Laboratoire de Programmation statistique en C++

2ème informatique de gestion (2ème quadrimestre)

Année académique 2021-2022

Calcul de variables statistiques à partir de données contenues dans un fichier

Patrick Quettier

1 Informations générales : UE, AA et règles d'évaluation

Cet énoncé de laboratoire concerne l'unité d'enseignement (UE) " Introduction au traitement statistique des données " au niveau de l'activité d'apprentissage (AA) "_Programmation de questions statistiques".

1) Chaque étudiant doit être capable d'expliquer et de justifier l'intégralité du travail.

2) Les consignes de présentation des travaux sont fournies au laboratoire.

2 Le contexte : Calcul de variables statistiques à partir de données contenues dans un fichier

L'application à développer dans le cadre de ce laboratoire consiste en la programmation d'une application en C++ sous Linux (à l'aide de la machine virtuelle fournie en début d'année scolaire aux cours de C++ et de Unix).

Cette application comportera 2 grandes parties :

- Réaliser une étude statistique 1D
- Réaliser une étude statistique 2D

Ce laboratoire consiste en la programmation d'une application en C++ permettant le calcul de variables statistiques à partir d'un fichier de données.

2.1 Étude statistique 1D

2.1.1 Quelques rappels de base :

Une **population** est un ensemble d'individus (ou objets) ayant un point commun.
Un **échantillon** est un sous-ensemble de la population étudiée.

Par exemple la population est l'ensemble des étudiants de la Haute Ecole. Un échantillon est un ensemble de quelques étudiants de la Haute Ecole tirés aléatoirement.

Une variable statistique est la caractéristique sur laquelle s'effectue l'étude. Elle est **quantitative** si elle peut être mesurée. Elle est **qualitative** si elle exprime une qualité, c'est-à-dire qu'elle ne se mesure pas.

Exemple de variable quantitative : l'âge d'une personne, la taille d'une personne.

Exemple de variable qualitative : une personne est mariée ou non, une personne est âgée de moins de 20 ans ou non.

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 variable quantitative discrète : le nombre d'enfants par famille, le nombre de cours qu'un étudiant doit suivre.
Exemple variable quantitative continue : le poids d'une personne, le temps pour courir 5 km.

Exemple, 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

On définit :

L'étendue : c'est la différence entre la plus grande et la plus petite valeur de l'échantillon. Etendue = E = $x_{max} - x_{min}$

La médiane : c'est la valeur centrale qui divise la distribution en 2 parties égales.

Lorsque l'effectif total de la série est impair, (EffTotal = 2*n + 1), c'est simple, la médiane est la $(n+1)^{ème}$ valeur.

Si par contre, l'effectif total est pair, il s'agit alors de ($n^{ème}$ valeur + $(n+1)^{ème}$ valeur)/2 Dans le cas d'une série discrète le calcul est aisé.

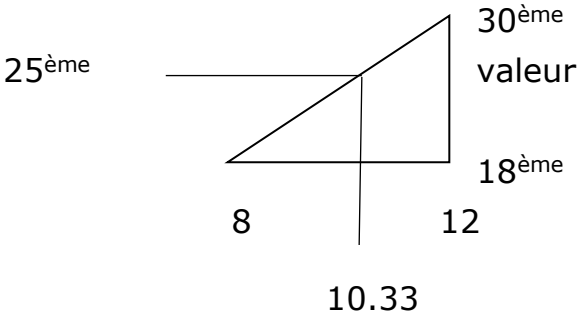
Dans le cas d'une série continue, il faut faire une interpolation linéaire. Par exemple : soit la série suivante :

[0 ; 5[10
[5 ; 8[8
[8 ;12[12 L'effectif total étant paire (50), la médiane est la
[12 ;15[11 moyenne entre la 25 et la $26^{ème}$ valeurs.

[15; 20[	9
Effectif total	50

La 25ème valeur est située dans l'intervalle [8 ;12[(celles-ci sont supposées répartie uniformément dans l'intervalle). Puisqu'il y a 18 éléments dans les intervalles précédents, elle est la 7ème valeur de l'intervalle [8 ;12[qui en contient 12.

Pour cet intervalle, l'étendue est égale à 4 et l'effectif à 12.
Etendue = 4
$n_i = 12$



Répartition uniforme des valeurs de l'intervalle dans l'étendue ➔ 4/12 entre chaque valeurs

Mediane_1 (25ème valeur) = $8 + \frac{4}{12} * 7 = 10.33$

Début de l'intervalle

Position dans l'intervalle

Idem pour la 26ème valeur : Mediane_2 (26ème valeur) = $8 + \frac{4}{12} * 8 = 10.67$

Il n'y a plus qu'à faire la moyenne : Médiane=(10,33+10,67)/2=10,5

Le mode : c'est la valeur la plus fréquente dans un échantillon. Dans le cas d'un étude continue, c'est l'intervalle avec l'effectif le plus grand.

Remarque : Il peut y avoir plusieurs modes.

Dans l'exemple avec le nombre d'enfants par ménage, 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

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

Pour des raisons de précision, il est préférable d'utiliser la formule suivante :

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

L'écart-type est un indicateur de la dispersion.

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

Exemple :
La valeur de l'écart-type pour nos familles 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 x_i 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 x_i est hors des limites $x_i \pm$ 2s mais endéans des limites xi $\pm$ 3s, on suspecte un problème.
Si x_i est hors des limites $x_i \pm$ 3s, le processus est hors contrôle.

2.1.2 Programmation (1ère partie)

On demande de créer une classe **EtudeStat1D** dont le rôle est de calculer les paramètres statistiques d'une série statistique à 1 dimension et d'afficher les résultats dans un rapport :

– 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()[1] qui affiche le résultat de l'étude 1D

```
Nom :                Sujet de l'étude :
type :


Donnees :
---------
        …

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

Contrôle de qualité:
valeur min :         valeur Maximum :
```

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

2.1.3 Format des fichiers de données :

Plusieurs fichiers de données vous seront fournis pour tester votre programme.

Ils seront tous sous le format suivant :

1ère ligne : Nom de la statistique
2ème ligne : Sujet de l'étude
3ème ligne : Type de données (Continue /Discrète)
Lignes suivantes : Données

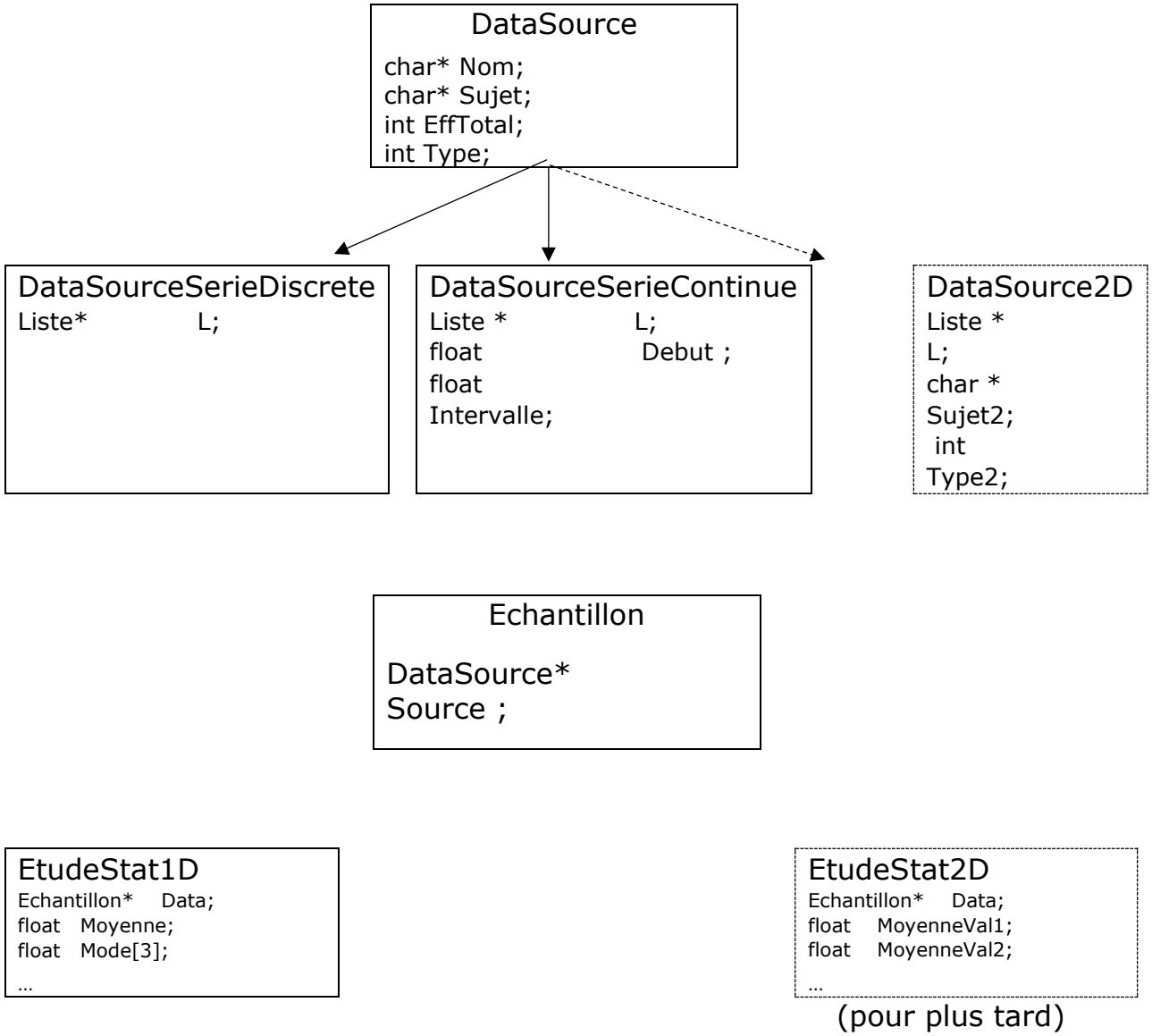
Si le fichier possède plusieurs sujets d'études, les différents champs seront séparés par le caractère ' : '.

Exemple :

```
Résultats examens des Info (2 dernières années) Examens présentés.
ORG.ENT(/20):POO(/20):SYST.EXPL(/20):RESEAUX(/20):RES.TECH.INTERNET(/20):
ANGLAIS(/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
…
```

2.1.4 Classes à développer

Pour cela, il faut les différentes classes suivantes :



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 Liste L (variable membre d'une des classes DataSourceXXX).

Quelle que soit l'étude faite, on doit disposer de cette Liste, et les éléments de cette Liste 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 (qui seront les éléments de notre liste) :

[2] float

[3] int

Data1D
float Val;
int Eff;

Data2D
float Val1;
float Val2;

Ensuite, on peut créer notre classe Echantillon. C'est elle qui contiendra nos données pour effectuer ensuite notre étude statistique.

Notre classe Echantillon() aura donc comme variable membre une liste DataSource* . Suivant le type d'étude statistique (1D discrète ou continue), DataSource* sera interprété comme un DataSourceSerieDiscrete* ou un DataSourceSerieContinue* (ou DataSource2D* plus tard).

REMARQUE :

Pour créer Liste, je vous conseille de lire les données et de créer une liste intermédiaire triée, ensuite de créer la Liste à partir de la liste triée (cela facilitera le calcul de l'effectif de chaque valeur).

Dans le cas d'une série statistique discrète c'est assez facile.

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

REMARQUE :
En ce qui concerne Liste et ListeTriee, il faut utiliser celles créées lors du dossier de C++ durant le 1er quadrimestre (basées sur la structure suivante). Reprendre l'Iterateur peut aussi s'avérer intéressant…

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

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

Le code de l'application aura l'allure suivante :

```
#include <iostream>
using namespace std;
#include "EtudeStatDescriptive.h"
#include "ExceptionBase.h"
#include "Data1D.h"


int main(int argc,char* argv[])
{
  try
  {
    EtudeStatDescriptive E(argc,argv);
  }
  catch (ExceptionBase e)
  {
    cout << "Err. " << e.getMessage() << endl;
  }
}
```

Il faut passer le nom du fichier de données à analyser en paramètre sur la ligne de commande plus éventuellement le numéro de la colonne de données qui nous intéresse.

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

Et la classe "principale" qui lancera le constructeur de l'étude souhaitée :

```
#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()
{
}
```

Autres classes :

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

```
class DataSourceSerieDiscrete:public DataSource
{
private:
   Liste<Data1D>   *L;
…
}
```

```
class DataSourceSerieContinue:public DataSource
{
private:
   Liste<Data1D> *L;
   float      Debut;
   float      Intervalle;
…
}
```

```
class EtudeStat1D
{
private:
   Echantillon*    E;
   float     Moyenne;
   float     EcartType;
   float     Mediane;
   float     Mode[3];
   …
}
```

3 Etapes de développement Etudestat1D

Comme pour le cours de C++ du 1er quadrimestre, il vous est demandé de permettre de tracer vos constructeurs et destructeurs si c'est spécifié lors de la compilation.

3.1 La classe de base pour stocker les données

Le but est de développer la classe Data1D qui contiendra une valeur et son effectif. Dans le cas d'une étude discrète, il s'agit réellement de la valeur et de son effectif. Dans le cas d'une étude continue, la valeur sera le milieu de l'intervalle et l'effectif représente le nombre de valeurs présentes dans cet intervalle.

Cette classe aura comme variables membres **val** de type float pour la valeur et **Eff** de type int pour l'effectif.

Il faut prévoir les 3 constructeurs classiques (par défaut, d'initialisation et de copie), un destructeur, les getters et setters, et surcharger l'opérateur << et un opérateur de comparaison qui sera nécessaire pour la liste triée.

Il faut penser à l'instanciation des templates de liste et listeTriée pour cette classe :

```
#include "Data1D.h"
template struct Cellule<Data1D>;
template class  Liste<Data1D>;
```

```
#include "Data1D.h"
template class ListeTriee<Data1D>;
```

3.2 Hiérarchie de classes DataSourceXXX

Il s'agit à présent de développer une hiérarchie de classes dont le but est de contenir les données lues dans le fichier.

La classe mère DataSource :

Elle contiendra les données présentent dans les fichiers de données fournis quelle que soit le type de l'étude (1D discrète, 1D continue ou 2D). Il s'agit du nom de l'étude (**Nom**), du Sujet de l'étude (**Sujet**), de l'effectif total (**EffTotal**) et du type de l'étude (**Type**) qui vaudra 0 pour une étude discrète et 1 pour une étude continue.

Il faut écrire les constructeurs, le destructeur, les getters et setters ainsi qu'une méthode Affiche() qui affiche le nom, le sujet, l'effectif total et le type de l'étude.

La classe fille DataSourceSerieDiscrete :

Elle hérite de la classe DataSource et elle y ajoute un pointeur de Liste **L** qui pointe vers la liste de Data1D contenant les données de l'étude.

Il faut écrire les constructeurs, le destructeur, les getters et setters ainsi qu'une méthode Affiche() qui affiche le nom, le sujet, l'effectif total, le type de l'étude et la liste de Data1D. PS : pensez à utiliser les méthodes de la classe mère.

La classe fille DataSourceSerieContinue :

Elle hérite de la classe DataSource et elle y ajoute un pointeur de Liste **L** qui pointe vers la liste de Data1D contenant les données de l'étude, le début du 1er intervalle et la taille des intervalles.

Il faut écrire les constructeurs, le destructeur, les getters et setters ainsi qu'une méthode Affiche() qui affiche le nom, le sujet, l'effectif total, le type de l'étude, la liste de Data1D, le début du 1er intervalle et la taille des intervalles. PS : pensez à utiliser les méthodes de la classe mère.

3.3 Créer la source de données à partir du fichier

Le but ici est de développer la classe Echantillon. Elle aura pour but de lire les données dans le fichier et de créer le DataSourceXXX approprié.

Cette classe contient un pointeur de DataSource qui pointera selon l'étude sur un DataSourceSerieDiscrète ou un DataSourceSerieContinue (ou un DataSource2D plus tard).

Elle doit disposer d'un constructeur d'initialisation recevant le nom du fichier de données ainsi qu'un entier indiquant soit que le fichier n'a qu'une seule colonne de données, soit le numéro de la colonne qui nous intéresse.

```
Echantillon(const char *name,int val)
```

Les fichiers sont des fichiers textes. Il faudra donc convertir les valeurs en float.

Dans le cas d'une série continue, faites attention que le séparateur entre la partie entière et la partie décimale en C++ est le point "**.**". Si les valeurs dans le fichier utilisent une virgule "**,**" comme séparateur, il faut la remplacer par un point avant de faire la conversion.

3.4 La classe réalisant le calcul des variables statistiques

Pour finir, il faut développer la classe EtudeStat1D qui va calculer les diverses variables statistiques souhaitées grâce à l'échantillon créé à partir des données lues.

Cette classe contient principalement un pointeur d'Echantillon. Elle contient aussi les variables membres nécessaires pour stocker les valeurs des différentes variables statistiques (Moyenne, EcartType, etc).

Elle est dotée d'un constructeur d'initialisation recevant le nom du fichier de données ainsi qu'un entier indiquant soit que le fichier n'a qu'une seule colonne de données, soit le numéro de la colonne qui nous intéresse. Ces paramètres seront utilisés lors de l'appel au constructeur d'Echantillon.

```
EtudeStat1D(const char*,int);
```

Le calcul des diverses variables statistiques se fait lors du passage dans le constructeur. En effet, elles dépendent des données de l'échantillon et il n'y a pas lieu de les recalculer par après.

Cependant, pour des raisons évidentes de lisibilités, il vous est demandé d'effectuer ces calculs dans des fonctions et d'appeler ces fonctions depuis le constructeur d'EtudeStat1D. Ces fonctions ne devant pas être appelées autrement, elles seront privées.

Il faut aussi prévoir un destructeur et une méthode permettant d'afficher le rapport de l'étude.

4 Etude statistique 2D

4.1 Introduction :

Une étude statistique 2D consiste à observer 2 variables simultanément sur une population. On peut ensuite se demander s'il y a un lien entre ces variables pour cette population. Par exemple, y a-t-il une relation entre le poids et la taille des individus ?

Dans ce cas, lorsque les 2 variables X et Y sont calculées simultanément, on dit que l'on a affaire à un problème de corrélation.

Ces 2 variables sont quantitatives et le plus souvent continues.

Exemple : soit l'étude suivante : Age d'un médecin (années) et expérience professionnelle (années)

Age du médecin x_i	Expérience y_i
40	5
46	20
38	13
34	9
33	8
47	22
44	20
55	27
41	16
31	6
45	20
42	17
60	34
42	15
32	8

On obtient donc, pour chaque mesure, un couple (X,Y).

Calcul de la corrélation :

Ce calcul est important, car il permet de connaitre la dépendance qui existe entre les 2 variables. Plus le coefficient de corrélation est proche des valeurs extrêmes -1 et 1, plus la corrélation entre les variables est forte. Une corrélation égale à 0 signifie que les variables sont indépendantes.

Le coefficient de corrélation n'est pas sensible aux unités de chacune des variables. Ainsi, par exemple, le coefficient de corrélation linéaire entre

l'âge et le poids d'un individu sera identique que l'âge soit mesuré en semaines, en mois ou en années. En revanche, ce coefficient de corrélation est extrêmement sensible à la présence de valeurs aberrantes.

Le coefficient de corrélation est donné par la formule :

$$Corr = r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{[\sum(x_i - \bar{x})^2][\sum(y_i - \bar{y})^2]}}$$

Mais, toujours pour des raisons d'erreurs d'arrondis, on préfère utiliser la formule suivante ;

$$Corr = r = \frac{\sum x_i y_i - \frac{(\sum x_i)(\sum y_i)}{n}}{\sqrt{\left[\sum {x_i}^2 - \frac{(\sum x_i)^2}{n}\right]\left[\sum {y_i}^2 - \frac{(\sum y_i)^2}{n}\right]}}$$

Pour l'exemple donné, r = 0.946

Il est aussi possible d'obtenir la relation entre ces 2 variables. Ainsi, à partir d'une variable on peut estimer la valeur de l'autre. Pour cela, il faut calculer la droite de régression.

Une droite a pour équation

$$y = a\,x + b$$

avec

$$A = a = \frac{\sum x_i y_i - \frac{(\sum x_i)(\sum y_i)}{n}}{\sum {x_i}^2 - \frac{(\sum x_i)^2}{n}}$$

$$B = b = \bar{y} - a.\bar{x}$$

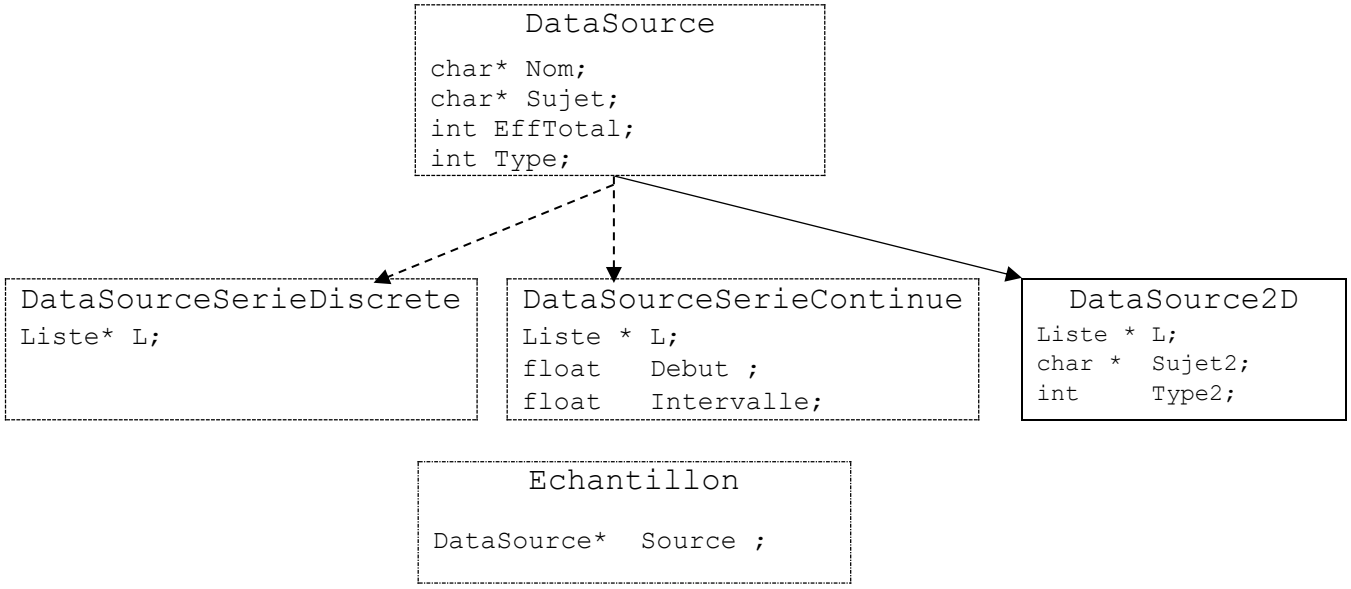
On demande de créer une classe **EtudeStat2D** dont le rôle est de calculer les paramètres statistiques d'une série statistique à 2 dimensions, à savoir : A,B,Corr .

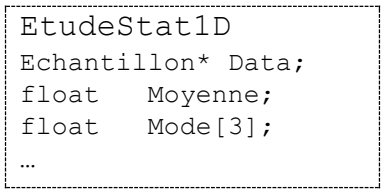
Et dans le cas d'une étude 2D, la méthode AfficheRapport() donnera :

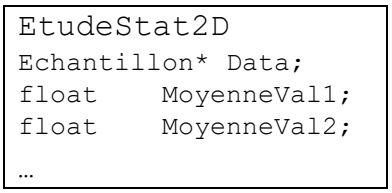
```
Etude statistique:
------------------

Titre : Temps de réaction complète (min) en fonction de la température (°)
Sujet de l'etude : Temperature (°)  --  Temps (minutes)
Effectif Total :  10
Type :  C      C

Valeurs:
        25 - 0.64
        45 - 1.27
        55 - 0.95
        85 - 1.85
        115 - 2.81
        125 - 2.8
        150 - 3.42
        165 - 4.3
        175 - 4.54
        200 - 4.7

   Moyenne Val1 : 114
   Moyenne Val2 : 2.728

Corrélation :
   Coefficient a : 0.0249807
   Coefficient b : -0.1198

1 : Prévision pour : Température (°)
2 : Prévision pour : Temps (min)
3 : Sortie
1
Entrer la valeur pour  Température (°) : 70
la valeur prévue  : 1.62885

1 : Prévision pour : Température (°)
2 : Prévision pour : Temps (min)
3 : Sortie           :
3
fin
```

4.2 Classes à développer

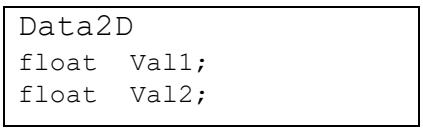
Pour cela, il faut les différentes classes suivantes :

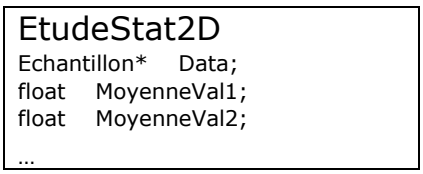






On doit disposer d'une liste de données de 2 valeurs (Valeur, Valeur) pour une étude 2D.
Il s'agit de la Liste<Data2D>.





Il va donc falloir :

- développer une classe Data2D.
- développer une classe DataSource2D qui contiendra (en plus des variables membres de la classe mère DataSource) le sujet et le type de la 2ème série de données ainsi que la liste de Data2D.

- ajouter un constructeur à la classe Echantillon. Ce constructeur reçoit un paramètre en plus que celui pour une étude 1D : le numéro de la 2ème colonne de données dans le fichier.
- Développer une classe EtudeStat2D qui calculera le coefficient de corrélation **r** ainsi que les coefficients **a** et **b** de la droite de régression linéaire.
- Dans la classe EtudeStatDescriptive le cas ou argc vaut 4 correspond à une étude 2D.

4.3 Etapes de développement Etudestat1D

Comme pour le cours de C++ du 1er quadrimestre, il vous est demandé de permettre de tracer vos constructeurs et destructeurs si c'est spécifié lors de la compilation.

4.3.1 La classe de base pour stocker les données

Le but est de développer la classe Data2D qui contiendra les 2 valeurs **Val1** et **Val2**.

Il faut prévoir les 3 constructeurs classiques (par défaut, d'initialisation et de copie), un destructeur, les getters et setters, et surcharger l'opérateur << et un opérateur de comparaison qui sera nécessaire pour la liste triée. La comparaison se fera sur **Val1**.

Il faut penser à l'instanciation des templates de liste et listeTriée pour cette classe.

4.3.2 La classe DataSource2D

Elle hérite de la classe mère DataSource. Elle y ajoute le sujet et le type de la 2ème série de données ainsi qu'un pointeur de Liste de Data2D

Il faut écrire les constructeurs, le destructeur, les getters et setters ainsi qu'une méthode Affiche() qui affiche le nom, le sujet, l'effectif total, le type de l'étude et la liste de Data2D.

4.3.3 La classe Echantillon

Il faut compléter la classe développée dans le cadre de l'étude 1D en y ajoutant un nouveau constructeur dédié à l'étude 2D. Ce constructeur reçoit le nom du fichier de données, le numéro de la colonne correspondant à la première série de données et le numéro de la colonne correspondant à la deuxième série de données.

```
Echantillon(const char*,int,int) ;
```

Voir les remarques faites dans le cadre de l'étude 1D pour les fichiers à lire et le séparateur des unités et des décimales.

4.3.4 La classe EtudeStat2D

Elle calcule le coefficient de corrélation **r** ainsi que les coefficients **a** et **b** de la droite de régression linéaire à partir de l'échantillon de données. Cette classe contient principalement un pointeur d'Echantillon. Elle contient aussi les variables membres nécessaires pour stocker les valeurs calculées (r, a et b).

Elle est dotée d'un constructeur d'initialisation recevant le nom du fichier de données ainsi que 2 entiers indiquant le numéro de la 1ère colonne et de la 2ème colonne qui nous intéressent. Ces paramètres seront utilisés lors de l'appel au constructeur d'Echantillon.

```
EtudeStat2D(const char*,int,int);
```

Pour des raisons évidentes de lisibilité et de modularité, il vous est demandé d'effectuer les calculs demandés dans des fonctions.