FEUILLE D'EXERCICES N°17 DE L'OPTION D'INFORMATIQUE.

Coupes dans un tableau

Dans tout l'énoncé, n est un entier donné avec $n \geqslant 1$ et $B = (b_0, b_1, \ldots, b_{n-1})$ est un n-uplet d'entiers relatifs. On dispose de la fonction Caml `min` de type `'a -> 'a -> 'a = <fun>` telle que `min p q` calcule le minimum des deux entiers relatifs p et q.

Une coupe C de B est une suite non vide d'éléments consécutifs de B, c'est-à-dire C est de la forme $(b_i, b_{i+1}, \ldots, b_j)$ avec $1 \leqslant i \leqslant j \leqslant n$. La longueur d'une telle coupe est $j - i + 1$. On note $C = B[i..j]$et E l'ensemble des coupes de B.

À toute coupe $C = B[i..j]$ on associe la somme $s(C) = b_i + b_{i+1} + \ldots + b_j$ de ses éléments. Une coupe C de B est dite minimale si la somme est minimale : $s(C) = \min\{s(X), X \in E\}$.

1. Écrire une fonction `SommeCoupe` de type `int vect -> int -> int -> int = <fun>` telle que `SommeCoupe B i j` retourne la somme $s(C)$ de la coupe de $B[i..j]$.

2. Utiliser la fonction précédente pour écrire une fonction `CoupeMinimale` de type `int vect -> int = <fun>` qui calcule la somme d'une coupe minimale de B. Déterminer, en fonction de n, le nombre d'additions et le nombres d'appels à la fonction `min` qu'effectue cette fonction.

3. Soit i donné, $1 \leqslant i \leqslant n$. Écrire une fonction `SousCoupeMinimale` de type `int vect -> int -> int = <fun>` telle que `SousCoupeMinimale B i` calcule la valeur minimale de la somme d'une coupe de B dont le premier terme est b_i, c'est-à-dire $\min\{s(B[i..j]),\ i \leqslant j \leqslant n\}$. Utiliser la fonction `SousCoupeMinimale` pour écrire une fonction `CoupeMinimaleBis` qui calculer la somme d'une coupe minimale de B. Déterminer, en fonction de n, le nombre d'additions et le nombres d'appels à la fonction `min` qu'effectue cette fonction.

4. Que calcule la fonction suivante :

```
let f B =
let c = ref B.(0) and n = vect_length B in
let s = ref !c in
for k =1 to n-1 do
    c := min (!c+B.(k)) (B.(k));
    s := min !s !c;
done;
!s;;
```

Justifier votre réponse. Que pensez-vous de ce programme?

5. Soit q un relatif donné. Écrire une fonction `SommeProche` de type `int vect -> int -> int = <fun>` telle que `SommeProche B q` calcule la somme d'une coupe de B la plus proche de q, c'est-à-dire $s(C)$ telle que :

$$|s(C) - q| = \min\{|s(X) - q|\,,\ X \in E\}$$

Proposer trois solutions correspondant aux solutions des questions Q2 à Q4 et, pour chacune d'elles, déterminer, en fonction de n, le nombres d'additions et le nombre de comparaisons qu'effectue votre fonction.

6. Une coupe $C = B[i..j]$ est un *plateau* si $b_i = b_{i+1} = \ldots = b_j$. La longueur d'un tel plateau est $j - i + 1$. Écrire une fonction `plateau` de type `int vect -> int * int = <fun>` tel que `plateau B` retourne le couple d'indices (i_O, j_0) tels que $C = B[i_0..j_0]$ soit le plateau de longueur maximale de B.

7. Une coupe $C = B[i..j]$ est une *monotonie* si $b_i < b_{i+1} < \ldots < b_j$. La longueur d'une telle monotonie est $j - i + 1$. Écrire une fonction `monotonie` de type `int vect -> int * int = <fun>` tel que `monotonie B` retourne le couple d'indices (i_0, j_0) tels que $C = B[i_0..j_0]$ soit la monotonie de longueur maximale de B.

8. À toute coupe $C = B[i..j]$ de B, on associe $\mu(C)$, égal au nombre d'indices k compris entre 0 et $n - j$, tels que $C = B[i + k..j + k]$. On a donc $\mu(C) \geq 1$ puisque $k = 0$ convient. Déterminer par ses indices i et j une coupe qui maximise $\mu(C)$, puis une coupe qui maximise le produit $\mu(C)\ell(C)$ où $\ell(C)$ est la longueur de C, puis une coupe C telle que $\mu(C) = 1$ et $\ell(C)$ soit maximal.