Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

CONCOURS DE CONTRÔLEUR DE LA CONCURRENCE, DE LA CONSOMMATION ET DE LA RÉPRESSION DES FRAUDES DES 6 ET 7 MARS 2017

Concours externe à dominante économique et juridique

Concours externe à dominante scientifique et technologique

ÉPREUVE N° 2 : **option D**

durée : 5 heures - coefficient 4

Traitement automatisé de l'information : établissement d'un algorithme (ordinogramme) et écriture du programme dans l'un des langages choisis dans la liste ministérielle : Cobol, Java, PHP, VB/ASP.NET.

Le sujet comporte 4 pages et se compose de *six groupes* de questions qui devront toutes être traitées.

L'UTILISATION D'UNE CALCULATRICE EST AUTORISEE

MINISTÈRE DE L'ÉCONOMIE
DE L'INDUSTRIE ET DU NUMÉRIQUE

Au sein d'une équipe de programmeur vous êtes chargé de réaliser plusieurs modules nécessaires pour des applications. Vous devez donc traiter l'ensemble des questions de A à F ci-après :

A. Le point le plus proche

Plaçons nous par exemple dans le contexte du développement d'une couche logicielle pour un écran tactile multipoint (smartphone, table interactive, touchsmart, . . .).

Pour implémenter le double clic, nous avons besoin de déterminer le point le plus proche par rapport à un ensemble de points (i.e. l'ensemble des points détectés dans la frame précédente). Remarquons que ce type de calcul pourrait aussi bien aider à déterminer la direction à prendre pour un robot devant passer par ensemble de cibles, ou encore pour un livreur de colis.

A.1 Ecrire une fonction qui détermine quel point est le plus proche d'un point P parmi un ensemble de N points.

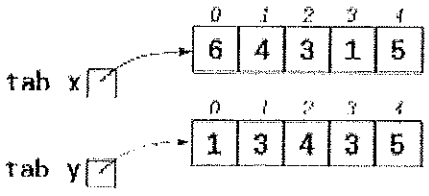
Le point P sera représenté par ses coordonnées entières xP et yP.
L'ensemble de points sera représenté par deux tableaux d'entiers x et y, chacun de taille N, stockant respectivement les abscisses et les ordonnées des points. La fonction retournera l'indice du point le plus proche.

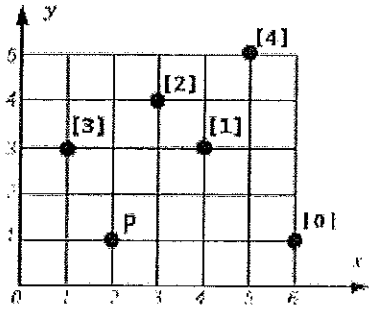
Le prototype de la fonction sera le suivant :

```
int le_plus_proche (int xP, int yP, int *x, int *y, int N) ;
```

Par exemple, soient les deux tableaux tab x et tab y de taille 5 représentant les coordonnées d'un ensemble de points.

Soit le point P de coordonnées (2, 1).





L'appel de la fonction le plus proche (1, 2, tab x, tab y, 5) devra alors retourner 3, c'est à dire l'indice du point de coordonnées (1,3).

A.2 Pour réaliser cet exercice, vous utiliserez et écrirez aussi une fonction qui calcule la distance euclidienne entre deux points.

Son prototype sera le suivant :

```
double distance (int xA, int yA, int xB, int yB) ;
```

B. Chiffres, lettres et caractères de ponctuation

B.1 Ecrire une fonction qui compte le nombre de chiffres, de lettres et de caractères de ponctuation, dans la chaîne de caractères qui lui est passée en paramètre.

Après appel de la fonction, les 3 valeurs comptées seront récupérables dans les variables entières dont l'adresse aura été donnée en argument de la fonction.

Le prototype de la fonction sera donc le suivant :

```
void compter (const char *s, int *cpt_chiffre, int *cpt_lettre, int *cpt_ponctuation);
```

Aide :
Voici quelques fonctions de ctype.h et string.h
_ La fonction int isalpha(char c) retourne vrai si c, est un caractère alphabétique, faux sinon.
_ La fonction int isdigit(char c) retourne vrai si c, est un chiffre, faux sinon.
_ La fonction int ispunct(char c) retourne vrai si c, est un caractère de ponctuation, faux sinon.
*_ La fonction int strlen(const char *s) retourne la longueur de la chaîne passée en paramètre.*

B.2 Ecrire une fonction main() qui appelle la fonction compter() de la question précédente sur la chaîne de caractère suivante : « Vive la programmation ! ! ! Et 1, et 2, et 3... zéro ! ». Ensuite le programme affichera le nombre de chiffres, de lettres et de caractères de ponctuation de cette chaîne.

C. Consonnes et voyelles

C.1 Ecrire une fonction qui retourne vrai si le caractère passée en paramètre est une voyelle. Le prototype de la fonction sera le suivant :

```
char voyelle (char c);
```

C.2 Ecrire une fonction qui compte le nombre de consonnes, de voyelles et de caractères autres, dans la chaîne de caractères qui lui est passée en paramètre. Après appel de la fonction, les 3 valeurs comptées seront récupérables dans les variables entières dont l'adresse aura été donnée en argument de la fonction.

Le prototype de la fonction sera donc le suivant :

```
void compter_lettres (const char *s, int *cpt_voyelle, int *cpt_consonne, int *cpt_autres);
```

Aide :
La fonction int isalpha(char c) retourne vrai si c'est un caractère alphabétique, faux sinon.
*La fonction int strlen(const char *s) retourne la longueur de la chaîne passée en paramètre.*

D. Nombres impairs

Ecrire un programme dans lequel l'utilisateur donne n au clavier, et qui affiche les n premiers nombres impairs.

Par exemple, si n vaut 7 le programme affichera : 1 3 5 7 9 11 13

E. Positiver les tableaux

Ecrire une fonction qui remplace par son opposé chaque nombre négatif du tableau d'entiers, de taille N, donné en paramètre.

Le prototype de la fonction sera le suivant :

```
void positiver (int *a, int N);
```

F. TRACABILITE DE L'HUILE D'OLIVE

La traçabilité des produits alimentaires est aujourd'hui une exigence majeure pour l'exportation de tous les produits agroalimentaires et constitue de ce fait un enjeu économique important pour ce secteur.
La mise en place de la traçabilité à l'huilerie Mirbeau s'appuie sur l'adaptation du système d'information actuel permettant d'assurer le suivi de l'huile produite depuis la parcelle d'un verger jusqu'à l'expédition du produit fini, grâce à l'identification des différents lots à chaque phase de la production.

Il s'agit notamment de :
- la **récolte des olives** (lots de récolte),
- le **transport des olives vers le moulin et son stockage** (lots d'approvisionnement),
- la **trituration** au moulin (lots de trituration),
- le **stockage de l'huile** (lot de production).

On s'intéresse d'abord à la base de données existante.

Gestion de la récolte des olives
Un lot de récolte regroupe les olives d'une même parcelle récoltées le même jour. Le producteur appartient à un groupement en fonction de son origine géographique. La date, le mode et les conditions de récolte et de stockage dans l'exploitation sont essentielles pour justifier de la qualité de l'huile obtenue. Tous les traitements phytosanitaires effectués sur les parcelles sont aussi mémorisés. Un extrait du schéma relationnel de la base de données prenant en compte ces informations est fourni ci-dessous.

Ecrire les traitements ou ordre SQL suivantes en vous basant sur le schéma relationnel fourni :

7a. Liste des noms des producteurs du groupement nommé « BioOliCentre ».
7b. Quantité d'olives récoltée par producteur (code du producteur, quantité).
7c. Liste des codes des parcelles n'ayant subi aucun traitement phytosanitaire depuis le 1er janvier 2011.

Extrait du schéma relationnel de la base de données du moulin
OrigineGéographique (code, nom)
code : clé primaire
Groupement (code, nom, adresse, codeOrigineGéographique)
code : clé primaire
codeOrigineGeographique : clé étrangère en référence à code de OrigineGéographique
Producteur (code, nom, prénom, adresse, codeGroupement)
code : clé primaire
codeGroupement : clé étrangère en référence à code de Groupement
Parcelle (code, superficie, nombreDePieds, variété, responsablePlantation, codeProducteur)
code : clé primaire
codeProducteur : clé étrangère en référence à code de Producteur
Récolte (noLot, date, qtéRécoltée, mode, conditionDeRécolte,
conditionStockageDansExploitation, codeParcelle)
noLot : clé primaire
codeParcelle : clé étrangère en référence à code de Parcelle
TraitementPhytosanitaire (code, produitUtilisé, date, codeParcelle)
code : clé primaire
codeParcelle : clé étrangère en référence à code de Parcelle