UNIVERSITÉ D'AVIGNON
ET DES PAYS DE VAUCLUSE

CENTRE
D'ENSEIGNEMENT
ET DE RECHERCHE
EN INFORMATIQUE

Nom de la formation
Nom du parcours/spécialité
UE Nom de l'unité d'enseignement

Titre du document

13 avril 2019
Nom/numéro du groupe de travail

Prénom1 Nom1
Prénom2 Nom2
Prénom3 Nom3

CERI - LIA
339 chemin des Meinajariès
BP 1228
84911 AVIGNON Cedex 9
France

Tél. +33 (0)4 90 84 35 00
Fax +33 (0)4 90 84 35 01

http ://ceri.univ-avignon.fr

Responsables
Prénom4 Nom4
Prénom5 Nom5

Sommaire

1 Introduction

L'utilisation de logiciels tels que MS Word ou Libre Office pour la rédaction de rapports aboutit généralement à des documents très hétérogènes, à la lisibilité douteuse, ce qui entraîne des difficultés lors de leur évaluation. Pour cette raison, c'est LaTeX[9, 27] qui a été retenu pour les rapports de cette UE.

La mise en forme d'un document est aussi importante que son contenu. Il est clair qu'un rapport bien présenté mais dont le contenu est inadapté ou incorrect obtiendra une note faible. Mais de la même façon, un rapport dont le contenu est excellent et dont la présentation est mauvaise récoltera probablement une mauvaise note, car il sera difficile d'en comprendre les idées. Donc pour résumer, une bonne présentation est une condition nécessaire, mais pas suffisante pour produire un bon rapport. Tout ça pour dire que la mise en forme de vos rapports sera évaluée au même titre que leur contenu.

LaTeX, pour simplifier, est un langage permettant de programmer la mise en forme d'un document. Donc, au lieu de composer votre document dans un traitement de texte, vous allez écrire seulement sa structure et son contenu sous la forme d'un code source (extension `.tex`). Vous devrez ensuite compiler ce code source pour obtenir un fichier `.pdf`.

LaTeX et les outils associés sont libres (et gratuits). Il s'agit d'un standard dans le domaine de la recherche académique : la plupart des articles soumis à des conférences ou à des journaux scientifiques sont mis en forme grâce à ce système.

Mais le but principal de cette UE n'est pas d'apprendre à utiliser LaTeX, c'est pourquoi le présent document tient lieu à la fois de tutoriel et de modèle pour la rédaction de rapports. Il ne s'agit pas d'un manuel complet sur LaTeX, mais seulement d'une présentation des fonctionnalités requises dans le cadre de la rédaction d'un rapport standard. N'hésitez pas à aller chercher d'autres sources d'information en ligne, par exemple ces WikiBooks en français [19] ou en anglais [20].

Le code source LaTeX de ce document vous est fourni afin que vous l'utilisiez comme base pour vos propres rapports, après en avoir supprimé les parties inutiles et les commentaires.

2 Compilation des documents

Comme expliqué en introduction, LaTeX permet de produire un document PDF en compilant un fichier source. On peut distinguer deux façons de réaliser cette compilation : soit via une application locale, soit grâce à une application Web.

La première méthode est beaucoup plus rapide lors de la compilation des documents, mais la procédure d'installation est plus ou moins complexe en fonction du système d'exploitation concerné. La seconde met plus de temps à produire le document, mais ne nécessite aucune installation, et intègre des fonctionnalités collaboratives (partage de documents, suivi des modifications, gestion des versions, système de commentaires, etc.).

Pour ces raisons, dans le contexte de cette UE, c'est cette dernière approche qui a été retenue. Nous utiliserons l'application Web *OverLeaf*, qui est disponible à l'adresse suivante :

`https://www.overleaf.com/signup?ref=d62cb1694be6`

Vous devez d'abord vous créer un compte (c'est gratuit), puis vous pourrez définir et compiler vos propres documents en ligne. Sélectionnez un identifiant correspondant à votre nom réel, et non pas à un surnom. Idéalement, quelque chose de la forme `prenom.nom`. En effet, la trace de votre activité sur OverLeaf est susceptible d'être utilisée lors de votre évaluation.

Le source LaTeX du présent document est disponible sur OverLeaf à l'adresse suivante :
`https://www.overleaf.com/latex/templates/modele-rapport-uapv/pdbgdpzsgwrt`
Il s'agit d'un modèle à partir duquel vous devez instancier vos propres rapports. Pour cela, utilisez le bouton *Open as template* présent à l'URL indiquée ci-dessus.

3 Généralités

Cette section regroupe des informations basiques, dont certaines sont spécifiques à ce document LaTeX en particulier (Sections 3.1 et 3.2), alors que les autres sont valables pour tous les documents LaTeX en général (Sections 3.3 à 3.5).

3.1 Options de la classe

La première instruction d'un document LaTeX est la classe du document produit (un peu l'équivalent d'une feuille de style). Pour ce rapport, il s'agit de la classe `ceri/rapport.cls`.

Celle-ci peut s'utiliser sans option, comme c'est le cas pour ce document :

```
\documentclass{ceri/rapport}
```

Document allégé. Il existe une option `light`, qui produit une version incomplète de la page de titre afin d'accélérer la compilation du document :

```
\documentclass[light]{ceri/rapport}
```

> **Remarque :** Ne rendez jamais un PDF produit avec cette option, elle est destinée à être utilisée seulement pendant la phase de rédaction.

Document étendu. L'option `full` rajoute quant à elle une table des figures et une table des tableaux en début de document :

```
\documentclass[full]{ceri/rapport}
```

À n'utiliser que pour de longs rapports contenant de nombreuses figures et/ou tableaux (ex. mémoire de M2). La page de titre apparaît aussi, comme quand on utilise la classe sans aucune option.

Document à imprimer. L'option `handout` permet d'obtenir un rendu du document plus adapté à une impression papier :

```
\documentclass[handout]{ceri/rapport}
```

En particulier, les environnements reproduisant la sortie d'un terminal (cf. Section 7.3) sont mis en forme en inverse vidéo pour éviter les fonds foncés.

3.2 Informations du rapport

Les informations apparaissant sur la page de titre sont à indiquer en début de document, après la déclaration de la classe. Il faut utiliser pour cela des macro LaTeX prédéfinies, qui sont décrites ci-dessous.

Formation. La commande `\major` est obligatoire. Elle sert à donner la formation délivrant l'UE :

```
\major{Master d'Informatique}
```

Parcours. La commande `\specialization` est obligatoire. Elle permet d'indiquer le parcours (ou la spécialité) de la formation :

```
\specialization{Ingénierie logicielle}
```

Unité d'enseignement. La commande `\course` est obligatoire. Elle est utilisée pour indiquer l'UE concernée par le rapport :

```
\course{Algorithmique et programmation}
```

Auteurs. La commande `\author` est elle aussi obligatoire. Elle sert à lister les auteurs, en les séparant par des `\\` :

```
\author{
  Sophie Dupont \\
  Abdel Cherif \\
  John McDouglas
}
```

Encadrants. La commande `\advisor` fonctionne exactement comme `\author`, excepté qu'elle est optionnelle. Elle permet de lister les enseignants encadrant le travail des étudiants. Attention, il ne faut l'utiliser que pour un rapport de stage, et non pas pour un projet classique.

```
\advisor{
  Fen Zhou \\
  Serigne Gueye
}
```

Groupe. La commande `\group` permet d'indiquer le nom ou le numéro du groupe de travail auquel les auteurs appartiennent. Elle est optionnelle, à n'utiliser que si le travail est réalisé en groupe :

```
\group{G12}
```

Titre. La commande `\title` est obligatoire. Elle reçoit en paramètre le titre du document :

```
\title{Rapport du premier semestre}
```

Résumé. La commande `\summary` est optionnelle. Elle est utilisée pour insérer un résumé en début de document.

```
\summary{Ce document décrit l'activité réalisée au premier semestre. Nous avons
   d'abord effectué une revue bibliographique, puis proposé un modèle de logiciel,
   et commencé l'implémentation.}
```

3.3 Résumé de la syntaxe

Un document LaTeX contient principalement : du texte, qui est rendu tel quel ; et des commandes et environnements, qui déclenchent des traitements spécifiques.

Commandes. Les noms de commande LaTeX commencent par `\`, par exemple `\macommande`. Elles prennent éventuellement des options, entre crochets, par exemple `\macommande[monoption]`. Elles peuvent également recevoir des paramètres, entre accolades, par exemple `\macommande{monparametre}`. Ou les deux : `\macommande[monoption]{monparametre}`.

Environnements. Les environnements LaTeX permettent de définir des traitements plus complexes. Ils sont délimités par deux commandes `\begin` et `\end` prenant le nom de l'environnement en paramètre. Par exemple, le texte du document est contenu dans l'environnement `document` :

```
\begin{document}
Texte du document, etc.
\end{document}
```

Commentaires. Les commentaires sont marqués grâce au caractère %, qui est valable pour toute une ligne (comme // en langage C ou Java). Par exemple :

```
% ceci est un commentaire
```

Le caractère \ peut être utilisé pour désactiver ce caractère spécial (ainsi que tous les autres caractères spéciaux, d'ailleurs).

Paragraphes. Pour aller à la ligne et ainsi créer des paragraphes, il suffit de laisser une ligne vide. Par exemple :

```
Ce texte forme un premier paragraphe.

Et voici un second paragraphe.
```

> **Remarque :** La commande \\ n'est pas équivalente à un saut de ligne. Ne l'utilisez jamais pour aller à la ligne entre deux paragraphes.

3.4 Mise en forme de base

La mise en forme de base est décrite en détail dans les références déjà citées [19, 20]. Voici rapidement les commandes les plus utiles.

Gras et italique. Le texte peut être mis en **gras** avec la commande `\textbf` ou en *italique* avec `\textit` :

```
\textbf{Texte en gras}
\textit{Texte en italique}
```

Identificateurs. Tous les identificateurs (i.e. les noms de classes, méthodes, variables, constantes...) que vous citez dans le texte doivent être mis en forme avec la commande `\texttt`. Celle-ci permet d'insérer du texte avec une police de caractères à chasse fixe [21] (de type *Courrier New* [22]). Par exemple, pour obtenir `MyClass`, on fait :

```
\texttt{MyClass}
```

> **Remarque :** Nous insistons particulièrement sur ce point, qui nous permet de rapidement repérer les identificateurs à la lecture de votre document, et donc de l'évaluer plus facilement.

Listes. Il est possible d'inclure des listes non-numérotées avec l'environnement `itemize` et la commande `\item`. Par exemple :

```
\begin{itemize}
  \item premier élément ;
  \item 2\ieme élément ;
  \item troisième élément.
  \begin{itemize}
    \item on peut aussi placer des sous-listes ;
    \begin{itemize}
      \item et rajouter encore d'autres niveaux ;
      \item etc.
    \end{itemize}
    \item on continue au niveau supérieur ;
  \end{itemize}
  \item et au niveau encore supérieur.
\end{itemize}
```

Ce qui donne le résultat suivant :

- premier élément ;
- 2e élément ;
- troisième élément.
  - on peut aussi placer des sous-listes ;
    - et rajouter encore d'autres niveaux ;
    - etc.
  - on continue au niveau supérieur ;
- et au niveau encore supérieur.

On peut également définir des listes numérotées automatiquement, en utilisant l'environnement `enumerate` (à la place d'`itemize`). On utilise toujours `\item` pour les éléments de la liste. Par exemple :

```
\begin{enumerate}
  \item premier élément ;
  \item 2\ieme élément ;
  \item troisième élément.
  \begin{enumerate}
    \item là encore, on peut placer des sous-listes ;
    \item ça marche exactement pareil que pour les listes non-numérotées ;
    \begin{itemize}
      \item on peut même mélanger les deux types de listes comme ici ;
      \item et ici.
    \end{itemize}
  \end{enumerate}
\end{enumerate}
```

Ce code source donne le résultat suivant :

1. premier élément ;
2. 2e élément ;
3. troisième élément.
   (a) là encore, on peut placer des sous-listes ;
   (b) ça marche exactement pareil que pour les listes non-numérotées ;
   - on peut même mélanger les deux types de listes comme ici ;
   - et ici.

Couleurs. La commande `\textcolor` permet de modifier la couleur du texte lui-même, tandis que la commande `\colorbox` sert à changer sa couleur de fond. Par exemple, pour obtenir du texte rouge et un fond vert, on écrit :

```
du \textcolor{red}{texte rouge} et un \colorbox{green}{fond vert}
```

Pour les couleurs, vous pouvez soit utiliser le nom de la couleur en anglais, par exemple ici `red` ou `green`, soit définir une couleur précise en utilisant des valeurs numériques [19, 20].

Remarque : Utilisez les couleurs avec une extrême parcimonie.

Notes de bas de page. Pour définir une note de bas de page, on utilise la commande `\footnote`, comme ici [1] :

```
comme ici\footnote{La numérotation se fait automatiquement.}
```

Remarque : Évitez au maximum les notes de bas de page. Pour une meilleure lisibilité, placez vos remarques directement dans le texte, entre parenthèses.

Mode mathématique. Toutes les variables mathématiques doivent obligatoirement être mises en forme en utilisant le *mode mathématique*. On entre et on sort de ce mode en utilisant le caractère $. Par exemple, le fait d'écrire `$y = ax + b$` donne le rendu $y = ax + b$.

Certaines commandes telles que `\times` ne sont acceptées que dans le mode mathématique. Par exemple, `$7 \times 2 = 14$` donne $7 \times 2 = 14$. La Section 4.4 donne plus de détails sur l'utilisation du mode mathématique.

Divers. La commande `\ieme` permet d'insérer l'abréviation française pour *deuxième*, *troisième*, etc. : 2e, 3e...

1. La numérotation se fait automatiquement.

Le caractère tilde ~ est interprété comme un espace *insécable*, i.e. les deux mots qu'il relie ne peuvent pas être séparés par une fin de ligne. Pour afficher le caractère tilde lui-même, on utilise la commande `\textasciitilde`.

Cette classe permet d'utiliser directement le point médian de l'écriture inclusive. Par exemple : chercheur·se·s, utilisateur·rice, candidat·e·s.

3.5 Structure du document

Vous n'avez pas besoin de vous soucier de la mise en forme des titres de sections : celle-ci est déjà prévue dans la classe fournie. Vous devez seulement utiliser les commandes appropriées pour indiquer où commence chaque section. Celles-ci sont listées dans la Table 1.

Niveau hiérarchique	Commande correspondante
1	`\section`
2	`\subsection`
3	`\subsubsection`
4	`\paragraph`

Table 1. Commandes utilisées pour définir les titres de sections et sous-sections.

Ainsi, pour définir une section de niveau 1, on fait :

```
\section{Titre de ma section}
\label{sec:MaSection}
Dans cette section, je vais expliquer bla bla bla bla bla...
```

Chaque section est identifiée de façon unique grâce à un label. Celui-ci doit être spécifié juste après la définition de la section, en utilisant la commande `\label`. Celle-ci reçoit le label en paramètre (dans l'exemple ci-dessus : `sec:MaSection`). Quel que soit le niveau de la section, le label qui lui est associé doit être préfixé par `sec:`.

Le niveau 3 n'est pas réellement utilisé dans ce document, car il n'y est pas nécessaire. Pour l'exemple, le voici :

3.5.1 Exemple de section de niveau 3

Notez qu'il est nécessaire de mettre une section de niveau 3 entre la section de niveau 2 et celle de niveau 4, sinon une section factice de niveau 3 numérotée 0 sera automatiquement créée.

Le niveau 4 n'est pas vraiment une section mais plutôt un titre de paragraphe, comme ci-dessous :

Exemple de section de niveau 4. Pour cette raison, le titre n'est pas sur une ligne séparée, mais il est directement placé dans le paragraphe concerné.

> **Remarque :** Ne définissez pas une section seulement pour y placer 1 ou 2 lignes de texte. Une section doit avoir un contenu assez long, sinon ce n'est pas la peine de la créer.

4 Éléments flottants et/ou numérotés

LaTeX permet d'insérer un certain nombre d'éléments dits *flottant* : figures (Section 4.1), tables (Section 4.2), et algorithmes (Section 4.3). On les qualifie de *flottant* car LaTeX est susceptible de ne pas les insérer exactement là où l'utilisateur les définit. Ces éléments sont numérotés automatiquement par LaTeX. Il en va de même pour les équations (Section 4.4), bien qu'il ne s'agisse pas d'éléments flottant.

Ces éléments doivent toujours posséder une légende (sauf les équations) et être numérotés. De plus, ils doivent être mentionnés, décrits et commentés dans le texte. Toute référence à l'un de ces éléments doit être faite en utilisant la commande `\ref`, décrite plus en détail dans la Section 5.1.

4.1 Figures

On insère une figure en utilisant l'environnement `figure` :

```
\begin{figure}[htb!]
  \centering
  \includegraphics[scale=0.5]{images/univ.jpg}
  \caption[Université d'Avignon]{La façade de l'Université d'Avignon.}
  \label{fig:uapv}
\end{figure}
```

Ce code source a pour résultat la Figure 1.

Figure 1. La façade de l'Université d'Avignon.

Positionnement. L'option `[htb!]` sert à positionner la figure. Notez que LaTeX se charge de placer automatiquement les éléments flottant dans le document.

Il est possible que l'élément ne soit pas placé exactement à l'endroit spécifié par l'utilisateur, si le contenu du document l'empêche. C'est normal, laissez le système gérer cela. C'est d'ailleurs pour cette raison qu'il faut systématiquement mentionner les éléments flottant en utilisant le système de renvois de LaTeX (cf. Section 5.1).

La commande `\centering` permet de centrer la figure, et doit toujours être utilisée.

Contenu. La figure est ici une image contenue dans le fichier, qui est insérée grâce à la commande `\includegraphics`. Celle ci prend en paramètre le chemin et le nom du fichier concerné, qui est ici `images/univ.jpg`.

Utilisez un format de fichier permettant de représenter les images de façon *vectorielle* [26], comme par exemple PDF. La qualité est bien meilleure qu'avec une représentation *matricielle* [25] comme JPEG, BMP, PNG, ou GIF. Et de plus, la taille de l'image est généralement inférieure en vectoriel.

L'option permet de spécifier le niveau de zoom `x` de l'image avec `scale=x` (taille divisée par 2, dans l'exemple). Il est aussi possible de préciser sa largeur avec `width=x` ou sa hauteur avec `height=x`. La valeur `x` peut alors être exprimée en cm (ex. `width=15cm`), ou bien comme une fraction de la largeur de la ligne grâce à la commande `\textwidth` (ex. `width=0.75\textwidth`).

Légende. Toute figure doit obligatoirement être décrite par une légende. Celle-ci est insérée grâce à la commande `\caption`. Elle prend en paramètre le texte de la légende (ici : *La façade de l'Université d'Avignon*), et éventuellement en option une version plus courte de cette légende (ici : *Université*

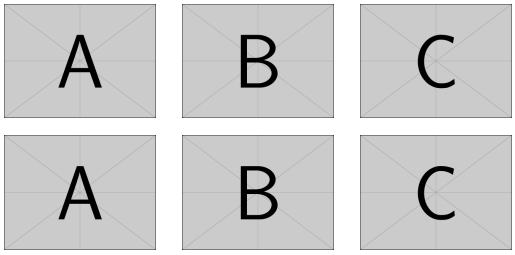


Figure 2. Exemple de figure composée de plusieurs images.

d'Avignon). Cette version courte, le cas échéant, est destinée à être affichée dans la *table des figures* (cf. Section 3.1). LaTeXse charge de numéroter automatiquement la figure.

Label. Chaque figure doit être désignée par un label unique dans le document, afin d'y faire des renvois (cf. Section 5.1). Ce label est défini juste après la légende, en utilisant la commande `\label` (qui prend le label en paramètre). Le nom unique de la figure est préfixé par `fig:`. Dans l'exemple, le label est `fig:uapv`.

Composition. On peut composer une figure en y insérant plusieurs images, graphiques ou diagrammes, en allant à la ligne et en ajustant les espacements, comme dans l'exemple ci-dessous :

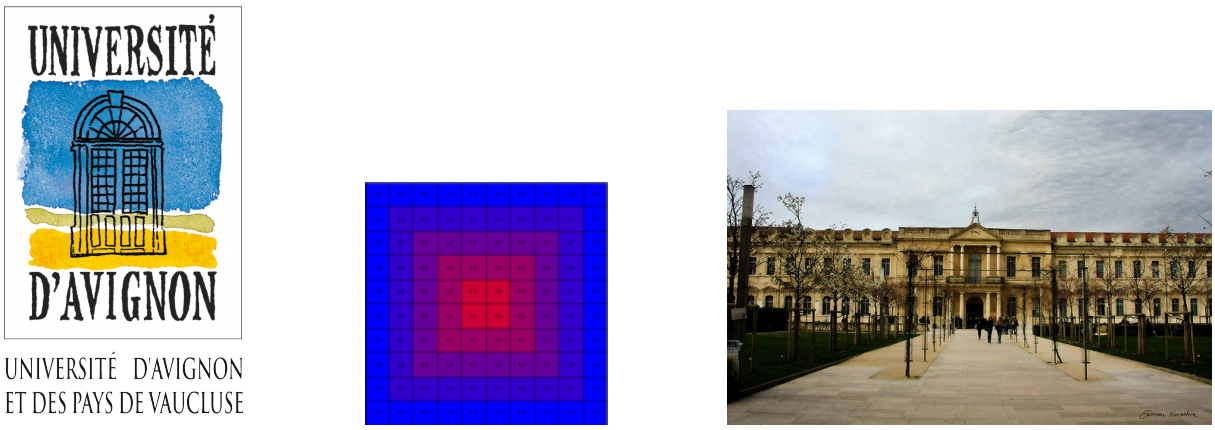
```
\begin{figure}
   \centering
   \includegraphics[width=2cm]{example-image-a} \hspace{1mm}
       \includegraphics[width=2cm]{example-image-b} \hspace{1mm}
       \includegraphics[width=2cm]{example-image-c} \\
   \vspace{2mm}
   \includegraphics[width=2cm]{example-image-a} \hspace{1mm}
       \includegraphics[width=2cm]{example-image-b} \hspace{1mm}
       \includegraphics[width=2cm]{example-image-c} \\
   \caption{Exemple de figure composée de plusieurs images de tailles différentes.}
   \label{fig:composée}
\end{figure}
```

Après compilation, ce code source produit la la Figure 2.

Notez qu'il est possible de contrôler plus finement le placement des images composant la figure, par exemple en insérant un environnement `tabular` (cf. Section 4.2) dans `figure`.

Sous-figures. Il est possible d'insérer plusieurs sous-figures dans une même figure, au moyen de la bibliothèque `subcaption`. On utilise pour cela l'environnement `subfigure`, de la façon suivante :

```
\begin{figure}
   \centering
   \begin{subfigure}[t]{0.2\textwidth}
       \includegraphics[width=\textwidth]{ceri/logo.jpg}
       \caption{Première sous-figure.}
       \label{fig:subfig1}
   \end{subfigure}
   \hfill
   \begin{subfigure}[t]{0.2\textwidth}
       ...
   \end{subfigure}
   \hfill
   \begin{subfigure}[t]{0.2\textwidth}
       ...
   \end{subfigure}
   \caption{Exemple de figure contenant trois sous-figures.}
```



(a) Première sous-figure. **(b)** **(c)** Troisième sous-figure.

Figure 3. Exemple de figure contenant trois sous-figures.

```
    \label{fig:subfigures}
\end{figure}
```

Ce code source a pour résultat la Figure 3, qui contient les sous-figures suivantes : Figures 3a, 3b et 3c.

L'option de `subfigure` contrôle l'alignement vertical de la figure (ici : `t` pour *top*), tandis que son paramètre définit sa largeur (ici : `0.2\textwidth`, soit 20% de la largeur disponible). La commande `\hfill` permet d'insérer un espace *élastique* entre les sous-figures, qui s'ajuste automatiquement à leur largeur.

On peut définir une légende et un label pour chaque sous-figure, en plaçant les commandes `\caption` et `\label` dans l'environnement `subfigure`. En laissant la légende vide, on obtient seulement la lettre, comme pour la Figure 3b. On peut aussi définir une légende et un label pour la figure entière, en les plaçant après le dernier environnement `subfigure`.

4.2 Tables

L'environnement permettant d'insérer une table est `table` :

```
\begin{table}[htb!]
  \centering
  \rowcolors{1}{LightColor}{}
  \begin{tabular}{l c r r}
    \hline
    \rowcolor{DarkColor}
    \textbf{Texte} & \textbf{Texte} & \textbf{Entiers} & \textbf{Réels} \\
    \hline
    Blabla bla bla bla bla & Blabla bbla bla & 21 & 12,62 \\
    Blabla bla bla bla & Blabla bla bla bla bla & 12~356 & 3~456,21 \\
    Blabla bla bla & Blabla bla bbla bla & 1 & 45,87 \\
    Blabla blabla bla bla & $y = ax + b$ & 456 & 72,16 \\
    Blabla bla blaa bla & Blabla bla bla bla & 78 & 89,28 \\
    \hline
  \end{tabular}
  \caption[Exemple de table hétérogène]{Exemple de table contenant du texte, des
      valeurs entières, réelles et des formules mathématiques.}
  \label{tab:exemple}
\end{table}
```

Le résultat du code source ci-dessus est la Table 2.

Texte	Texte	Entiers	Réels
Blabla bla bla bla bla	Blabla bbla bla	21	12,62
Blabla bla bla bla	Blabla bla bla bla bla	12 356	3 456,21
Blabla bla bla	Blabla bla bbla bla	1	45,87
Blabla blabla bla bla	$y = ax + b$	456	72,16
Blabla bla blaa bla	Blabla bla bla bla	78	89,28

Table 2. Exemple de table contenant du texte, des valeurs entières, réelles et des formules mathématiques.

Positionnement. Le contrôle de la position de la table au moyen de l'option `[htb!]` de l'environnement `table`, ainsi que de la commande `\centering`, fonctionne exactement comme pour les figures.

Contenu. La syntaxe permettant de définir le contenu des tables en LaTeX est relativement compliquée. Vous vous limiterez à des tables de la forme décrite dans cette sous-section : cette mise en forme doit être respectée, afin de garder des tables lisibles.

Le contenu de la table est défini au moyen de l'environnement `tabular`. Le paramètre permet de préciser à la fois le nombre de colonnes contenues dans la table, et leur alignement :

- `l` (*left*) pour une colonne alignée à gauche ;
- `r` (*right*) pour qu'elle soit alignée à droite ;
- `c` (*center*) pour qu'elle soit centrée.

Les données de la table sont ensuite décrites ligne par ligne. Chaque ligne de la table est décrite par une ligne de texte terminée par `\\`. Sur la ligne de texte, les colonnes sont séparées par le caractère `&`.

Mise en forme. Les lignes horizontales de séparation sont tracées grâce à la commande `\hline`. Une ligne sur deux est grisée afin de faciliter la lecture, grâce à la commande `\rowcolors{1}{LightColor}{}` (placée avant `tabular`).

La première ligne de la table contient les titres des colonnes. Le texte contenu dans cette ligne de titre doit être en **gras** (commande `\textbf`). La ligne doit être colorée de façon plus sombre, grâce à la commande `\rowcolor{DarkColor}` (placée juste avant le début de la ligne).

Vos tables doivent ressembler exactement à celles données en exemple dans ce document : couleurs, séparations, légende située en-dessous, etc. Les valeurs numériques doivent être alignées à droite. Les réels doivent être définis de manière à avoir le même nombre de décimales. Les valeurs textuelles peuvent être alignées à droite ou à gauche, ou bien centrées. Il est possible d'insérer des équations en-ligne (ie. non-numérotées) dans une table, en utilisant simplement le mode mathématique.

Légende. La légende de la table est insérée exactement comme pour les figures, en utilisant la commande `\caption`.

Label. Le label permettant de faire référence à la table s'insère aussi comme pour les figures, juste après la légende, et grâce à la commande `\label`. Il y a toutefois une différence : le label associé à un table est préfixé par `tab:` (au lieu de `fig:` pour les figures).

Table haute. Si votre table est haute et étroite, alors vous devez la décomposer afin de mieux utiliser l'espace. Ceci est illustré avec la Table 3, qui contient 50 lignes pour seulement 2 colonnes.

La décomposition en plusieurs parties est réalisée en utilisant l'environnement `minipage` (dans `table`). Voici un aperçu du code source de la Table 3 (consultez le code source de ce document pour le voir en entier) :

```
\begin{table}[htb!]
  \rowcolors{1}{LightColor}{}
```

Col1	Col2	Col1	Col2	Col1	Col2	Col1	Col2	Col1	Col2
1	hdc	11	ytr	21	ioe	31	zyc	41	oze
2	orf	12	xcv	22	snu	32	opv	42	anb
3	seh	13	fsd	23	dgp	33	azv	43	kir
4	lig	14	pui	24	zbu	34	xdf	44	sfk
5	jhg	15	gsf	25	zvz	35	yzd	45	bed
6	azr	16	aze	26	orv	36	sgg	46	pyj
7	opi	17	bcv	27	czr	37	sss	47	zdj
8	hgj	18	wxc	28	azh	38	zdv	48	pyj
9	xvc	19	uyr	29	cyh	39	sgu	49	zen
10	ree	20	ndz	30	ytj	40	vzd	50	oef

Table 3. Exemple de table haute et étroite, décomposée en 5 parties pour occuper plus efficacement l'espace disponible.

```
  \begin{minipage}[t]{0.19\linewidth} % 1ère partie de la page...
     \centering
    \begin{tabular}{l l}
      \hline
      \rowcolor{DarkColor} \textbf{Col1} & \textbf{Col2} \\
      \hline
      1 & hdc \\
      2 & orf \\
        ...
      \hline
    \end{tabular}
  \end{minipage}
  \begin{minipage}[t]{0.19\linewidth} % 2ème partie de la table...
    \centering
    \begin{tabular}{l l}
      \hline
      \rowcolor{DarkColor} \textbf{Col1} & \textbf{Col2} \\
      \hline
      11 & ytr \\
      12 & xcv \\
        ...
      \hline
    \end{tabular}
  \end{minipage}
  \begin{minipage}[t]{0.19\linewidth} % 3ème partie de la table...
    \centering
    \begin{tabular}{l l}
      \hline
      \rowcolor{DarkColor} \textbf{Col1} & \textbf{Col2} \\
      \hline
      21 & ioe \\
      22 & snu \\
        ...
      \hline
    \end{tabular}
  \end{minipage}
   ...
\end{table}
```

Chaque partie de la table doit être placée dans un environnement `minipage`. Ici, le paramètre `0.19\linewidth` indique que la chaque partie de la table occupe 19% de la largeur de la page. Il faut

donc adapter cette valeur en fonction du nombre de parties à afficher, et de leurs largeurs respectives.

Éditeur. La syntaxe LaTeX pour définir des tables n'est pas particulièrement pratique. Pour faciliter leur élaboration, il existe des éditeurs WYSIWYG en ligne permettant de définir une table graphiquement, et de générer le code LaTeX correspondant, qui peut ensuite être copié-collé dans un document. Par exemple LaTeX Complex Table Editor[2] ou Tables Generator[3].

> **Attention :** Si vous utilisez ce type d'éditeur, veillez à adapter le code source produit, de manière à respecter les contraintes indiquées dans ce tutoriel.

Fichier externe. Il est possible de lire le contenu de la table dynamiquement dans un fichier externe. Par exemple, la Table 4 est obtenue à partir du fichier `donnees/test.csv`. Voyez la documentation de la bibliothèque `cvsimple`[4][5] pour plus de détails.

Nom	Prénom	Genre	Âge
Karamazov	Fiodor	H	54
Karamazov	Dimitri	H	28
Karamazov	Ivan	H	24
Karamazov	Alexeï	H	19
Smerdiakov	Pavel	H	N/A
Svietlova	Agrafena	F	22
Verkhovtseva	Katerina	F	N/A

Table 4. Exemple de table obtenue automatiquement à partir du fichier CSV `donnees/test.csv`.

Le code source suivant est un aperçu de celui de la Table 4. Notez que l'on définit manuellement l'en-tête de la table. On indique le nom du fichier CSV (ici : `donnees/test.csv`) ainsi que les colonnes de ce fichier que l'on veut insérer dans la table (ici : `nom`, `prenom`, `genre`, `age`). On n'est pas obligé d'insérer toutes les colonnes du fichier, ni de respecter leur ordre dans le fichier.

```
\begin{table}[htb!]
  ...
  \begin{tabular}{l l l r}
    \hline
    \rowcolor{DarkColor}
    \textbf{Nom} & \textbf{Prénom} & \textbf{Genre} & \textbf{Âge} \\
    \hline
    \csvreader[head to column names, late after line=\\]
      % on indique le nom du fichier CSV ici :
      {donnees/test.csv}{}
      % on indique les noms de colonnes àinsérer dans la table :
      % (cf. le fichier CSV pour voir les correspondances)
      {\nom & \prenom & \genre & \age}
    \hline
  \end{tabular}
  ...
\end{table}
```

Sous-tables. Comme pour les figures (cf. Section 4.1), il est possible d'insérer des sous-tables dans un environnement `table`, grâce à la bibliothèque `subcaption`. On utilise pour cela l'environnement `subtable` de la façon suivante :

2. https://www.latex-tables.com
3. https://www.tablesgenerator.com/
4. https://ctan.org/pkg/csvsimple
5. On ne met d'URL dans le document que pour indiquer un lien vers une ressource non-bibliographique, comme ici (un logiciel).

```
\begin{table}[htb!]
   \begin{subtable}[t]{0.49\textwidth}
      \centering
      \rowcolors{1}{LightHeaderColor}{}
      \begin{tabular}[t]{l l}
         ...
      \end{tabular}
      \caption{Première sous-table}
      \label{tab:subtab1}
   \end{subtable}
   \hfill
   \begin{subtable}[t]{0.49\textwidth}
      \centering
      \rowcolors{1}{LightHeaderColor}{}
      \begin{tabular}[t]{l l}
         ...
      \end{tabular}
      \caption{Seconde sous-table}
      \label{tab:subtab2}
   \end{subtable}
   \caption{Exemple de table contenant des sous-tables.}
   \label{tab:subfigures}
\end{table}
```

Le principe est le même que pour subfigure. Le rendu du code source ci-dessus correspond à la Table 5, qui contient elle-même les Tables 5a et 5b.

Colonne 1	Colonne 2
1	Bla bla
2	Ble ble
3	Bli bli
4	Blo blo

(a) Première sous-table

Colonne 1	Colonne 2
1	Bla bla
2	Ble ble
3	Bli bli
4	Blo blo
5	Blu blu
6	Bly bly

(b) Seconde sous-table

Table 5. Exemple de table contenant deux sous-tables.

4.3 Algorithmes

Un algorithme peut être décrit sous la forme d'un diagramme de type *flowchart* [24], et apparaît alors dans le rapport en tant que *figure*. Il est également possible de le décrire sous la forme de pseudo-code. Il faut utiliser pour cela l'environnement `algorithm2e` basé sur la bibliothèque `algorithm2e` [6] :

```
\begin{algorithm2e}[htb!]
  \DontPrintSemicolon
  \KwData{$x, y$}
  \KwResult{M}

  \BlankLine
  \tcp{Initialisation}
  $z \leftarrow 50$\;

  \BlankLine
```

6. https://ctan.org/pkg/algorithm2e?lang=en

```
  \tcp{Boucle principale}
  \For{$i \leftarrow 1$ to $x$}{
    \While{$y \leq 18$}{
      \For{$u \in V$}{
        \If{$u \neq $v}{
          $a \leftarrow a + 1$\; \label{lne:example}
          $y \leftarrow x + y / 2$\;
        }
        \eIf{$x > y$}{
          $x \leftarrow x \times y$\;
        }{
          $y \leftarrow \sqrt{x + y / 2}$\;
        }
      }
    }

    \BlankLine
    \tcp{Un autre commentaire}
      $B \leftarrow (x + y)^a$\;
  }
  $M \leftarrow ||B||$\;

  \caption{Exemple d'algorithme bidon, représenté sous forme de pseudo-code.}
  \label{alg:exemple}
\end{algorithm2e}
```

Le résultat de ce code source est affiché dans l'Algorithme 1.

Data : x, y
Result : M

// Initialisation
1 $z \leftarrow 50$
// Boucle principale
2 **for** $i \leftarrow 1$ *to* x **do**
3 **while** $y \leq 18$ **do**
4 **for** $u \in V$ **do**
5 **if** $u \neq v$ **then**
6 $a \leftarrow a + 1$
7 $y \leftarrow x + y/2$
8 **end if**
9 **if** $x > y$ **then**
10 $x \leftarrow x \times y$
11 **else**
12 $y \leftarrow \sqrt{x + y/2}$
13 **end if**
14 **end for**
15 **end while**
// Un autre commentaire
16 $B \leftarrow (x + y)^a$
17 **end for**
18 $M \leftarrow ||B||$

Algorithme 1. Exemple d'algorithme bidon, représenté sous forme de pseudo-code.

Positionnement. La position de l'algorithme est spécifiée via l'option `[htb!]` de l'environnement, comme pour les figures et les tables. Par contre, à la différence des figures et des tables, il ne faut pas centrer l'algorithme (donc : pas de commande `\centering` ici).

Mise en forme. La commande `\DontPrintSemicolon` permet de ne pas afficher les points-virgules qui doivent conclure chaque instruction dans le code source LaTeX.

Contenu. Les commandes `\KwData` et `\KwResult` permettent respectivement de définir les entrées et sorties de l'algorithme (notées `Data` et `Result` dans l'Algorithme 1).

La commande `\BlankLine` permet de sauter une ligne dans l'algorithme. La commande `\tcp` prend du texte en paramètre, et l'affiche sous forme de commentaire dans l'algorithme. La fin d'une ligne comportant une simple instruction ou affectation doit être marquée par `\;`.

Les commandes `\For` et `\While` permettent de définir respectivement des boucles *Pour* et *Tant que*. Elles prennent chacune deux paramètres : le premier pour définir la condition de la boucle, et le second pour définir les instructions qui la constituent.

Les commandes `\If` et `\eIf` permettent de définir respectivement un *Si* simple (ie. sans *Sinon*) et un *Si...Sinon*. La première prend 2 paramètres, et la seconde 3. Le premier paramètre est la condition du *Si*, le deuxième contient les instructions exécutées si cette condition est vérifiée, et le troisième (seulement pour `\eIf`) celles exécutées dans le cas contraire (c'est le *Sinon*).

> **Remarque :** Il existe d'innombrables variantes de ces commandes, s'adaptant à une grande variété de situations. Pour en prendre connaissance, référez-vous au manuel du package `algorithm2e`, qui en contient la liste exhaustive.

Notations. Toutes les variables manipulées doivent être mises en forme en utilisant le mode mathématique (cf. Sections 3.4 et 4.4). En particulier, les affectations doivent être notées avec le symbole $\leftarrow$ (commande `\leftarrow` utilisée en mode mathématique).

Vous devez utiliser au maximum des notations *mathématiques* (par opposition à *informatiques*) dans votre pseudo-code, car cela permet de le rendre plus compact et plus lisible. Par exemple, si on veut parcourir tous les éléments d'une liste L avec une variable e, on utilisera en condition de la boucle *Pour* une expression de la forme : **for** $e \in L$ **do**... (commande `\in` pour $\in$).

Choisissez des noms de variables d'une seule lettre (possiblement une lettre grecque) et non pas des expressions à rallonge comme vous le feriez dans un programme informatique. Vous pouvez par contre utiliser des noms longs pour les fonctions, pour des raisons d'intelligibilité.

Légende. La légende de l'algorithme est insérée exactement comme pour les figures et tables, en utilisant la commande `\caption`.

Label. Le label permettant de faire référence à l'algorithme s'insère aussi comme pour les figures et tables, juste après la légende, et grâce à la commande `\label`. Ce label diffère par son préfixe, qui est `alg:` (au lieu de `fig:` pour les figures et `tab:` pour les tables).

Il est aussi possible de définir un label pour certaines lignes en particulier, comme par exemple la ligne 6 dans l'Algorithme 1. On préfixe alors le label par `lne:`, comme dans l'exemple.

Description informelle. Le pseudo-code constitue une description *formelle* de l'algorithme. En plus de cela, il vous faut en donner une description *informelle*, c'est-à-dire expliquer avec du texte comment il fonctionne. Donnez d'abord le principe général, puis entrez dans les détails. Listez et décrivez aussi chaque variable utilisée dans l'algorithme. Ajoutez ensuite un exemple d'application sur des données simples, en déroulant l'algorithme à la main et en expliquant les différentes étapes du traitement.

4.4 Équations

Le système LaTeXest particulièrement bien adapté à la mise en forme de formules mathématiques. Les équations peuvent y être introduites sous deux formes : *en*-ligne et *hors*-ligne.

Une équation en-ligne est définie en utilisant le mode mathématique (cf. Section 3.4), par exemple $y = ax + b$. Ce texte s'intègre à un paragraphe normal, ou même à une table. Cependant, la mise en forme en-ligne n'est pas adaptée à des formules prenant beaucoup de place. De plus, elle ne permet pas de numéroter l'équation, et donc d'y faire référence plus tard. Pour résoudre ces deux problèmes, il faut utiliser la mise en forme hors-ligne.

L'insertion d'une équation numérotée, en mode hors-ligne, se fait grâce à l'environnement `equation` :

```
\begin{equation}
  f(s) = h(s) + \max(g_1(s), g_2(s))
  \label{equ:exemple}
\end{equation}
```

L'équation correspondant à ce code source prend la forme suivante :

$$f(s) = h(s) + \max(g_1(s), g_2(s)) \tag{1}$$

Comme on peut le voir ci-dessus, *hors-ligne* signifie que l'équation est clairement séparée du texte.

Positionnement et légende. À la différence des figures, tables, et algorithmes, une équation n'est pas un élément flottant, donc elle est insérée à l'endroit où l'utilisateur la définit. Par conséquent, il n'y a pas lieu d'utiliser d'option de positionnement (donc pas de `[htb!]`) ni de définir de légende (donc pas de `\caption`). De plus, elle est centrée automatiquement (donc pas de `\centering`).

Mise en forme. Les instructions indiquées ici sont également valables pour la mise en forme des équations en-ligne. LaTeX offre de nombreuses possibilités de mise en forme mathématique, voici les principales. Voyez [19, 20] pour plus de détails.

L'opérateur de multiplication s'obtient avec la commande `\times` (ex. $x \times y$) : interdit d'utiliser l'astérisque `*`. Les mises en indice et en exposant s'obtiennent respectivement avec `_` (*underscore*) et `^`, par exemple : $\Phi = x^2 + x^{a+b} - x_1 - x_{n-1}$. La plupart des fonctions remarquables ont une commande dédiée, par exemple les fonctions trigonométriques (ex. $\cos x$), le logarithme (ex. $\ln x$), minimum et maximum (ex. $\min x$), etc. L'opérateur de sommation s'obtient avec la commande `\sum`, par exemple $\sum_{i=1}^{10} x + y$.

Les matrices ne doivent *pas* être représentées par des tables : le mode mathématique est prévu à cet effet, comme illustré par l'Eq.(2) :

$$M = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \tag{2}$$

Vous devez utiliser l'environnement `pmatrix`, qui fonctionne à peu près comme `tabular` :

```
\begin{equation}
  M =
  \begin{pmatrix}
    a & b & c \\
    d & e & f \\
    g & h & i
  \end{pmatrix}
  \label{equ:matrice}
\end{equation}
```

Label. Le label d'une équation est spécifié comme pour les figures, tables, et algorithmes : en utilisant la commande `\label`. À noter qu'il faut utiliser le préfixe `equ:` pour définir ce label.

Notations. Vos variables et fonctions doivent avoir des noms courts : une lettre et un indice, maximum. Pas de nom du type $mafonction$, mais plutôt f, f' f_1, etc.

Ne mélangez pas les notations informatiques et mathématiques. Par exemple, n'utilisez pas la notation tableau (array) dans une équation : pas de $f = a + tab[12]$.

5 Références

On distingue trois types de références dans un document LaTeX. Les *renvois* (Section 5.1) sont des liens *internes* vers des éléments numérotés du document (figures, tables, algorithmes, équations, etc.). Les références *externes* (Section 5.2) correspondent à des liens vers des documents Web. Enfin, les références *bibliographiques* (Section 5.3) pointent vers des documents listés dans la section bibliographique. Ces deux derniers types de références nous amènent à la notion de plagiat (Section 5.4).

5.1 Renvois

On a vu précédemment que LaTeX permet de définir un label pour marquer un endroit particulier du document, grâce à la commande `\label`.

Insertion du renvoi. Il est ensuite possible de faire référence à ce label, en utilisant la commande `\ref`, qui prend en paramètre le nom du label concerné. Par exemple, pour faire référence au label `abcdef`, on utilisera la commande suivante :

```
\ref{abcdef}
```

Le renvoi, qui est une référence interne au document, apparaît dans le texte sous la forme d'un lien hypertexte, ce qui est pratique pour la navigation. Si le label correspond à un objet numéroté (section, figure, table, algorithme, équation...), alors c'est son numéro qui sera affiché. La troisième colonne de la Table 6 illustre cela.

Type d'élément	Préfixe du label	Exemple de référence	Détails
Section	`sec:`	Section 5	Section 3.5
Figure	`fig:`	Figure 1	Section 4.1
Tableau	`tab:`	Table 6	Section 4.2
Équation	`equ:`	Eq.(2)	Section 4.4
Algorithme	`alg:`	Algorithme 1	Section 4.3
Ligne dans un algorithme	`lne:`	Ligne 6	Section 4.3
Listing	`lst:`	Listing 1	Section 7.1
Fichier texte	`fil:`	Fichier 1	Section 7.2
Console	`con:`	Console 1	Section 7.3

Table 6. Préfixes à utiliser pour définir les noms des labels, et exemples de rendus obtenus.

Nom du label. La convention veut qu'on utilise des préfixes spécifiques dans les noms des labels en fonction du type d'objet référencé. Vous devez utiliser cette convention dans vos rapports (voir la Table 6). La deuxième colonne de la Table 6 rappelle ces conventions.

Intégration au texte. Il est absolument obligatoire que tout élément flottant (figure, table, algorithme...) soit cité *explicitement* dans votre texte. Cette citation doit se faire nécessairement par `\ref`. Donc chacun doit avoir son propre label.

De plus, la nature de l'élément concerné doit être explicitement mentionnée, comme indiqué dans la dernière colonne de la Table 6. Autrement dit, il ne faut pas indiquer simplement "6", mais bien "Table 6". Enfin, on place un tilde ~ (qui représente un espace insécable) entre la nature de l'élément cité et la commande de renvoi elle-même, i.e. ici :

```
Table~\ref{tab:prefixes}
```

Bien sûr, vous ne devez pas vous contenter de citer chaque élément flottant : il doit être suffisamment décrit, et de façon précise. Un élément qui n'est pas mentionné dans le texte ne sert à rien, et on le considérera comme du remplissage lors de l'évaluation de votre rapport.

5.2 Références externes

On crée un hyperlien au moyen de la commande `\url`, qui prend l'URL en paramètre. Par exemple :

```
\url{http://univ-avignon.fr}
```

Ce qui donne le rendu suivant : http://univ-avignon.fr.

Indiquer directement un hyperlien de cette façon n'est approprié que s'il s'agit d'une ressource qui n'est *pas* un document (par exemple un logiciel, ou des données), **et** si elle n'est citée qu'*une seule fois* dans tout le document.

Dans le cas contraire, c'est à dire si la ressource est un document **ou** si elle est citée plusieurs fois dans le document, vous devez définir une référence bibliographique (cf. Section 5.3).

5.3 Bibliographie

Le système LaTeX est couplé à BibTeX[7] afin de gérer automatiquement les références bibliographiques du document. BibTeX facilite grandement la gestion des sources bibliographiques et leur insertion dans le rapport.

Fichier BibTeX. La procédure consiste d'abord à définir un fichier de type BibTeX décrivant chaque entrée bibliographique en détail. Le fichier `bibliographie.bib` fourni avec ce document est un exemple de fichier BibTeX. Il est placé dans le même dossier que le fichier `.tex` à compiler. Chaque entrée doit être associée à une clé unique permettant de l'identifier par la suite. Attention : n'utilisez pas de caractères accentués lorsque vous définissez vos clés.

Vous ne devez pas éditer ce fichier manuellement, car cela va causer des erreurs qui seront difficiles à détecter par la suite. Il vous faut utiliser le logiciel libre Jabref[8], qui est prévu à cet effet. Il est écrit en Java, et est donc compatible avec les principaux systèmes d'exploitation. Il permet d'ajouter/supprimer/éditer des entrées bibliographiques, et d'éditer/générer un fichier au format BibTeX. Il permet aussi de générer les clés en préservant leur unicité.

La commande `\addbibresource`, qui est présente en début de document, permet d'indiquer l'emplacement et le nom du fichier BibTeX à utiliser. Dans le cas du document présent, on a :

```
\addbibresource{bibliographie.bib}
```

Types d'entrées. On trouve différents types d'entrées bibliographiques, dont les principaux sont les suivants :

- `Article` : article publié dans un journal [4, 6] ;
- `Book` : monographie, i.e. livre intégralement rédigé par le même groupe d'auteurs [11, 31] ;
- `InBook` : chapitre dans une monographie [10, 15] ;

7. En réalité, nous utilisons ici BibLaTeX, une version plus récente du compilateur.
8. http://jabref.sourceforge.net/

- `Collection` : recueil composé de chapitres écrits par des auteurs différents, et compilés par des éditeurs [1, 14] ;
- `InCollection` : chapitre dans un recueil [5, 8] ;
- `Electronic` : page Web ou autre ressource disponible exclusivement en ligne [9, 19] ;
- `InProceedings` : article publié dans les actes d'une conférence [12, 18] ;
- `MasterThesis` et `PhDThesis` : mémoire de master ou de doctorat [7, 32] ;
- `TechReport` : travail publié en interne (à une université, entreprise, etc.) ou documentation technique [13, 16].

Les références bibliographiques (numéros entre crochets) associées à chaque type d'entrée listée ci-dessus sont des exemples, à consulter dans le fichier `bibliographie.bib`.

Champs nécessaires. Une entrée bibliographique doit toujours indiquer certaines informations, qui dépendent du type de l'entrée et prennent la forme de champs spécifiques. Ces champs sont détaillés dans la Table 7.

Dans tous les cas, il faut indiquer le DOI [23] s'il y en a un, et *sinon* l'URL pour pouvoir récupérer facilement le document concerné.

Type BibTeX	Description du champ	Champ BibTeX
Tous les types	Nom des auteurs	`author`
	Titre de l'article/ouvrage	`title`
	Année de publication	`year`
	DOI	`doi`
	URL (**seulement** s'il n'y a pas de DOI)	`url`
`@Article`	Nom du journal	`journal`
	Numéro de volume	`volume`
	Numéro de sous-volume	`issue`
	Numéros des pages	`pages`
`@Book/@Collection`	Société publiant le livre	`publisher`
	Ville et pays de cette même société	`address`
	Titre de la série (si le livre appartient à une série)	`series`
	Numéro de volume dans cette série (le cas échéant)	`volume`
`@InBook`	Même champs que pour `@Book`, avec en plus :	
	Titre du livre	`booktitle`
	Numéro de chapitre dans l'ouvrage	`chapter`
	Numéros des pages	`pages`
`@InCollection`	Même champs que pour `@InBook`, avec en plus :	
	Personnes ayant constitué le recueil	`editor`
`@InProceedings`	Même champs que pour `@Book`, avec en plus :	
	Nom de la conférence	`booktitle`
	Numéros des pages	`pages`
`@Electronic`	Entreprise ou institution hébergeant la ressource	`organization`
	URL de la ressource (obligatoire)	`url`
`@Master/@PhDThesis`	Université d'inscription	`institution`
	Type de mémoire (M1, M2, Doctorat, etc.)	`type`
`@TechReport`	Entreprise/université de rattachement	`institution`
	Type de rapport (Technique, Scientifique, etc.)	`type`
	Numéro de série unique du rapport	`number`

Table 7. Champs nécessaires aux différents types d'entrées bibliographiques BibTeX.

Citations. Dans le fichier `.tex`, on fait référence à une entrée bibliographique grâce à la commande `\cite` :

```
\cite{Wikibooks2010}
```

Cette commande prend en paramètre la clé unique associée à l'entrée bibliographique dans le fichier BibTeX correspondant.

Avec la mise en forme choisie pour ce modèle de document, une référence bibliographique prend la forme d'un numéro entre crochet. Pour l'exemple ci-dessus, on a le résultat suivant : [19]. Ce numéro permet de retrouver l'entrée bibliographique dans la liste située à la fin du document, qui est générée automatiquement par LaTeX. De plus, le numéro est un hyperlien permettant d'accéder directement à cette entrée en cliquant dessus.

Il est aussi possible d'utiliser `\textcite` à la place de `\cite`, pour mentionner le nom de l'auteur. Par exemple pour `\textcite{Masuda2016}` on obtient : Masuda et Lambiotte [11]. Vous pouvez mettre plusieurs références dans une même commande `\cite`, par exemple `\cite{Wikibooks2010, Wikibooks2011}` donne [19, 20].

Attention à bien séparer la citation du texte au moyen d'un espace : c'est "une citation [4]", et non pas "une citation[4]".

Section bibliographique. BibTeX et LaTeX vont s'occuper de numéroter les références automatiquement. La commande `\MyBibliography` permet d'insérer une liste des entrées références, généralement on la place à la fin du document (voir le code source de ce document).

5.4 Plagiat

Toutes les informations que vous intégrez à votre rapport et/ou que vous avez utilisées lors de votre travail, mais qui n'ont pas été produites par vous-même, doivent être référencées *explicitement*. Vous devez donc indiquer au moyen de références bibliographiques tout document ou ressource que vous utilisez, y compris ce qui est récupéré sur Internet. Cela vaut pour tout type d'information : document texte, images, vidéos, code source, algorithmes, idées diverses...

Principe général. Vous avez le droit de réutiliser absolument tout ce que vous désirez [9]. Par contre, vous devez à chaque fois indiquer exactement : *ce que* vous réutilisez, *à qui* vous l'avez pris, et *où* vous l'avez trouvé. De plus, vous devez expliquer *pourquoi* vous avez eu besoin de la ressource, expliquer ce que vous avez éventuellement modifié (par exemple s'il s'agit d'un algorithme), et décrire la *nature* de ces modifications.

Texte. Les règles sont plus précises lorsqu'il s'agit de texte :

- Soit vous citez ce texte *verbatim*, c'est à dire que vous l'insérez directement dans votre rapport. Dans ce cas là, il doit s'agir d'une *courte citation* (i.e. quelques lignes), et celle-ci doit être indiquée au moyen de *guillemets* : "...".
- Soit vous *reformulez* l'idée, i.e. vous vous l'appropriez et l'exprimez avec vos mots à vous. Dans ce cas-là, pas besoin de guillemets.

En plus de cela, dans les deux cas, il faut citer la référence bibliographique correspondant au travail réutilisé.

Position institutionnelle. Le non-respect de ces consignes constitue un *plagiat*, ce qui est considéré par Avignon Université comme une *fraude* [2] (au même titre que tricher à un examen, par exemple), et est donc passible de sanctions disciplinaires.

La notion de plagiat est définie dans le règlement intérieur de l'université [3]. L'UQÀM donne des exemples plus précis de cas de plagiat [17]. En particulier (mais pas seulement) :

9. Enfin : sauf si les consignes spécifiques du travail à réaliser l'interdisent.

- Insérer des images, graphiques, tables, données, etc., sans en citer la source ;
- Copier-coller du texte sans l'indiquer avec des guillemets et/ou sans en indiquer la source ;
- Reformuler du texte ou une idée sans en indiquer la source ;
- Traduire un texte sans en indiquer la source dans la langue originale ;
- Faire faire son travail par quelqu'un d'autre ;
- Effectuer collectivement un travail qui est supposé être fait individuellement ;
- Réutiliser un travail (*même le sien*) fait précédemment ou pour une autre UE, sans l'indiquer ;
- Présenter comme le sien un travail effectué par une autre personne, même avec son accord.

6 Diagrammes

Cette classe LaTeX est configurée pour permettre la conception de certains diagrammes spécialisés, au moyen de la bibliothèque `TikZ`[10]. Ceux-ci sont à insérer dans des environnement `figure` (eux-mêmes décrits en Section 4.1). On distingue ici les graphes (Section 6.1), les graphiques (Section 6.4), les diagrammes UML (Section 6.2) et les diagrammes de Gantt (Section 6.3).

6.1 Graphes

Cette section concerne les graphes au sens de la *théorie des graphes* [30], à ne pas confondre avec la notion de *graphique* [29]. On insère généralement ces derniers sous la forme d'image, comme expliqué en Section 4.1.

Afin d'obtenir un rendu uniforme pour tous les groupes, les représentation graphiques des graphes doivent être générées en utilisant directement la bibliothèque `TikZ` comme expliqué dans cette section.

Notations. Un graphe simple est un couple $G = (V,E)$ tel que $V = \{v_1,...,v_n\}$ est l'ensemble des sommets[11] et $E = \{e_1,...,e_m\}$ celui des arêtes[12]. Les nombres de sommets n et d'arêtes m sont respectivement l'ordre et la taille du graphe.

Une arête $e_i \in V^2$ $(1 \leq i \leq m)$ est un couple de *sommets* $e_i = (v_j,v_k)$, avec $v_j,v_k \in V$ $(1 \leq j,k \leq n)$. Une arête est une relation asymétrique, i.e. les deux sommets sont reliés de la même façon, donc (v_j,v_k) est équivalent à (v_k,v_j). Par facilité de manipulation, on supposera que les sommets composant le couple sont placés dans l'ordre lexicographique.

Un graphe orienté contient des arcs à la place des arêtes. À la différence d'une arête, un arc[13] est une relation asymétrique, orientée du sommet v_j vers le sommet v_k. Donc l'arc (v_j,v_k) n'est plus équivalent à (v_k,v_j). Un graphe pondéré est un triplet $G = (V,E,w)$, dans lequel on a une fonction supplémentaire $w : E \to \mathbb{R}_+^*$ qui associe un poids à chaque arrête (ou arc pour un graphe pondéré orienté). Ce *poids* est une valeur réelle strictement positive. On note $w(e_i)$ le poids de l'arête e_i.

Diagramme. Un diagramme TikZ s'insère dans un environnement `figure` (décrit en Section 4.1). Le code source ci-dessous correspond à la Figure 4 :

```
\begin{figure}[htb]
  ...
  \begin{tikzpicture}
    % sommets
    \node[shape=circle,draw=black] (nnnn) at (0,0) {$v_1$};
    \node[shape=circle,draw=black] (node) at (0,3) {$v_2$};
    \node[shape=circle,draw=black] (truc) at (2.5,4) {$v_3$};
    \node[shape=circle,draw=black] (D) at (2.5,1) {$v_4$};
    \node[shape=circle,draw=black] (Zz) at (5,3) {$v_5$} ;
```

10. https://ctan.org/pkg/pgf?lang=en
11. Aussi appelés *nœuds*.
12. Aussi appelées *liens non-orientés*.
13. Aussi appelé *lien orienté*.

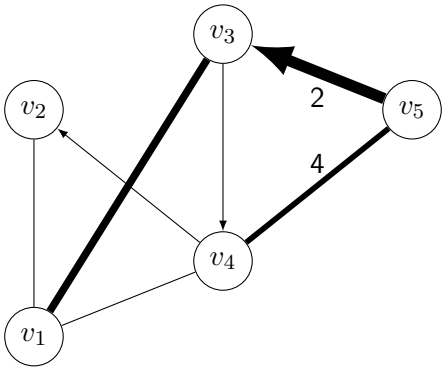


Figure 4. Exemple de représentation d'un graphe, produite en utilisant la bibliothèque TikZ.

```
      % liens
    \draw (nnnn) -- (node); % une arête
    \draw (nnnn) -- (D); % une autre arête
    \draw[line width=1mm] (nnnn) -- (truc); % arête de poids différent
    \draw[line width=0.75mm] (D) -- (Zz) node [midway, left, above] {4};
    \draw[->] (truc) -- (D); % arc
    \draw[-{>[scale=1.5]}] (D) -- (node); % arc avec une flèche plus grande
    \draw[-{>[scale=2]}, line width=1.5mm] (Zz) -- (truc) node [midway, left, below]
        {2};
  \end{tikzpicture}
   ...
\end{figure}
```

Le diagramme est défini au moyen de l'environnement `tikzpicture`. Les sommets sont insérés grâce à la commande `\node[options] (nom) at (x,y) {txt};` (attention au point-virgule !), où :

- `options` est une liste d'options graphiques,
- `nom` est le nom interne du sommet (i.e. dans le cadre de TikZ, un nom utilisé plus tard pour définir les arêtes),
- `(x,y)` est la position du sommet dans le plan,
- `txt` est le texte affiché à l'intérieur du sommet.

Les arêtes sont insérées via de la commande `\draw (s1) -- (s2);`, où `s1` et `s2` sont les *noms internes* deux sommets connectés par l'arête. Pour une arête pondérée, on utilise l'option `line width` `\draw`, qui permet de modifier l'épaisseur. Pour un arc, on utilise l'option `->` qui ajoute une flèche indiquant l'orientation de la relation.

Fichier externe. Le script permettant de définir un graphe peut devenir assez vite long. Sa présence dans le code source du document y rend la navigation plus difficile. Pour éviter cela, il est possible d'externaliser ce code source, c'est à dire de le placer dans un fichier séparé, qui est ensuite importé dans le document principal.

On utilise pour cela la commande `\input`, qui prend en paramètre le nom du fichier à inclure. Supposons qu'on déplace dans un fichier `diagrammes/graphe.tex` la partie du code source de la Figure 4 correspondant à l'environnement `tikzpicture`. Alors, le code source dans le document principal peut être simplifié de la façon suivante :

```
\begin{figure}[htb]
  \centering
   \input{diagrammes/graphe.tex}
   \caption[Exemple de graphe]{Exemple de représentation d'un graphe, produite en
       utilisant la bibliothèque \texttt{TikZ}.}
  \label{fig:graphe}
\end{figure}
```

6.2 Diagrammes UML

Pour les diagrammes UML, l'idéal est d'utiliser un outil externe spécialisé, pouvant produire soit un fichier contenant l'image sous forme de diagramme, soit directement du code `TikZ` à copier-coller dans le rapport.

Fichier. Comme cela a déjà été indiqué en Section 4.1, veillez à sélectionner un outil capable de produire des images vectorielles *vectorielles* [26], par exemple au format PDF. Cela permet au lecteur de zoomer sans perte de qualité. Attention à ne pas enregistrer simplement une image *matricielle* [25] dans un PDF : cela n'aurait aucun intérêt.

De nombreux logiciels gratuits permettent d'éditer des diagrammes et de les enregistrer sous forme vectorielle. C'est notamment le cas du logiciel libre Dia [14], qui est disponible pour la plupart des systèmes d'exploitation.

Une fois l'image exportée, son importation dans le rapport se fait exactement comme expliqué pour les images en Section 4.1.

Code source. Dia permet d'exporter les diagrammes au format PDF, mais il est aussi capable de générer directement du code LaTeX, qui peut ensuite être intégré tel quel dans le document. Pour ce faire, suffit d'aller dans le menu *Fichier* > *Exporter*, et de choisir le format *Macros PGF LaTeX (*.tex)*.

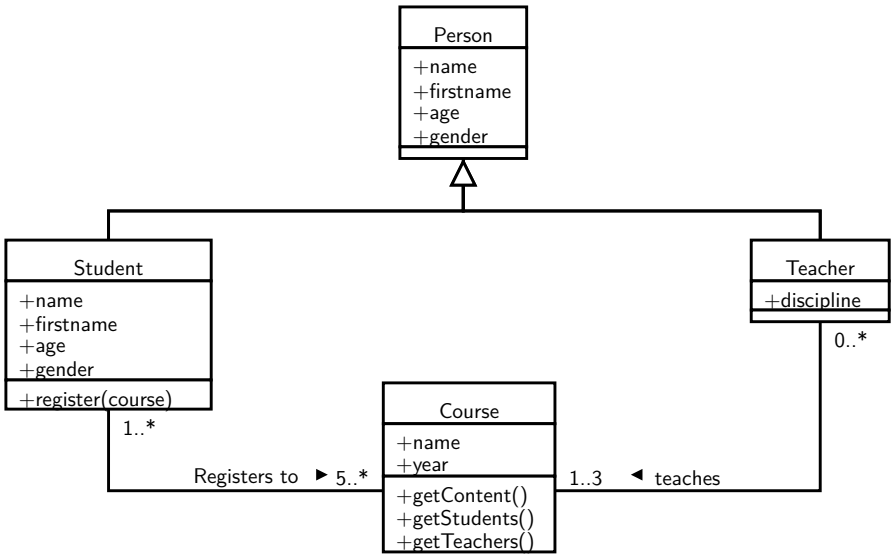


Figure 5. Exemple de diagramme de classes.

L'intérêt de cette approche, par rapport à un PDF, est de pouvoir modifier directement le code LaTeX si on a des ajustement à apporter au diagramme. L'insertion de cette forme de diagramme dans le document se fait par l'environnement `figure`, en utilisant la commande `\import` déjà introduite pour les graphes (Section 6.1) :

```
\begin{figure}[htb!]
  \centering
  \resizebox{0.75\linewidth}{!}{\input{diagrammes/classdiag.tex}}
  \caption{Exemple de diagramme de classes.}
  \label{fig:diag}
\end{figure}
```

Ce code source a pour résultat la Figure 5, qui affiche le fichier `diagrammes/classdiag.tex` produit avec Dia. Notez que la commande `\input` permet d'inclure du code LaTeX situé dans un autre fichier,

14. http://dia-installer.de

quel qu'il soit (i.e. pas uniquement pour un diagramme comme ici).

La commande `\resizebox` permet simplement d'ajuster la taille du diagramme produit. Ici, la valeur 0.75 indique que l'on utilise 75% de la largeur correspondant à une ligne de texte.

6.3 Diagrammes de Gantt

Ce thème est configuré pour permettre l'insertion de diagrammes de Gant basiques tel que celui présenté dans la Figure 6, au moyen de la bibliothèque `pgfgantt` [15]. Si vous trouvez mieux ailleurs, ou si vous avez besoin de plus de fonctionnalités, vous pouvez utiliser un autre outil, y compris externe. Cependant, veillez alors à ce que cet outil génère bien un fichier PDF. Notez que l'avantage de l'approche présentée ci-dessous est que le diagramme est facilement éditable directement dans le document LaTeX.

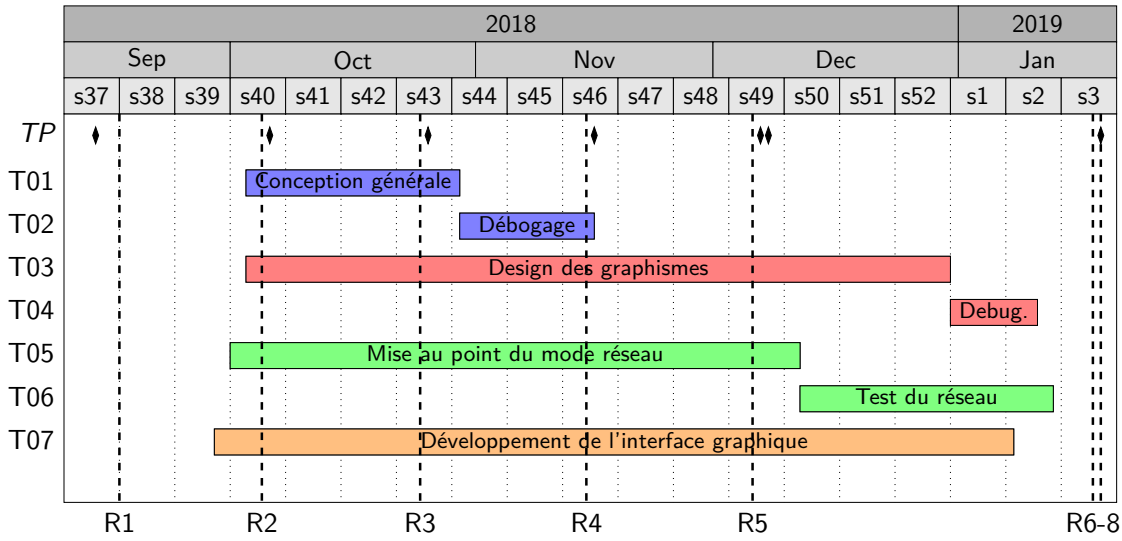


Figure 6. Diagramme de Gantt pour le premier semestre. Chaque couleur correspond à un étudiant : **Prénom1 Nom1**, **Prénom2 Nom2**, **Prénom3 Nom3**, et **Prénom4 Nom4**. Les ♦ représentent les séances de TP, tandis que les lignes verticales correspondent aux dates des rendus.

Pour définir un diagramme de Gantt, vous devez utiliser l'environnement `ganttchart`, lui même inséré dans un environnement `figure`. Il est possible de détailler les commandes décrivant le diagramme soit directement dans le document LaTeX principal, soit dans un fichier `.tex` séparé qu'il faut alors importer avec la commande `\input`, exactement comme nous l'avions fait pour les diagrammes UML en Section 6.2.

Dans le cas de l'exemple de la Figure 6, nous avons opté pour la seconde approche. L'extrait de code source ci-dessous présente les points principaux pour le diagramme contenu dans le fichier `diagrammes/ganttdiag.tex` :

```
\begin{figure}[htb!]
  ...
   \setcounter{myWeekNum}{37} % semaine de départ
   \begin{ganttchart}[...
          x unit=0.11cm, % largeur d'une journée
       ...,
      ]{2018-09-10} % date de début de la période considérée
      {2019-01-20} % date de fin de la période considérée
      % indications temporelles
    \gantttitlecalendar[title/.style={fill=black!30,draw=black}]{year} \\
      ...
```

15. https://ctan.org/pkg/pgfgantt?lang=en

```
      % séances de TP
    \ganttmilestone{TP}{2018-09-13} %TP01
      ...
      % tâches
    \ganttbarbis{T01}{Tâche 01}{2018-10-03}{2018-10-29}{blue!50} \\
      ...
      % dates de rendu
    \drawverticalline{2018-09-16}{R1}
      ...
  \end{ganttchart}
  ...
\end{figure}
```

Remarque : Bien entendu, il est recommandé d'externaliser le code source du diagramme de Gantt, grâce à la commande `\input`, car ce code source est généralement assez long.

Période représentée. La commande `\setcounter{myWeekNum}{xx}` permet d'indiquer le numéro `xx` (dans l'année) de la semaine de départ pour la période représentée.

Les deux derniers paramètres de l'environnement `ganttchart` doivent être utilisés pour indiquer le début et la fin de la période considérée. Dans l'exemple, il s'agit de 2018-09-10 et 2019-01-20.

Jalons. Il est possible de définir des jalons en utilisant la commande `\ganttmilestone`. Elle prend en paramètre un texte optionnel (affiché dans la marge gauche du diagramme) et la date associée au jalon.

Ces jalons prennent la forme de losanges noirs dans le diagramme. Dans l'exemple, on les utilise pour représenter les séances de TP.

Tâches. Les tâches sont ajoutées au moyen de la commande `\ganttbarbis`, et prennent la forme de barres horizontales dans le diagramme. La commande prend en paramètre un nom court (affiché dans la marge gauche), un nom plus long (affiché dans la barre représentant la tâche) et une couleur (utilisée pour colorer la barre). Le nom long doit être ajusté pour tenir dans la largeur de la barre.

Échéances. Les échéances apparaissent sous forme de droites verticales, obtenues au moyen de la commande `\drawverticalline`. Celle-ci prend en paramètre une date permettant de situer l'échéance, et un nom court qui est placé au-dessous de la droite dans le diagramme.

6.4 Figures géométriques

Il est possible de définir des figures géométriques manuellement grâce à TikZ, comme on l'a fait en Section 6.1 pour les graphes. Cependant il est plus pratique d'utiliser un éditeur WYSIWYM puis de l'exporter sous forme de fichier PDF ou de code TikZ. Nous en mentionnons deux ici (sans souci d'exhaustivité).

GeoGebra. C'est ce que permet par exemple GeoGebra [16]. Cet éditeur libre est disponible sous la forme d'une application bureau multiplateforme (à installer sur votre machine), mais aussi d'une application Web (avec des fonctionnalités réduites, néanmoins).

Il permet de composer le graphique ou la figure géométrique de façon intuitive. Il suffit ensuite d'aller dans son menu *Download as* et de sélectionner PDF pour la télécharger sous forme de fichier PDF, ou bien *PGF/TikZ* pour produire automatiquement un fichier contenant le code source TikZ correspondant. Vous pouvez ensuite insérer la figure dans le document LaTeX à partir de l'image (PDF) avec `\includegraphics`, ou à partir du code source avec `\input`.

16. https://www.geogebra.org

À titre d'exemple, la Figure 7 a été générée avec GeoGebra. Elle correspond aussi bien à l'image contenue dans le fichier `images/geogebra.pdf` qu'au code source du fichier `diagrammes/geogebra.tex`.

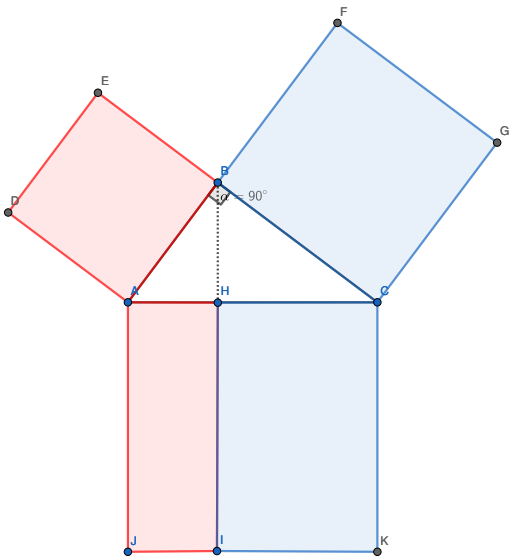


Figure 7. Illustration de la démonstration du Théorème de Pythagore proposée par Euclide [28], générée au moyen de GeoGebra, et correspondant aux fichiers `diagrammes/geogebra.tex` et `images/geogebra.pdf`.

Alternatives. On peut citer des alternative à GeoGebra, notamment l'application Web Matcha [17], qui fonctionne à peu près de la même manière, et l'application bureau multiplateforme et libre Inkscape [18], qui permet également de traiter des diagrammes plus généraux.

7 Contenu à chasse fixe

Cette section est dédiée aux environnements permettant d'insérer du texte en utilisant une *police à chasse fixe* [21] : code source (Section 7.1), contenu d'un fichier texte (Section 7.2), texte affiché dans une console (Section 7.3). La plupart de ces fonctionnalités sont inutiles dans le cadre de cette UE, mais elles sont décrites par souci d'exhaustivité.

7.1 Code source

Il est très déconseillé d'insérer du code source dans un rapport, car cela correspond à un niveau de détail qui est rarement nécessaire, et rallonge inutilement le document. Les éléments relatifs aux points de conception doivent plutôt être décrits en utilisant des diagrammes UML (Section 6.2), tandis que ceux relatifs au traitement sont décrits de façon plus synthétique sous forme de pseudo-code (Section 4.3). La présence de code source répond à un besoin vraiment très spécifique. Avant d'inclure du code source dans votre rapport, réfléchissez donc bien à la nécessité et à la pertinence de cette action. Dans le cadre de cette UE, n'incluez du code source que si cela vous est demandé explicitement.

Quoi qu'il en soit, ce thème permet d'insérer du code source grâce à la bibliothèque `listings` [19] et à son environnement `lstlisting` :

17. https://www.mathcha.io
18. https://inkscape.org/
19. https://ctan.org/pkg/listings?lang=en

```java
// Hello.java
import javax.swing.JApplet;
import java.awt.Graphics;

public class Hello extends JApplet
{  public void paintComponent(Graphics g)
   {  g.drawString("Hello, world!", 65, 95);
   }
}
```

Le code source correspondant au listing précédent est le suivant :

```
\begin{lstlisting}[language=Java]
// Hello.java
import javax.swing.JApplet;
import java.awt.Graphics;

public class Hello extends JApplet
{  public void paintComponent(Graphics g)
   {  g.drawString("Hello, 'world'!", 65, 95);
   }
}
\end{lstlisting}
```

Langage de programmation. L'option `language` de l'environnement `lstlisting` permet de spécifier le langage du code source mentionné (ici : Java).

Positionnement. Remarquez qu'à la différence du pseudocode (Section 4.3), nous ne considérons pas le code source comme un élément flottant, et il n'est par conséquent pas numéroté. La raison en est que l'environnement `lstlisting` autorise que son contenu soit coupé par un changement de page quand cela est nécessaire. Par conséquent, LaTeX le place toujours exactement là où il est défini dans le document, et n'a pas besoin de le déplacer ailleurs comme c'est parfois le cas pour les figures ou autres éléments flottants.

Légende. Il est cependant possible d'avoir besoin de faire un renvoi vers du code source situé ailleurs dans le document. On a alors besoin de définir un numéro et un label pour ce listing. Ceci est possible grâce à aux options `caption` et `label` de l'environnement `lstlisting`, qui fonctionnent comme les commandes de même nom utilisées avec les éléments flottants.

Par exemple, pour rajouter une légende au listing précédent, on fait :

```
\begin{lstlisting}[language=Java,caption={Applet Java affichant le message
    \textit{Hello World !}},label={lst:exemple}]
// Hello.java
import javax.swing.JApplet;
...
\end{lstlisting}
```

Ce qui donne le résultat suivant :

```java
// Hello.java
import javax.swing.JApplet;
...
```

Listing 1. Applet Java affichant le message *Hello World !*

On peut faire référence à ce listing avec la commande `\ref`, comme on le ferait avec un élément flottant : Listing 1.

7.2 Contenu d'un fichier

Cette classe LaTeX propose un environnement spécifique à l'affichage du contenu de fichiers texte. À l'instar du code source, l'utilisation de cet environnement ne sera pas fréquente pour cette UE.

L'environnement s'appelle `filetext`, et il est basé sur la bibliothèque `framed`[20] :

```
Première ligne du fichier texte ;
deuxième ligne, avec de la couleur ici, là, et là ;
et enfin une troisième ligne.
```

Le code source correspondant est le suivant :

```
\begin{filetext}
Première ligne du fichier texte ; \\
deuxième ligne, avec de la couleur \textcolor{red}{ici}, \hl{là}, et
    \colorbox{green}{là} ; \\
et enfin une troisième ligne.
\end{filetext}
```

Notez que les retours à la ligne doivent être indiqués explicitement avec `\\`. Comme illustré dans l'exemple ci-dessus, il est possible de mettre certaines parties du contenu textuel en relief, modifiant la couleur du texte ou de l'arrière-plan.

Positionnement. Comme pour le code source (Section 7.1), cet environnement n'est pas un élément flottant. Il n'est pas numéroté et apparaît là où il est défini. Au cas où on voudrait faire un renvoi vers ce type d'élément plus tard, il est toutefois possible de forcer la définition d'un titre et d'un numéro. On utilise pour cela les options `caption` et `label` de l'environnement, comme c'était déjà le cas pour `lstlisting` :

```
\begin{filetext}[caption={Contenu d'un fichier texte avec
    légende},label={fil:exemple}]
...
\end{filetext}
```

Ce code source produit le Fichier 1.

```
Contenu d'un fichier texte, cette fois avec une légende.
```

Fichier 1. Contenu d'un fichier texte avec légende

7.3 Console

Cette classe LaTeX propose également un environnement dédié à l'affichage d'interactions avec une console système. À l'instar de l'environnement permettant d'afficher le contenu de fichiers textuels (Section 7.2), celui-ci est basé sur la bibliothèque `framed`, et son utilisation sera très rare dans le cadre de cette UE.

L'environnement se nomme `consoletext` et son rendu est le suivant :

```
invite/>macommande monparametre &
macommande: command not found
invite/>■
```

Il fonctionne exactement comme l'environnement `filetext`, seul le rendu visuel est différent :

```
\begin{consoletext}
invite/>macommande monparametre \& \\
macommande: command not found \\
invite/>$\blacksquare$
\end{consoletext}
```

20. https://ctan.org/pkg/framed?lang=en

Il est là aussi possible de forcer l'insertion d'une légende et d'un numéro, comme pour les contenus de fichiers textes (Section 7.2), en procédant exactement de la même façon. Le rendu obtenu est illustré par la Console 1.

```
invite/>
```

Console 1. Invite de commande

8 Conclusion

Il est recommandé de consulter le code source de ce document pour compléter les explications que vous venez de lire.

LaTeX nécessite un minimum d'effort pour être pris en main, mais il n'y a rien d'insurmontable, notamment pour des étudiants en informatique (d'autant plus s'ils sont déjà familiers de langages comme HTML). Le contrôle obtenu et le rendu du document sont largement supérieurs à celui des logiciels de traitement de texte, y compris ceux commerciaux comme MS Word.

À noter qu'un très grand nombre d'extensions existent pour traiter des besoins spécifiques : diagrammes particuliers, documents de présentation de type Powerpoint, etc. Il est aussi possible d'intégrer du code Python ou R (ou autres langages) dans un document LaTeX, qui sera réexécuté à chacune de ses compilations. À vous d'explorer toutes ces possibilités !

Références

[1] U. Brandes et T. Erlebach, éds. *Network Analysis : Methodological Foundations*. T. 3418. Lecture Notes in Computer Science. Springer, 2005. DOI : 10.1007/b106453.

[2] Commission de la formation et de la vie universitaire. *Réglement général des études en licence, master et licence professionnelle – 2018-2019*. Rapp. tech. Avignon : Avignon Université, 2018, p. 1–20. URL : http://univ-avignon.fr/formations/etudes-et-scolarite/modalites-de-controle-des-connaissances/modalites-de-controle-des-connaissances-1458.kjsp.

[3] Conseil d'administration. *Règlement intérieur de l'université*. Rapp. tech. Avignon : Avignon Université, 2016, p. 1–25. URL : https://e-doc.univ-avignon.fr/affaires-juridiques/statuts-et-reglement-interieur-de-luapv/.

[4] J.-V. Cossu, V. Labatut et N. Dugué. « A review of features for the discrimination of Twitter users : application to the prediction of offline influence ». In : *Social Network Analysis and Mining* 6 (2016), p. 1–23. DOI : 10.1007/s13278-016-0329-x.

[5] L. Danon, J. Duch, A. Arenas et A. Díaz-Guilera. « Community structure identification ». In : *Large Scale Structure and Dynamics of Complex Networks : From Information Technology to Finance and Natural Science*. World Scientific, 2007. Chap. 5, p. 93–113. DOI : 10.1142/9789812771681_0006.

[6] S. Fortunato. « Community detection in graphs ». In : *Physics Reports* 486.3-5 (2010), p. 75–174. DOI : 10.1016/j.physrep.2009.11.002.

[7] A. Gerbaud. « Modélisation de réseaux d'interaction par des graphes aléatoires ». PhD Thesis. Grenoble, FR : Université Bordeaux I, 2010. URL : http://www.theses.fr/s123781.

[8] V. Labatut et J.-M. Balasque. « Detection and Interpretation of Communities in Complex Networks : Methods and Practical Application ». In : *Computational Social Networks : Tools, Perspectives and Applications*. Sous la dir. d'A. Abraham et A.-E. Hassanien. Springer, 2012. Chap. 4, p. 81–113. DOI : 10.1007/978-1-4471-4048-1_4.

[9] LaTeX Project. *LaTeX – A document preparation system*. LaTeX Project. 2010. URL : http://www.latex-project.org/.

[10] K. Mainzer. « Complex Systems and the Evolution of Computability ». In : *Thinking in Complexity : The Computional Dynamics of Matter, Mind and Mankind*. 2007. Chap. 5, p. 179–226. DOI : 10.1007/978-3-540-72228-1_5.

[11] N. Masuda et R. Lambiotte. *A Guide to Temporal Networks*. World Scientific, 2016. DOI : 10.1142/9781786341150.

[12] A. Mauttone, M. Labbé et R. Figueiredo. « A Tabu Search approach to solve a network design problem with user-optimal flows ». In : *VALIO/EURO Conference on Combinatorial Optimization*. Buenos Aires, AR, 2008, p. 1–6. URL : https://hal.archives-ouvertes.fr/hal-01882382/.

[13] D. C. Paraskevopoulos, T. Bektas, T. Gabriel Crainic et C. N. Potts. *A Cycle-Based Evolutionary Algorithm for the Fixed-Charge Capacitated Multi-Commodity Network Design Problem*. Rapp. tech. Montréal : Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, 2013, p. 1–31. URL : https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2013-08.pdf.

[14] R. Pastor-Satorras, M. Rubi et A. Diaz-Guilera, éds. *Statistical Mechanics of Complex Networks*. T. 625. Lecture Notes in Physics. Springer, 2003. DOI : 10.1007/b12331.

[15] J. Reichardt. « Modularity of Dense Random Graphs ». In : *Structure in Complex Networks*. T. 766. Lecture Notes in Physics. 2009. Chap. 5, p. 69–86. DOI : 10.1007/978-3-540-87833-9_5.

[16] M. Rosvall et C. T. Bergstrom. *Fast stochastic and recursive search algorithm*. Rapp. tech. Umeå, SE : Department of Physics, Umeå University, 2009. URL : http://www.tp.umu.se/~rosvall/algorithm.pdf.

[17] UQAM. *Éviter le plagiat*. Université du Québec à Montréal. 2019. URL : http://www.infosphere.uqam.ca/rediger-un-travail/eviter-plagiat.

[18] Y.-C. Wei et C.-K. Cheng. « Towards efficient hierarchical designs by ratio cut partitioning ». In : *IEEE International Conference on Computer Aided Design*. 1989, p. 298–301. DOI : 10.1109/ICCAD.1989.76957.

[19] WIKIBOOKS. *Guide pour la programmation en LaTeX*. Wikibooks. 2010. URL : http://fr.wikibooks.org/wiki/LaTeX.

[20] WIKIBOOKS. *Guide to the LaTeX markup language*. Wikibooks. 2011. URL : http://en.wikibooks.org/wiki/LaTeX.

[21] WIKIPEDIA. *Chasse (typographie)*. Wikipedia. 2018. URL : https://fr.wikipedia.org/wiki/Chasse_(typographie).

[22] WIKIPEDIA. *Courier (typeface)*. Wikipedia. 2018. URL : https://en.wikipedia.org/wiki/Courier_(typeface).

[23] WIKIPEDIA. *Digital Object Identifier*. Wikipedia. 2018. URL : https://fr.wikipedia.org/wiki/Digital_Object_Identifier.

[24] WIKIPEDIA. *Flowchart*. Wikipedia. 2011. URL : http://en.wikipedia.org/wiki/Flowchart.

[25] WIKIPEDIA. *Image matricielle*. Wikipedia. 2018. URL : https://fr.wikipedia.org/wiki/Image_matricielle.

[26] WIKIPEDIA. *Image vectorielle*. Wikipedia. 2018. URL : https://fr.wikipedia.org/wiki/Image_vectorielle.

[27] WIKIPEDIA. *LaTeX*. Wikipedia. 2011. URL : http://fr.wikipedia.org/wiki/LaTeX.

[28] WIKIPEDIA. *Pythagorean theorem*. Wikipedia. 2019. URL : https://en.wikipedia.org/wiki/Pythagorean_theorem.

[29] WIKIPEDIA. *Représentation graphique des données*. Wikipedia. 2018. URL : https://fr.wikipedia.org/wiki/Repr%C3%A9sentation_graphique_de_donn%C3%A9es.

[30] WIKIPEDIA. *Théorie des graphes*. Wikipedia. 2018. URL : https://fr.wikipedia.org/wiki/Th%C3%A9orie_des_graphes.

[31] L. A. WOLSEY. *Integer programming*. New York, USA : Wiley-Interscience, 1998. URL : https://www.wiley.com/en-us/Integer+Programming-p-9780471283669.

[32] R. T. WONG. « Accelerating Benders decomposition for network design ». PhD Thesis. Massachusetts Institute of Technology, Department of Electrical Engineering et Computer Science, 1978. URL : https://dspace.mit.edu/handle/1721.1/34293.