TP MP2I : Rencontre au milieu

Présentation

Dans ce TP on implante l'algorithme de « rencontre au milieu » vu en cours.
On donne dans l'archive **meet.zip** un fichier **meet_skeleton.c** qui contient notamment :

```
typedef long long LL;//long long au moins 64 bits de long
void triFusion(int i, int j, LL tab[], LL tmp[]);//tri fusion
void affiche(LL tab[], int size);//affichage d'un tableau
```

plus des procédures de tests.

La fonction `triFusion` implante le tri fusion d'un tableau `tab` d'entiers entre les indices `i,j`. Le tableau `temp` est un tableau auxiliaire qui sert à la fusion des sous-parties triées de `tab`.

Q.1 Écrire

```
int dichot(LL T[], LL x, int n);
```

Cette fonction effectue la recherche dichotomique du plus grand indice i de T tel que $T_i \leq x$ ou -1 si $x < \min(T)$.
Le tableau `T` est supposé trié dans l'ordre croissant.

Le projet utilise les fonctions outils présentées plus haut. On crée deux tableau globaux :

```
LL X[2000005],Y[2000005];
```

Ces tableaux contiendront les sommes des sous-ensembles constitués respectivement avec la première partie puis la seconde de l'ensemble de nombres étudié.

Les tests dans les `bench` se font avec

```
LL a[] = {3, 34, 4, 12, 5, 2};
```

Q.2 Ecrire la procédure

```
void tabsommes(LL a[], LL x[], int n, int c)

```

Cette procédure met dans le tableau `x` toutes les sommes calculées avec les éléments de `a`

- de l'indice 0 à l'indice $n-1$ si $c = 0$;
- de l'indice n à l'indice $c+n-1$ si $c \neq 0$.

Indication.
- Utiliser l'opérateur bitwise << pour calculer des puissances de deux.
- Le codage binaire d'un nombre $i \leq N$ écrit sur N bits représente de façon bijective un sous-ensemble de $[\![0, N-1]\!]$.
- Utiliser l'opérateur bitwise & et le précédent pour obtenir la valeur du bit j de l'entier i.

Pour faire le calcul des sommes des différents sous-ensemble de la première moitié des éléments de `a` : tabsommes(a, X, n/2, 0). Et pour la seconde moitié tabsommes(a, Y, n−n/2,n/2).

Exemple d'appel à la fonction de tests `void testtabsomme()` : :

```
--------- testtabsomme ----------
a=[3, 34, 4, 12, 5, 2]
Sommes 1er moitié X=[0, 3, 34, 37, 4, 7, 38, 41]
Sommes 2nd moitié Y=[0, 12, 5, 17, 2, 14, 7, 19]
--------- FIN testtabsomme ----------
```

Q.3 Ecrire la fonction

```
LL meetInTheMiddle(LL a[], int n, LL S)
```

qui implante l'algorithme de rencontre au milieu.
Exemple d'appel à la fonction de tests fournie :

```
a=[3, 34, 4, 12, 5, 2]
partie triée Y : [0, 12, 5, 17, 2, 14, 7, 19]
Plus grande valeur inférieure ou égale à 10 :  10
```