Dans le cas d'une expression logique commençant par "il existe" on va faire un OU des valeurs logiques des différents termes de la formule (en partant de faux)

Voir les listings caml et Maple