Projet : données libres

Note 2015/11/19 : l'énoncé des trois premiers exercices est complet ; vous avez de quoi bien commencer le projet d'ici le TP de la semaine prochaine. L'énoncé des exercices suivants sera raffinée dans les jours qui viennent, mais vous pouvez commencer à travailler dessus dès a présent.

1. Prélude

Ce projet est volontairement long afin que chacun puisse s'exprimer en fonction de ses compétences, de son éventuelle expérience préalable et de ses goûts. À vous de choisir un sous-ensemble des questions adapté. Il a été conçu pour qu'un étudiant sans expérience de programmation préalable mais ayant suivi le module avec assiduité puisse facilement avoir au minimum 12. D'autre part, l'expérience que vous accumulerez au fur des premières questions vous fera paraître les questions ultérieures plus simples. Et vous aurez un vrai sentiment d'accomplissement en progressant dans le sujet.

Accrochez vous, et demandez de l'aide aux autres étudiants (sans copier leur code bien évidemment, cela n'aurait aucun intérêt) ainsi qu'à vos chargés de TD.

Table des matières

1. Prélude 1
2. Introduction 1
3. Pour aller plus loin 4
4. Références 4

2. Introduction

De plus en plus d'organisations, gouvernementales ou autres, mettent leurs données à la disposition du public, afin que celui-ci puisse se les approprier, en trouver des utilisations nouvelles et créatives, et aussi contrôler le fonctionnement de ces organisations sur la base de faits concrets.

Par exemple tout un chacun peut maintenant, grâce aux données mises en ligne récemment par la RATP, implanter une application adaptée à ses besoins spécifiques ; par exemple qui déclenche le réveil matin 25 minutes avant l'arrivée du RER, que celui ci soit à l'heure ou pas ; ou d'un RER précédent en cas de gros retard annoncé sur la ligne. D'autres données incluent le cadastre ou le processus de rédaction des lois à l'assemblée nationale, etc. Le citoyen peut ainsi consulter les amendements aux projets de lois auquel son député a participé, et voir si ceux-ci sont conforme aux engagements de campagne de ce dernier.

Comme les données peuvent être de grande taille, en extraire des informations intéressantes requiert la plupart du temps des traitements automatisés. Ce projet est l'occasion de constater que vous avez dès à présent les moyens de commencer à traiter automatiquement ce type de données ; ou d'autres ! Par exemple des résultats d'expériences de physique.

À noter : en d'autres occasions, la bonne approche serait d'utiliser des bibliothèques existantes d'analyse de données et d'apprentissage automatique ; on peut par exemple penser à pandas et scikit-learn. Pour des traitements simples sur de petites données, un tableur suffirait.

Ici nous n'utiliserons que la bibliothèque standard de C++ (et éventuellement la SDL pour les graphiques), quitte à réimplanter des outils de base ; en effet l'objectif est aussi de mettre en pratique ce que vous avez appris dans le cours Info 111 et de mieux comprendre comment fonctionnent ces outils.

Exercice 1 (De l'eau, de l'eau)**.**
Dans ce premier exercice, nous ferons des statistiques simples sur un jeu de données contenant les volumes d'eau distribués mensuellement à Paris en 2011.

`http://opendata.paris.fr/explore/dataset/volumes_d_eau_distribues/?tab=table`

Pour simplifier, nous vous fournissons dans l'archive un fichier donnees/volumes_d_eau_distribues.txt contenant ces données dans un format texte simple : chaque ligne contient un mois et le volume d'eau distribuée ce mois-ci.

(1) Télécharger et extraire l'archive Projet−DonneesLibres.zip.

(2) Écrire un programme eau−total.cpp qui calcule et affiche le volume total d'eau distribué à Paris en 2011.

(3) Écrire un programme eau−moyen.cpp qui calcule et affiche le volume d'eau moyen distribué par mois en 2011.

(4) Écrire un programme eau−max.cpp qui calcule et affiche le mois où le plus grand volume d'eau a été distribué en 2011, ainsi que ce volume d'eau.

Exercice 2 (Déchets)**.**
Les tonnages de déchets des bacs jaunes récoltés chaque mois de 2011 dans chaque arrondissements de Paris sont disponibles sur :

`http://opendata.paris.fr/explore/dataset/tonnages_des_dechets_bacs_jaunes/?tab=table`

Malheureusement les totaux annuels n'ont pas été inclus comme c'est le cas pour :

`http://opendata.paris.fr/explore/dataset/tonnages_des_dechets_bacs_verts/?tab=table`

Écrire un programme dechets.cpp qui utilise le fichier simplifié donnees/tonnages_des_dechets_bacs_jaunes.txt de l'archive pour calculer le tonnage annuel de déchets des bacs jaunes par arrondissement. Le programme affichera l'arrondissement qui produit le plus de déchets des bacs jaunes par année, ainsi que le tonnage annuel de déchets des bacs jaunes produit par cet arrondissement.

Exercice 3 (Vers une bibliothèque générique)**.**
Comparez les programmes écrits pour les deux exercices précédents. Vous constaterez des duplications. L'objectif de cet exercice est d'en extraire, si vous ne l'avez pas déjà fait, une mini-bibliothèque de fonctions réutilisables, afin d'éviter toute duplication et de pouvoir traiter simplement tout nouveau jeu de données dans un format similaire.

(1) Écrire un programme eau−total2.cpp se comportant comme eau−total.cpp mais en implantant et en utilisant les fonctions `litTableauInt`, `colonne` et `total`, documentées ci-dessous :

```
/** Construction d'un tableau 2D d'entiers lu depuis un fichier
 * @param fichier le nom d'un fichier contenant un nombre fixe
  * d'entiers par lignes, séparés par des espaces
 * @param nb_colonnes le nombre de colonnes du fichier
 * @return un tableau d'entiers à deux dimensions
 **/
vector<vector<int>> litTableauInt(string fichier, int nb_colonnes);
```

```
/** Extraction d'une colonne d'un tableau d'entiers
 * @param t un tableau 2D d'entiers
```

```
 * @param i un numéro de colonne
 * @return la colonne i, représentée par un vecteur d'entiers
 **/
vector<int> colonne(vector<vector<int>> t, int i);
```

```
/** Total d'un tableau
 * @param un tableau d'entiers
 * @result la somme de ces entiers
 **/
int total(vector<int> t);
```

(2) Réimplanter de même eau−moyen.cpp avec aussi la fonction suivante :

```
/** Moyenne d'un tableau
 * @param un tableau d'entiers
 * @result la moyenne de ces entiers
 **/
int moyenne(vector<int> t);
```

(3) Réimplanter de même eau−max.cpp avec aussi les fonctions suivantes :

```
/** Max d'un tableau
 * @param un tableau d'entiers
 * @result le plus grand de ces entiers;
    comportement non défini si le tableau est vide
 **/
int max(vector<int> t);
```

```
/** Position du max d'un tableau
 * @param un tableau d'entiers
 * @result la position du plus grand de ces entiers;
    le premier s'il y en a plusieurs;
    -1 si le tableau est vide
 **/
int maxPosition(vector<int> t);
```

Pour éviter la duplication de code, nous allons maintenant regrouper toutes les fonctions dans une bibliothèque tableau−donnees et utiliser la compilation séparée.

(3) Compléter le fichier tableau−donnees.cpp. Voir le fichier tableau−donnees.h pour la documentation.

(4) Compléter les tests dans tableau−donnees−test.cpp.

(5) Créer un projet Code : :Blocks contenant les fichiers tableau−donnees.cpp, tableau−donnees.h, et tableau−donnees−test.cpp. Lancer les tests et déboguer si nécessaire.

(6) Écrire un programme eau−total3.cpp se comportant comme eau−total2.cpp mais en utilisant les fonctions de la bibliothèque. Créer un projet Code : :Blocks pour l'exécuter.

(7) Faire de même pour tous les autres programmes des exercices 1 et 2.

Exercice 4 (Un premier fichier csv)**.**
Même exercice que le 1, mais en partant du fichier csv (après suppression manuelle des espaces dans la deuxième colonne).

Indications : utiliser la fonction getline.

TODO pour le sujet :

– Description du format csv : une ligne d'entête, ...
– Documentation de la fonction getline

Exercice 5 (Généralisation)**.**
Extraction d'un début de bibliothèque générique pour les fichiers csv :

(1) Lecture d'un tableau 2D de chaînes de caractères depuis un fichier ;
(2) Extraction d'une colonne ;
(3) Conversion d'une colonne en colonne numérique (int, float, ...) en utilisant les string stream ; version avancée : éviter la duplication avec une unique fonction de conversion avec un template.

Exercice 6 (Applications)**.**
(1) Quel est l'arbre le plus haut de Paris ? Le plus vieux ?
`http://opendata.paris.fr/explore/dataset/arbresremarquablesparis2011/`
(2) Calculer le tonnage annuel de déchets des bacs jaunes par arrondissement en partant du fichier csv d'origine
`http://opendata.paris.fr/explore/dataset/tonnages_des_dechets_bacs_jaunes/?tab=table`
(3) Quel arrondissement produit le plus de déchets ?
(4) Quelles sont les deux stations de métro / vlib / ... les plus proches l'une de l'autre à Paris.
`https://www.data.gouv.fr/fr/datasets/velib-paris-et-communes-limitrophes-idf/`

Exercice 7.

(1) En utilisant la bibliothèque graphique du TP 8, dessiner la carte de toutes les communes de France à partir du fichier csv fourni sur :
`https://www.data.gouv.fr/fr/datasets/decoupage-administratif-communal-francais`
On pourra par exemple représenter chaque commune par un cercle de centré sur ses coordonnées géographiques et de rayon proportionnel à la population.
(2) Positionner sur cette carte d'autres éléments intéressants.
(3) Positionner sur cette carte plusieurs barycentres de la france (population / surface / ...)

Exercice 8.
Quelle est la voiture commercialisée en France qui consomme le plus ? Le moins ?
Quelle marque, en moyenne sur toute la gamme commercialisée en France, produit les véhicules qui consomment le plus ? le moins ?
`https://www.data.gouv.fr/fr/datasets/emissions-de-co2-et-de-polluants-des-vehicules-`

3. Pour aller plus loin

Exercice ♣ 9.
Lister les stations de métros à Paris où ont été retrouvés des lecteurs MP3.

Exercice ♣ 10.
Implanter l'application mentionnée dans l'introduction (RéveilRER) sous forme d'une application Android.

Exercice ♣ 11.
À vous de trouver des applications intéressantes des données ouvertes !

4. Références

– `http://opendata.paris.fr`

– http://data.gouv.fr
– http://www.ideeslibres.org/