Etude statistique1D.

Sans donner un cours de statistique, nous allons programmer les méthodes de base qui permettent l'étude statistique d'un problème.

Quelques rappels semblent nécessaires.

On définit :
Une **population** : il s'agit d'un ensemble d'individus (ou objets) ayant un point commun.

Un **échantillon** : c'est un sous ensemble de la population étudiée.

Exemple : la population étant l'ensemble des habitants de Liège. Un échantillon étant les étudiants masculins de l'inpres.

Une **variable** est la caractéristique sur laquelle s'effectue l'étude.

Cette variable peut être **qualitative** si elle exprime une qualité, c'est-à-dire qu'elle ne se mesure pas.

Elle peut être **quantitative** si elle peut être mesurée.

Exemple : une personne est mariée ou non.

Elle est âgée de moins de 20 ans.

Une variable quantitative peut être **discrète** lorsqu'elle ne peut prendre qu'un nombre fini de valeurs, elle est **continue** lorsqu'elle peut prendre un nombre infini de valeurs dans un intervalle.

Exemple :

Le nombre d'enfants par famille. Le poids d'une personne.

Soit l'étude suivante : Recensement du nombre d'enfants par ménage dans un échantillon de 133 familles.

Enfants/ménage x_i	Répétition n_i
0	2
1	8
2	10
3	52
4	25
5	14
6	17
7	2
8	0
9	2
10	1
Total	n = 133

$n = \sum x_i \, . \, n_i$

On définit :

L'étendue :

L'étendue est la différence entre la plus grande et la plus petite valeur de l'échantillon.

$$\text{Etendue} = E = x_{max} - x_{min}$$

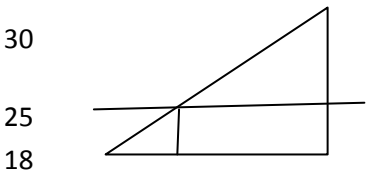
La médiane :

La médiane est la valeur centrale par excellence car elle divise la distribution en 2 parties égales.
Lorsque l'effectif total de la série est impaire, (EffTotal = 2*n + 1), c'est simple, la médiane est la $(n+1)^{ème}$ valeur.
Si par contre, l'effectif total est paire, il s'agit alors de ($n^{ème}$ valeur + $(n+1)^{ème}$ valeur)/2
Dans le cas d'une série discrète, pas de problème.
Dans le cas d'une série continue, il faut faire une interpolation linéaire.
Par exemple : soit la série suivante :

[0 ; 5[	10
[5 ; 8 [	8
[8 ;12 [	12
[12 ;15[	11
[15; 20[	9
	50

la médiane est donc la moyenne entre la 25 et la $26^{ème}$ valeurs.

La $25^{ème}$ valeur est dans l'intervalle [8 ;12[(celles-ci sont supposées répartie uniformément dans l'intervalle) et est la $7^{ème}$ valeur de l'intervalle qui en contient 12.



Etendue = 4
$n_i = 12$

$\text{Mediane}_1 = 8 + \frac{4}{12} * 7$ = 10.33

Début de l'intervalle

Position dans l'intervalle.

Idem pour la $26^{ème}$ valeur et faire la moyenne.

Le mode :
Le mode est la valeur la plus fréquente dans un échantillon.

Remarque : Il peut y avoir plusieurs modes.

Exemple : dans notre exemple, le mode est 3.

La moyenne arithmétique :

La **moyenne arithmétique** d'un échantillon d'effectif n est notée $\bar{x}$ et est définie par l'expression

$$Moy = \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Cette valeur situe le centre de la distribution statistique.

Remarque : elle est sensible aux valeurs aberrantes (car elle fait intervenir toutes les valeurs de l'échantillon).

L'écart-type :

L'écart-type d'un échantillon de données d'effectif n est noté s et est défini par l'expression

$$EcartType = s = \sqrt{\frac{\sum n_i.(x_i - \bar{x})^2}{n}}$$

Où pour des raisons de précision, il et préférable d'utiliser la formule suivante

$$EcartType = s = \sqrt{\frac{\sum n_i.{x_i}^2 - \left(\sum n_i.x_i\right)^2/n}{n}}$$

L'écart-type est un indicateur de dispersion.

Pour 2 échantillons ayant la même moyenne, si **s1** > **s2**, le premier échantillon est plus dispersé que le second.

Exemple :

La valeur de l'écart-type pour nos famille est s = 1.68

Dans le cas d'une distribution suivant une symétrie de Gauss, on peut dire que 95% de l'effectif est situé dans l'intervalle [x – 2s, x + 2s] .

Coefficient de variation :

Le calcul de l'écart-type et de la moyenne est très utile pour déterminer la précision.

Le coefficient de variation, exprimé en % est donné par la formule

$$CV = \frac{s}{\bar{x}}.100\%$$

Plus ce coefficient est faible, plus les mesures sont précises.

Remarque :

Une autre application du calcul de l'écart-type et de la moyenne concerne le domaine de contrôle de qualité.

Si la valeur de xi est comprise dans l'intervalle [x – 2s, x + 2s], on dit que le processus est sous contrôle. (c'est-à-dire. Le cas de 95% des valeurs observées)

Si xi est hors des limites xi $\pm$ 2s mais endéans des limites xi $\pm$ 3s, on suspecte un problème.

Si xi est hors des limites xi $\pm$ 3s, le processus est hors contrôle.

On demande de créer une classe **CSerieStatistiques1D** dont le rôle est de calculer les paramètres statistiques d'une série statistique à une dimension :
A savoir :
– tendance centrale : moyenne (getMoyenne()), mode (getMode(…) ; attention au cas multimodal), médiane (getMediane()).

– dispersion : écart-type (getEcartType()), étendue (getEtendue()), coefficient de variation (getCV()).

– une méthode AfficheRapport().

Pas de commentaire sur les premières méthodes.

La méthode AfficheRapport() [1]

```
Nom :
Sujet de l'étude :
type :



Effectif total :
Nom :
Sujet de l'étude :
type :


Donnees :
---------
        …

Effectif total :

   Moyenne :
   Médiane :
   Mode :          0 : 0 : 0
   Ecart type :
   Coefficient de variation : %

Contrôle de qualité:
```

[1] Voir la description des fichiers de données plus bas.

```
valeur min     :
valeur Maximum :

Valeurs de la statistique sous contrôle.
```

Réalisation :

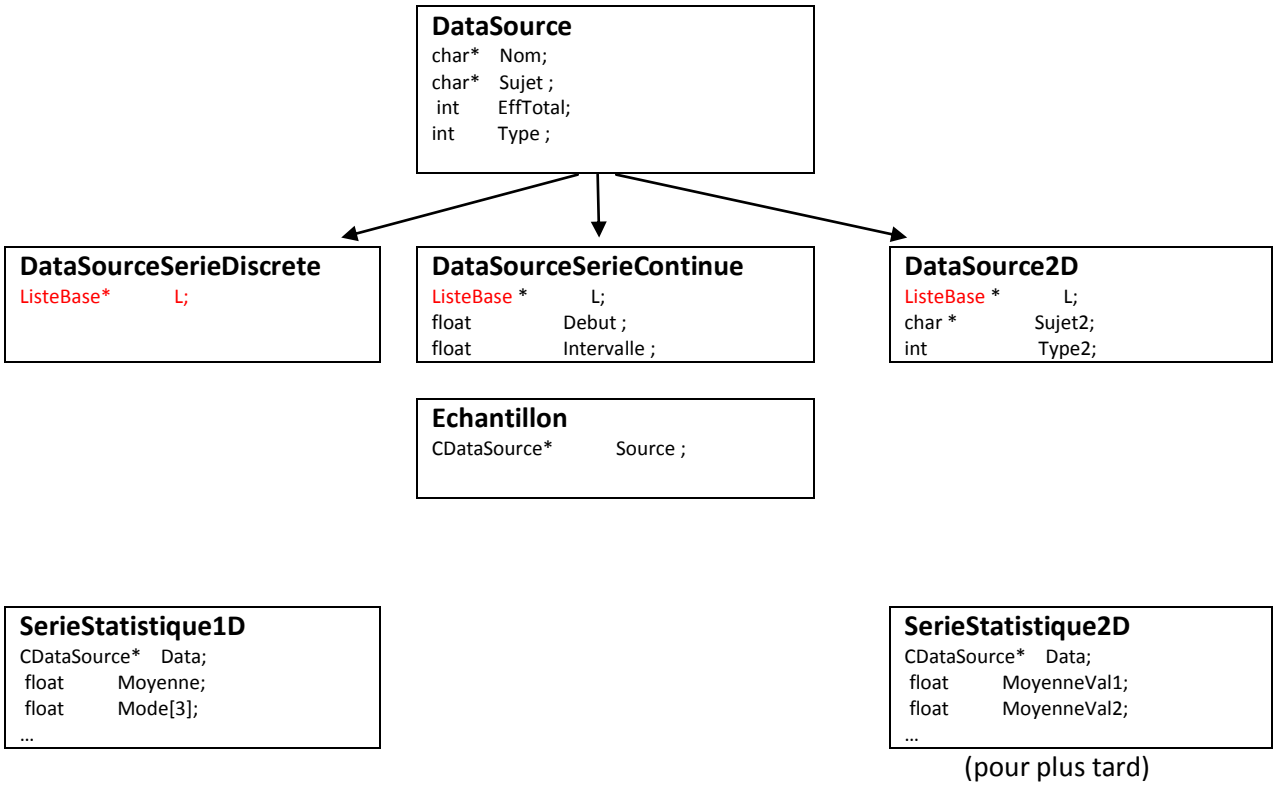
Il vous sera livré plusieurs fichiers de données pour tester votre programme.
Ils seront tous sous le format suivant :
1ère ligne : Nom de la statistique
2ème ligne : sujet de l'étude
3èmè ligne : Type de données (Continue /Discrète)
Lignes suivantes : Données

Si le fichier possède plusieurs sujets d'études, les différents champs seront séparés par le caractère ' : ' .

Exemple :

```
Résultats examens des Info (2 dernières années) Examens présentés.
ORG.ENT(/20):POO(/20):SYST.EXPL(/20):RESEAUX(/20):RES.TECH.INTERNET(/20):AN
GLAIS(/20):APOO(/20)
C:C:C:C:C:C:C
1.5:13.5:13.5:11.4:12.2:11.4:12
8.5:12.5:12.5:12.5:10.6:11:15.4
6:12.5:12.5:12.5:10.6:11:15.4
…
```

Pour cela, il faut les différentes classes suivantes :



(pour plus tard)

En fait, on doit disposer d'une liste de données de 2 valeurs ((Valeur[2], Effectif[3]) pour une statistique 1D ou (Valeur, Valeur) pour une 2D). Il s'agit de la ListeBase.

Quelque soit l'étude faite, on doit disposer de cette ListeBase, et les éléments de cette ListeBase seront interprétés comme (Valeur,Effectif) s'il s'agit d'une statistique 1D ou de (Valeur, Valeur) s'il s'agit d'une 2D.

Pour cela, il faut définir les classes suivantes :

Maintenant, on peut créer notre échantillon, c'est-à-dire nos données pour effectuer ensuite notre étude statistique.

Notre classe Echantillon() aura donc comme variable membre une liste DataSource* .

Suivant le type de statistique (discrète ou continue), La DataSource* sera interprétée comme un **DataSourceSerieDiscrete* ou un DataSourceSerieContinue* .**[4]

.

[2] float
[3] int
[4] Voir exercice test4 de Mr Wagner

REMARQUE :

Pour créer ListeBase, je vous conseille de lire une liste intermédiaire qui sera triée, ensuite de créer la ListeBase. (car il faut calculer l'effectif de chaque valeur)

REMARQUE :

Dans le cas d'une série statistique discrète, pas de problème.

Dans le cas d'une série statistique continue, il faut évidemment connaitre la valeur minimale et la maximale ainsi que la longueur de l'intervalle. Par exemple, si on calcule le poids des hommes adultes en Belgique, on ne commence pas à partir de 0 avec un intervalle de 20kg. Cela n'aura pas de sens. Il faut donc afficher ces 2 valeurs (Min et Max), et demander à l'opérateur d'introduire la valeur de l'intervalle ainsi que son point de départ.

REMARQUE :

En ce qui concerne la ListeBase et ListeTriee, il faut utiliser celles crées lors du premier dossier.

```
template <class Type> struct Noeud
   { Type               T;
     Noeud<Type>        *pSuivant;
   };
```

Note : dans les fichiers que je vous transmets, j'utilise une méthode Ajout()

getElement(i). A vous de (peut-être) compléter votre code.

REMARQUE :

Pour mieux interpréter, je vous donne 2 classes QT pour afficher l'histogramme et l'histogramme cumulé.

Il ne faut pas modifier le code, mais respecter le nom des variables et des méthodes utilisées

(d'où ma remarque précédente).

On vous donne également le makefile correspondant à l'application ainsi que le programme de départ.

L'application aura donc l'allure suivante :

```
student@solaris11DMSept2015:~/CPPStat2016Bis$ cat Applic.cpp
#include <iostream>
using namespace std;
#include "EtudeStatDescriptive.h"
#include "ExceptionBase.h"
#include "Data1D.h"
#include <QApplication>

QApplication* a;

int main(int argc,char* argv[])
{
a = new QApplication(argc,argv);
try
{
EtudeStatDescriptive E(argc,argv);
}
catch (ExceptionBase e)
{
cout << "Err. " << e.getMessage() << endl;
}
}
student@solaris11DMSept2015:~/CPPStat2016Bis$


student@solaris11DMSept2015:~/CPPStat2016Bis$ Applic
Err. mauvais parametres
student@solaris11DMSept2015:~/CPPStat2016Bis$ Applic DonneeNbEnf.txt
```

…

Et enfin la classe principale

```
student@solaris11DMSept2015:~/CPPStat2016Bis$ cat
Utile/EtudeStatDescriptive.cpp
#include "EtudeStatDescriptive.h"
#include "ExceptionBase.h"
#include "EtudeStat1D.h"

EtudeStatDescriptive::EtudeStatDescriptive(int argc, char*argv[])
{
if (argc == 2)
   EtudeStat1D E(argv[1],0);
if (argc == 3)
   EtudeStat1D E(argv[1],atoi(argv[2]));
//if (argc == 4)
//   EtudeStat2D E(argv[1],atoi(argv[2]),atoi(argv[3]));
else
      throw ExceptionBase("mauvais parametres");
}

EtudeStatDescriptive::~EtudeStatDescriptive()
{

}
```

Reste maintenant à connaitre les paramètres et méthodes

```
class DataSource
{
private:
        char*   Nom;
        char*   Sujet;
        int     EffTotal;
        int     Type;

...

class DataSourceSerieDiscrete:public DataSource
{
private:
        ListeBase<Data1D>           L;
…



class DataSourceSerieContinue:public DataSource
{
private:
        ListeBase<Data1D>           L;
        float                       Debut;
        float                       Intervalle;
…

class EtudeStat1D
{
private:
        Echantillon*    E;
        float                       Moyenne;
        float                       EcartType;
        float                       Mediane;
        float                       Mode[5];
…

template <class Type> class ListeBase
{
…

        virtual Type Ajout(Type);
         Type getElement(const int) const;
        int Size() const;
        ListeBase<Type> operator=(const ListeBase<Type>&);
…
```