TD 2 – Manipulation de Fichiers

Lors de l'exécution d'un programme, les données saisies par l'utilisateur ainsi que les calculs/opérations réalisés dessus sont stockés dans des variables. À la fin de nos programmes, les variables sont automatiquement détruites et la mémoire libérée. Afin de pouvoir conserver les données saisies par l'utilisateur et/ou des calculs que l'on aurait effectués, il est nécessaire de les sauvegarder en mémoire (dite ''principale''). Ceci s'effectue à l'aide de fichiers.

1. Comptes bancaires

1. Création de fichier

- Définir sous forme de structure le type **compte** composé de :
  - un **entier numero_de_compte**,
  - une **chaine de 20 caractères nom_titulaire**,
  - une **chaine de 100 caractères adresse_titulaire**,
  - un **réel solde**.
- Écrire la fonction **saisirCompte** qui permet à l'utilisateur de créer un compte en saisissant au clavier les informations, puis qui retourne ce compte (aucun paramètre en entrée). Vérifier pendant la saisie que les informations sont cohérentes (par exemple que le champ ''nom du titulaire'' est non- vide, que le ''numéro du compte'' n'est pas un nombre négatif, ...).
- Dans le programme principal, créer un fichier *comptes.dat* et ouvrez le en ''ajout''.

2. Manipulation de fichier

- Écrire la procédure **ajouterCompte** qui permet à l'utilisateur de saisir un compte et de le sauvegarder à la fin d'un fichier appelé *comptes.dat* .
- Écrire la fonction **afficherCompte** qui lit le contenu du fichier *comptes.dat* puis l'affiche à l'écran, et renvoie le nombre de comptes affichés.
- Écrire la procédure **trierCompte** qui transfère le contenu du fichier *comptes.dat* dans un tableau de comptes et qui trie ce tableau dans l'ordre croissant des soldes. Puis, on affiche le tableau.
- Écrire la procédure **majCompte** qui prend en entrée un numéro de compte afin de **m**ettre **à j**our les informations (solde et/ou adresse) correspondant au compte dans le fichier *comptes.dat*.
- Écrire le programme principal qui permettra à l'utilisateur de choisir entre les opérations ci-dessus ou quitter le programme. L'ouverture et la fermeture du fichier seront faites dans le programme principal.

2. Relevés météo

On considère un fichier *releves.dat* contenant des relevés météo sous la forme suivante :

NOM_VILLE HEURE_1 TEMPERATURE_1 HEURE_2 TEMPERATURE_2

Exemple de contenu : Limoges 6 -1,5 18 12,5 Reykjavik 4 -15 16 3 …

NOM_VILLE est une **chaine de 20 caractères**, HEURE est un **entier** et TEMPERATURE est un **réel**.

- Définir le type **releves_meteo**
- Écrire une procédure permettant d'afficher à l'écran les informations lues dans le fichier sous la forme (utilisez le caractère de tabulation ''\t'') :

  Limoges :
  - 6h : -1,5 degrés
  - 18h : 12,5 degrés
- Afficher également la température moyenne calculée sur la base du premier relevé météo de chaque ville.
- Écrire une procédure pour saisir le nom d'une ville (Limoges par exemple) et son code postal international (F87000 par exemple) afin de remplacer dans le fichier le nom saisi (s'il existe) par ce code.

3. Fichiers texte

On suppose que dans les exercices de cette partie on passe en argument des fonctions un pointeur sur un fichier contenant déjà des données ; l'ouverture et la fermeture du fichier seront faites dans le programme principal .

- Écrire la procédure **afficherFichier** qui affiche le contenu du fichier à l'écran.
- Écrire la procédure **copierFichier** qui copie les données vers un autre fichier dont le nom est saisi au clavier par l'utilisateur.
- Écrire la procédure **analyserFichier** qui affiche un rapport d'analyse contenant :
  - le nombre total de caractères dans le fichier ;
  - le nombre de caractères représentant des chiffres ('0', '1', etc.) ;
  - le nombre de voyelles ;
  - le nombre de lignes (une ligne est terminée par le caractère de contrôle EOL)
- Modifier la procédure d'affichage pour que chaque ligne affichée contienne au plus 10 caractères.
- Modifier la procédure d'analyse pour afficher également :
  - le nombre de mots – on considère qu'un mot est une suite de caractères ne contenant ni ponctuation ni espace. Un espace peut être une suite de blancs, de tabulations (caractère de contrôle TAB) ou fins de ligne ;
  - le nombre de palindromes (mots qui peuvent se lire dans les deux sens).
- Écrire le programme principal qui doit permettre à l'utilisateur de choisir entre ces différentes opérations.