# Índice general

# Preparación de la plantilla

Brayan Riveros

La plantilla funciona en Windows, Linux y Overleaf aunque debido a que gran parte del código es parte del diseño no es posible trabajar la plantilla con Overleaf de la mejor manera, por ello tenemos dos opciones la primer forma es trabajar localmente con todas las características la revista y la segunda es mediante una versión sin estilo para Overleaf. Primero veremos lo necesario para trabajar de manera local desde Windows o Linux y posteriormente se explicará como trabajar desde Overleaf o similares.

## 1.1. Programas requeridos

Los programas requeridos para el funcionamiento de la plantilla son los siguientes

1. Distribución de TEX(obligatorio). Para sistemas Unix-like se recomienda TeX-Live full 2017 o posterior, para Windows se debe tener MiKTeX full (si ya lo tiene instalado puede ser de gran ayuda actualizar los paquetes para minimizar los posibles errores que existen entre versiones).
2. IDE (obligatorio). Como IDE se recomienda TeXstudio.
3. Git (obligatorio). Para el control de versiones del artículo y poder usar la plantilla se debe instalar git.
4. Bibliografía (opcional). El programa JabRef para administrar la bibliografía.
5. InkScape (opcional). Para imágenes vectoriales .svg se debe tener instalado este programa y además tener la ruta de instalación en las variables de entorno del sistema.
6. Docker (opcional). Para evitar la instalación individual y las configuraciones correspondientes la imagen compatible con plantilla es

https://hub.docker.com/r/koppor/texlive.

## 1.2. Instalación de plantilla

Primero en creamos una carpeta en nuestro PC personal, allí abrimos la terminal (GIT bash) y digitamos el siguiente comando, para efectos pedagógicos suponemos que la carpeta raíz se denomina CopiaDeTrabajo

```
git init
```

Este comando inicia un Git

Luego en nuestra carpeta git creamos un submodulo de la siguiente forma

```
git submodule add https://github.com/RevistaMathUD/Plantilla10.1
```

Este comando crea el submodulo de la plantilla

Posteriormente abrimos el archivo **base.zip** el cual se encuentra en la ruta

**Plantilla10.1/launch/**

Lo descomprimimos con winrar (Windows) o gestor de archivadores (Ubuntu) el archivo en la carpeta donde iniciamos nuestro git (CopiaDeTrabajo) las carpetas deben quedar como la imagen 1.1. Para continuar con la explicación de la instalación de la plantilla supondremos que nuestro artículo se llama Ecuación de difusión. Ahora seguimos los siguientes pasos

## 1.3. Pasos de configuración

1. Cambie el nombre de la carpeta NAMEARTICLE ubicada en la carpeta **articles** por el nombre de su articulo usando lowerCamelCase sin ningún número (ver apartado 3. Estándares), en este caso se llamará **ecuacionDeDifusion**.

2. Cambie el nombre del archivo BIBLIOGRAPHY.bib ubicado en la carpeta **references** por el nombre del articulo (el mismo dado en el paso anterior, no olvide que el formato debe quedar .bib) este será el archivo de su bibliografia, así que en este ejemplo tendríamos **ecuacionDeDifusion.bib**

3. Abra el archivo **load.sty** (preferiblemente desde TeXstudio) ubicado en la carpeta antigua NAMEARTICLE **(ecuacionDeDifusion)**. Busque y reemplace el texto <carpetaArticulo> por el nombre dado en el paso 1.

4. Abra el archivo **article.tex** ubicado en la carpeta antigua NAMEARTICLE **(ecuacionDeDifusion)** y modifique los datos <autor>, <nombreDelArticulo>, <autorNombreCompleto>, <correoElectronico> y <universidad>.

   **Nota**

   Si el artículo es de varios autores, consulte el apartado 2 (guía de usuario).

5. Configuremos los comandos que se necesitan para poder compilar correctamente, para ello hay dos opciones, la primera es la configuración manual (ver sección 1.5) y la otra es ir Opciones ->Cargar perfil buscamos el archivo **profileGeneral.txsprofile** ubicado en la carpeta **settings**.

6. Compile el archivo **main.tex** ubicado en la carpeta raiz **CopiaDeTrabajo**, es normal que la bibliografía compile con un error en el archivo main.aux, debido paquete **`bibunits`** solo debe compilar los archivos bu*.aux.

   **Notas**

   - Antes de compilar la plantilla, asegúrese de tener la distribución del LaTeX actualizada.
   - Como la plantilla está en producción se necesita una conexión constante a internet.
   - Si esta trabajando en Unix-like o el contenerdor docker (ver sección 1.7) debe agregar la opción a la plantilla, enginedownload=wget. Esto es, en el archivo main.tex debe agregarse la opción

     ```
     \designacademycos{
       root=Plantilla10.1/,
       template=magazineStyleUD,
       enginedownload=wget,
     }
     ```

     Si no funciona, debe instalar wget en su sistema.
   - La bibliografía no compila automáticamente, para ello TeXstudio tiene la opción de ejecutar con F8 o desde

     Herramientas->Bibliografía.

- Para no cambiar constantemente de pestañas y poder compilar en TeXstudio seleccionamos primero la pestaña de nuestro archivo **main.tex** y posteriormente vamos a

  Opciones->Documento Principal->Explícito:main.tex

- Es importante que al redactar el artículo tenga buenas practicas con su código y las condiciones para poder ser revisado, para ello es necesario leer el apartado de estándares 3.

- Como información adicional el motor de compilación que se usará para producir la revista es pdflatex y bibtex.

## 1.4. Estructura de carpetas

Una vez compilado el código y siguiendo nuestro ejemplo tendríamos una estructura de carpetas similar a

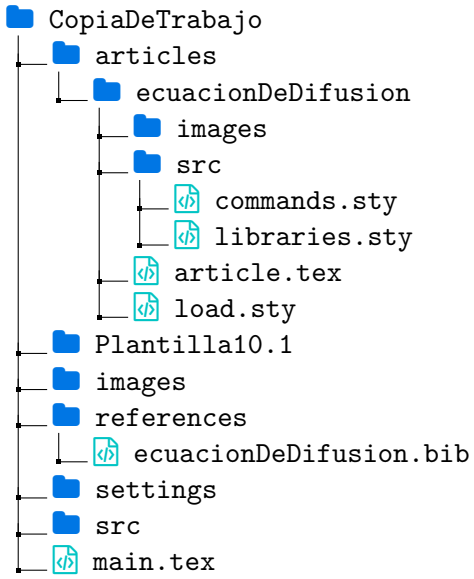



Figura 1.1: Estructura de carpetas del autor

Los únicos archivos que el autor puede modificar son aquellos contenidos en la carpeta de su artículo y el archivo de su bibliografía (claro el **main.tex** se debe modificar solo en la instalación) también tengamos en cuenta que el archivo **load.sty**

no se debe modificar luego de la instalación de la plantilla. El autor debe usar los archivos **commands.sty** y **libraries.sty** para la creación de sus propios comandos y agregar los paquetes necesarios para su artículo respectivamente.

Para tener en cuenta los paquetes que carga la plantilla por defecto y no agregarlos nuevamente a nuestro archivo **libraries.sty** la plantilla de la RevistaUD tiene los paquetes

```
\RequirePackage{import, amsmath, amsfonts, amssymb}
\RequirePackage{yfonts, lmodern}%German fonts, computer modern font
\RequirePackage{ragged2e}%alinear texto
\RequirePackage{forloop}
\RequirePackage{tikz}
\usetikzlibrary{shadows.blur, shapes.symbols, babel, tikzmark}
\RequirePackage{wrapfig} %imagen entre texto
\RequirePackage{float} %posicionamineto de imagenes
\RequirePackage{graphics, graphicx, svg}
\RequirePackage{epigraph}
\RequirePackage{tcolorbox}
\RequirePackage[letterpaper]{geometry}
\RequirePackage[
  colorlinks,
  allcolors = referenceLink,
  pdfpagelabels=true
]{hyperref}
\RequirePackage{pgfopts}
```

Debido que muchas veces encontramos documentos LaTeX muy desordenados recomendamos el uso del paquete **import**, este paquete nos permite llamar archivos desde el código latex.

Por ejemplo, en nuestro archivo **article.tex** necesitamos una grafo en tikz, así que para no colocar el código directamente en el archivo creamos uno nuevo llamado **grafo.tex** y lo guardamos en la carpeta **images/tikz** (se tiene que crear la carpeta tikz) y desde el archivo **article.tex** lo llamamos así

```
\subimport{images/tikz}{grafo.tex}
```

Si es una imagen lo agregamos al entorno **figure** de manera normal, es decir quedaría de esta manera

```
\begin{figure}[h]
  \subimport{images/tikz/}{grafo.tex}
```

```
  \caption{Grafo}
  \label{fig:grafo}
\end{figure}
```

**Nota**

El comando subimport requiere una ruta relativa, para más información consulte su documentación en

https://www.ctan.org/pkg/import

## 1.5. Compilación

Una vez actualizado la distribución del LaTeX (MiKTeX – TeXLive) debemos descargar e instalar TeXstudio (https://www.texstudio.org/)

opciones ->configurar TeXstudio ->pestaña órdenes.

En PdfLaTeX debe tener esta línea de comando

pdflatex.exe –shell-escape -synctex=1 -interaction=nonstopmode %.tex

normalmente solo se debe agregar –shell-escape, también debemos configurar la orden del BibTeX, en la misma pestaña órdenes nos ubicamos en BibTeX debe ir esta línea de comando

bibtex ?me*/*.aux

además debemos verifiquar que en la pestaña Compilar la herramienta bibliográfica por defecto sea BibTeX, con esto la plantilla debe compilar perfectamente con los botones "compilar" y "compilar & ver" recuerde que con F8 compila la bibliografía. En resumen, la compilación debe ser

PdfLaTeX ->BibTeX ->PdfLaTeX

**Nota**

Para mayor velocidad de compilación debemos desmarcar las opciones de repetir las órdenes de compilación contenidas (debemos marcar la opción Mostrar opciones avanzadas) en

Órdenes ->PdfLaTeX
Compilar ->Compilador por defecto.

Aunque recuerda que algunas características deben compilarse dos veces.

## 1.6. Git

El uso de git es requerido, aunque no es obligatorio tener un conocimiento previo de su manejo, así que una vez iniciada la plantilla debemos iniciar a manejar los comandos básicos

1. git add.
2. git commit.
3. git push.
4. git pull.
5. git branch.
6. git merge.
7. git submodule.

Claro es importante tener un repositorio para una copia de seguridad del artículo o poder trabajar de manera colaborativa, así que es necesario contar con una cuenta de github.com.

En los siguientes videos de YouTube encontraremos información de git y su funcionamiento

1. ¿Qué es Git y cómo funciona?
2. Cursos Git - Submodule
3. Curso de Git y Github - 8 Ramas y Uniones (Merge)
4. Git y Github | Curso Práctico de Git y Github Desde Cero

Nota

- Como los videos no son realizados por la revista la fecha de los links en funcionamiento son 22/05/2020.
- Tenga en cuenta que estos videos están orientados para programadores.

## 1.7. Docker

Para usuarios avanzados en temas de programación no debe ser un problema el uso de docker, para ahorrar configuraciones en el archivo **baseArticulo.zip** encontraremos en la carpeta **settings** el archivo de perfil de TeXstudio. Para esto cargamos el archivo **profileDocker.txsprofile** desde

Opciones->Cargar Perfil

**Bug**

Al compilar con docker pdflatex el "synctex" no funciona correctamente en TeXstudio, es decir, la sincronización del código fuente y el archivo PDF no funciona.

Para descargar la imagen requerida (koppor/texlive) vamos a

Herramientas->Usuario->DockerTexliveInstall

Una vez descargada la imagen podemos compilar normalmente desde TeXstudio.

## 1.8. Trabajo colaborativo

Una vez entendido el funcionamiento de git podemos realizar un trabajo colaborativo con git y github, claramente es importante el funcionamiento del comando submodule, pull y push.

Una forma básica de trabajar sería que cada autor trabajara en rama diferente y archivo diferente y al finalizar se hace un merge con la rama master donde en el archivo **article.tex** se llame (paquete import) cada uno de los trabajos realizados por cada autor.

## 1.9. Overleaf

Para trabajar en overleaf es necesario tener en cuenta que no podrá observar el diseño y estilo con el que se publicara los artículos, pero pensando en la comodidad de los usuarios la plantilla tiene una opción que nos permite compilar lo primero que debemos hacer es descargar el archivo .zip a su computador desde

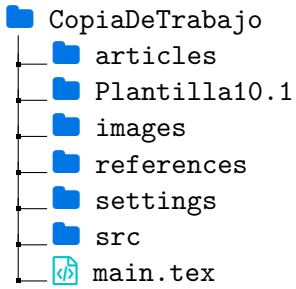
https://github.com/RevistaMathUD/Plantilla10.1

Descomprimimos el archivo .zip en una carpeta de preferencia, a modo de ejemplo la llamaremos CopiaDeTrabajo, y posiblemente la carpeta tendrá el nombre de **Plantilla10.1 - master** por lo que debemos renombrarla como **Plantilla10.1**.

Dentro de la carpeta buscaremos y abrimos el archivo **base.zip** ubicado en

**Plantilla10.1/launch/**

el cual lo descomprimimos en la carpeta **CopiaDeTrabajo**, así que tendríamos la estructura




Posteriormente eliminamos el archivo **base.zip** y finalmente comprimimos en formato .zip todos los documentos de la carpeta **CopiaDeTrabajo** con algún programa como WinRar o similar el cual nombraremos como nuestro artículo usando lowerCamelCase sin ningún número. Por ejemplo

**ecuacionDeDifusion.zip**

Finalmente en Overleaf selecionamos New Project ->Upload Project y subimos el archivo que acabamos de crear, en ese caso el archivo **ecuacionDeDifusion.zip**. Para finalizar seguimos los pasos de la sección 1.3 y verificamos que en el preámbulo exista la opción **platform=overleaf**, es decir, que encontremos un código similar a

```
\datemplate{
  %más opciones
  platform=overleaf,
  %más opciones
}
```

Además debe comentar la lineas donde se requiera archivos como **cover.tex** o archivos que dependan del diseño o estilo de la revista.

# Guía de usuario

Brayan Riveros

## 2.1. Estructura del autor

Una vez realizada la instalación de la plantilla y siguiendo con nuestro ejemplo de la creación del artículo "ecuación de difusión" cuya carpeta es **ecuacionDeDifusion** se tiene las siguientes lineas en el archivo **main.tex**

```
\def\ecuacionDeDifusion{articles/ecuacionDeDifusion}
\def\ecuacionDeDifusionImages{\ecuacionDeDifusion/images}
```

Estos comandos nos sirven para la ruta del artículo y la ruta de las imágenes que usará el autor respectivamente. También consideramos la estructura básica con los siguientes archivos del autor.

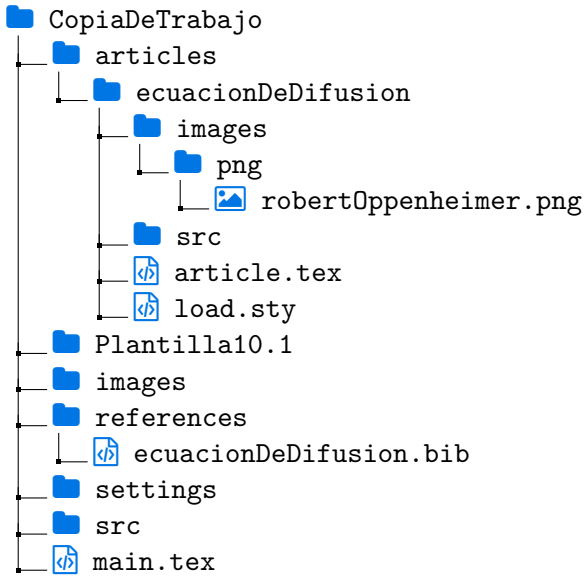



Figura 2.1: Estructura de carpetas del autor

## 2.2. Artículo personalizado

Antes de iniciar la redacción de cualquier artículo entendamos la estructura básica del mismo, el archivo **article.tex** debe ser similar a

```
{\justifying
  \chapterauthor{<Autor>}
  \chapter{<nombre del articulo>}\label{art:<nombre
  ↪  upperCamelCase>}
}\cleanalldata
```

Los siguientes datos son los que se mostrarán en el encabezado del artículo

<autor>= Nombre del autor, como guste el autor.
<nombre del articulo>= Es el nombre del artículo completo.

El comando **\label** es la referencia del artículo y debe ser escrito en lowerCamelCase (ver apartado 3 estándares).

Así que siguiendo con nuestro ejemplo tendríamos algo de esta forma

```
{\justifying
  \chapterauthor{Brayan Riveros}
  \chapter{Ecuación de difusión}\label{art:ecuacionDeDifusion}
}\cleanalldata
```

El primer contenido de cualquier artículo es la ficha técnica del autor y el abstract, para ello disponemos del comando **infoauthorarticle** y los entornos **infoauthorarticlebox**, **abstract**.

Veamos el funcionamiento de cada uno de ellos

- ✔ **Comando infoauthorarticle**. Este comando se usará si el artículo es realizado por un solo autor, donde los argumentos son el nombre completo, correo y universidad, por ejemplo

  ```
  \infoauthorarticle{Brayan Camilo Riveros
  ↪  Castellanos}{camilor2611@gmail.com}{Universidad Distrital
  ↪  Francisco José de Caldas}
  ```

Cuyo resultado lo vemos a continuación

Brayan Camilo Riveros Castellanos
@ camilor2611@gmail.com
Universidad Distrital Francisco José de Caldas

✔ **Entorno infoauthorarticlebox**. Este entorno lo usaremos si el artículo es de varios autores, por ejemplo, un artículo con dos autores

```
\begin{infoauthorarticlebox}
  \faPencil\hspace{5pt} <nombreAutor1>.\\
  \faAt\hspace{9pt}\href{mailto:<email1>}{<email1>}.\\
  \faPencil\hspace{5pt} <nombreAutor2>.\\
  \faAt\hspace{9pt}\href{mailto:<email2>}{<email2>}.\\
  \faUniversity\hspace{2pt} <universidad>.
  \tcblower
  \tcbincludegraphics[transparent, marginsInteriorNull,
  ↪  graphics options={width=2cm}]{png/users.png}
\end{infoauthorarticlebox}
```

Un resultado práctico quedaria como

Brayan Camilo Riveros Catellanos.
@ camilor2611@gmail.com.
Kevin Sebastián Pineda Jaramillo.
@ kevin961312@gmail.com.
Universidad Distrital Francisco José de Caldas.

✔ **Entorno abstract**. Este entorno nos permitirá escribir el resumen de nuestro articulo, cuyo código es

```
\begin{abstract}
  Aquí se escribe el abstract.
\end{abstract}
```

**Nota**

El comando **\cleanalldata** es necesario para la fase de producción de la revista, no se puede eliminar, al igual que el comando **{\justifying}** cuyo funcionamiento es justificar el texto.

### Opcionales

- ✔ **Comando epigraph**. Si queremos agregar un epigrafo agregamos una linea de código similar a la siguiente después de el comando **\chapter**.

```
\epigraph{<frase>}{--- \textup{<autor>}}
```

- ✔ **Comando chapterimage**. Este comando nos permite agregar una imagen representativa en el inicio de nuestro capítulo, para ello debemos usarlo antes de **\chapter**.

```
\chapterimage{<pathImagesCommand>/png/<nombreImagen>}{<alto>cm}
```

Si seguimos con nuestro ejemplo el código quedaría así

```
\chapterimage{\ecuacionDeDifusionImages/png/robertOppenheimer.png}
 ↪  {3cm}
```

El código siguiente es un ejemplo completo de nuestro archivo **article.tex** claro nuevamente siguiendo nuestra suposición del artículo de "ecuación de difusión".

```
{\justifying
  \chapterimage{\ecuacionDeDifusionImages/png/robertOppenheimer.png}
   ↪  {3cm}
  \chapterauthor{Brayan Riveros}
  \chapter{Ecuación de difusión}\label{art:ecuacionDeDifusion}
  \epigraph{Es perfectamente obvio que el mundo entero se va al
   ↪  infierno. La única oportunidad posible es que procuremos que
   ↪  no sea así}
  {--- \textup{Robert Oppenheimer}}
  \infoauthorarticle{Brayan Camilo Riveros
   ↪  Castellanos}{camilor2611@gmail.com}{Universidad Distrital
   ↪  Francisco José de Caldas}
```

```
  \begin{abstract}
    En este artículo expondremos la ecuación de difusión.
  \end{abstract}
  %contenido del artículo
  \section{Sección 1}
  En la sección 1 hablaremos de...
  \putbib
}\cleanalldata
```

Note

Observemos que el comando **\putbib** no ha sido explicado hasta el momento, este nos ayuda con las referencias de nuestro artículo, en la sección 2.3 veremos una explicación más detallada.

## 2.3. Bibliografía

La bibliografía se recomienda trabajarla con JabRef, lo importante de esta parte del artículo son las bibtexkey, ya que el compilador puede encontrar dos iguales y pasarla como advertencia. Para mantener un estándar la bibtexkey debe tener la siguiente estructura

```
bib:<nombre de libro o autor en upperCamelCase>
%----- o de manera alterna......
book:<nombre de libro en upperCamelCase>
```

Por ejemplo la bibtexkey es **bib:ecuacionesDiferenciales** lo podemos citar con el comando típico

```
\cite{bib:ecuacionesDiferenciales}
```

Para mostrar la bibliografía agregamos el comando **\putbib** entre **\justifying**, por ejemplo

```
{\justifying
Contenido de artículo...
\putbib
}\cleanalldata
```

Recuerde que debe compilar la bibliografía, esto se ha explicado en el apartado 1.

**Importante**: en el momento de la producción de la revista se debe unir todos los archivos de la bibliografía de todos los autores que han contribuido, por ello es importante no agregar elementos bibliográficos a nuestro archivo que no se referencien en nuestro artículo.

## 2.4. Entornos

Los siguientes entornos tienen dos argumentos obligatorios, si no se desea utilizar los argumentos se deben dejar en blanco, esto es {}{}. El primer argumento es el nombre del entorno y el segundo es la llave (key) de la referencia, vemos los entornos más usuales.

```
\begin{lemma}{Lema de Zorn}{lemaZorn}
  Contenido lema
\end{lemma}
Para referenciar el lema se utiliza \ref{lem:lemaZorn}
```

Ejemplo

**Lema 2.1: Lema de Zorn**

Contenido lema

Según el lema 2.1 obtenemos ...

```
\begin{theorem}{Teorema de pitágoras}{pitagoras}
  Contenido teorema
\end{theorem}
Para referenciar el teorema se utiliza \ref{th:pitagoras}
```

Ejemplo

**Teorema 2.1: Teorema de pitágoras**

Contenido teorema

Según el teorema 2.1 concluimos...

```
\begin{definition}{Dominio entero}{domEntero}
  Contenido definición
\end{definition}
Para referenciar la definición se utiliza \ref{def:domEntero}
```

Ejemplo

**Definición 2.1: Dominio entero**

Contenido definición

Según la definición 2.1 es claro que ...

De manera similar funcionan lo siguientes entornos

```
%-------Corolario-----------
\begin{mycor}{Nombre corolario}{nom}
  Contenido corolario
\end{mycor}
Para referenciar el corolario se utiliza \ref{cor:nom}
%------propiedades----------
\begin{myprop}{números reales}{numReales}
  Contenido propiedades
\end{myprop}
Para referenciar las propiedades se utiliza \ref{prop:numReales}
%---------definición----------
\begin{mydef}{Dominio entero}{domEntero}
  Contenido definición
\end{mydef}
Para referenciar la definición se utiliza \ref{lem:domEntero}
%------proposición-----------
\begin{myproposi}{}{nomProposicion}
  Contenido proposición
\end{myproposi}
Para referenciar la proposición se utiliza
↪  \ref{proposi:nomProposicion}
```

Otros entornos predefinidos son

```
\begin{boxbasic}
  Contenido nota
\end{boxbasic}
%---------caja------------
\begin{boxbasic}[<título>]
  Contenido caja
\end{boxbasic}
%-----------tabla----------
\begin{mytable}[tabularx={X|X|X}, width=10cm]{<título>}
  cell A & Cell B & Cell B\\\hline
  cell C & Cell D & Cell B
```

```
\end{mytable}
%-------demostracion-------
\begin{proof}
  Supongamos que
\end{proof}
```

**Paquetes externos**

Recuerde que también podemos usar los comandos del paquete **wrapfig** que nos permite insertar una imagen alrededor de un texto.

# Estándares

Brayan Riveros

Antes de mencionar de los estándares del código de esta revista, tenga en cuenta que no son obligatorios, pero seguirlo puede evitar trabajo tanto para los productores de la revista como los autores, sin mas preambulo veamos que es la notación Camel Case

Camel case es una notación que nos permite nombrar las variables, funciones o clases en un lenguaje de programación, esta a su vez se divide en dos casos UpperCamelCase y lowerCamelCase.

**UpperCamelCase**. La escritura utilizando UpperCamelCase es iniciar en mayúscula cada palabra que compone el nombre de la variable, sin olvidar que esta no puede tener espacios ni caracteres especiales.

**lowerCamelCase**. La escritura utilizando lowerCamelCase es iniciar en minúscula cada palabra que compone el nombre de la variable, sin olvidar que esta no puede tener espacios ni caracteres especiales.

## 3.1. Comandos

Recuerde que la estructura de un comando es:

```
\newcommand{\<nombre>}[<arguments number>]{<function>}
```

Para nombrar correctamente los comandos se usará la notación lowerCamelCase

### Referencias

Todas las referencias de la mayor parte de los entornos tienen un prefijo, esto conlleva a una buena organización y previene el conflicto de referencias iguales para entornos diferentes.

**Convención**

Algunas referencias no dependen da la plantilla como lo es el caso de los artículos y ecuaciones, por ello se recomienda seguir el siguiente estándar:

- Artículo. `\label{art:<nombre>}`.
- Sección. `\label{sec:<nombre>}`
- Ecuación. `\label{eqn:<nombre>}`.
- Item. `\label{itm:<nombre>}`.
- Imagen. `\label{fig:<nombre>}`.
- Bibliografía. `\label{bib:<nombre>}` o `\label{book:<nombre>}`.
- Tabla. `\label{tab:<nombre>}`.

**Note**

Para el `<nombre>` utilizaremos la estandarización lowerCamelCase.

### Estándar de nombres de archivos

Para los nombres de todos los archivos del autor (incluyendo imágenes y código LaTeX) no se utilizará espacios ni caracteres especiales, entonces seguiremos el estándar lowerCamelCase, por ejemplo

**algebraLineal.tex** o **cuboRubik.png.**

**Note**

Los nombres de los archivos utiliza el idioma en el que esta redactado el artículo.

## 3.2. Sangrado (Indentación)

El contenido de los entornos debe iniciar con una tabulación, así como se muestra en el siguiente código:

```
\begin{<entorno>}
  Contenido del entorno.
\end{<entorno>}
```

De forma similar en las ecuaciones se realizan las respectivas tabulaciones en el símbolo & tal como se observa en el siguiente código:

```
\begin{align*}
  (a+b)c&=(a+b)\cdot c\\
        &=ac+bc.
\end{align*}
```

## 3.3. Condiciones de aceptación

Las condiciones de aceptación son las siguientes

- ✔ Se debe seguir las recomendaciones de código dadas en este documento.
- ✔ El artículo debe tener máximo 10 páginas.
- ✔ Se debe adjuntar todo el contenido de la carpeta root (CopiaDeTrabajo) en un archivo .zip, si tiene su artículo en github debe hacer una clonación del repositorio, es muy importante que en el archivo .zip se encuentre el pdf resultante (main.pdf).