DigitAll

Formación en
Competencias
Digitales

3
Creación de contenidos digitales

DigitAll

Formación en
Competencias
Digitales

Creación de contenidos digitales

Nivel A1

ÍNDICE

DigitAll

Creación de
contenidos digitales

3.1
DESARROLLO DE CONTENIDOS

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

Uso de formatos digitales

Uso de formatos digitales

La creación de contenidos digitales es fundamental para dar a conocer un producto digital. Al crearse un contenido y ser publicado en alguna plataforma digital (redes sociales, webs, etc.), el receptor puede encontrarlo al estar buscando información relacionada. Para garantizar la visibilidad de nuestros contenidos, independientemente de los motores de búsqueda, tendremos que asegurarnos de cumplir con unos estándares de calidad y seleccionar adecuadamente entre formatos de los mismos. Además, la difusión de los contenidos será fundamental.

La clasificación más general de los formatos para la creación de contenidos digitales incluye: **texto, imagen, audio y vídeo**.

El **texto** ha sido uno de los formatos estrella en la comunicación humana, pocos inventos han impactado en el desarrollo de nuestra cultura como lo hizo la imprenta. No obstante, aunque el texto parece siempre el mismo, vamos a poder encontrar textos publicados en diferentes medios digitales, como los blogs o páginas colaborativas como Wikipedia, así como distintos tipos de formatos digitales con los que trabajar. Hemos de tener en cuenta que la compatibilidad entre formatos no es total por lo que podemos perder información al transformar un tipo de archivo en otro. Entre los formatos más comunes nos encontramos:

Formato	TXT	RTF	ODT	DOC(X)	PDF
Texto enriquecido	No	Sí	Sí	Sí	Sí
Páginas	No	Sí	Sí	Sí	Sí
Incluye imágenes, tablas	No	Sí	Sí	Sí	Sí
Se abre comunmente con	Cualquier procesador de texto	Cualquier procesador de texto	Open Office	Microsoft Word	Adobe Acrobat Reader
Situación de uso recomendable	Solo texto no enriquecido. Fácil de abrir en cualquier situación.	Formato simple, pero admite texto enriquecido. Trabajo en varias plataformas.	Formato libre y abierto. Editable desde cualquier medio o plataforma	Más común. DOCX ocupa menos que DOC.	No se permite edición. Estándar para distribuir documentación

En **imagen**, encontramos fotografías o imágenes como contenidos más utilizados. Estas ayudan a hacer un contenido digital mucho más interesante y sugerente, mejorando la experiencia del receptor. Es uno de los recursos más utilizados en las redes sociales y en las páginas web, ofreciendo un apoyo al texto en publicaciones y siendo un reclamo de atención. Podemos encontrar fotografías, ilustraciones, infografías, memes y banners en cualquier rincón de la red. Los formatos más destacados son:

Formato	JPG	GIF	PNG	TIFF	RAW
Calidad	Alta	Media	Alta	Muy alta	Muy alta
Tamaño	Alto	Medio	Medio	Muy alto	Muy alto
Compresión permitida	Muy alta	Media	Alta	Alta	Alta
Pérdidas en la compresión	Sí	No	No	No	No
Número de colores	24 bits color, 8 bits blanco y negro	Hasta 256 colores	24 bits de color	1 a 64 bits	48 bits
Admite fondo transparente	No	Sí	Sí	Sí	No
Permite animaciones	No	Sí	No	No	No
Situación de uso recomendable	Cámaras digitales, imágenes, impresión, intercambio de imágenes	Internet, imágenes de reducido tamaño, logos	Internet, gráficos, iconografía, software	Imágenes de alta calidad, cámaras digitales, escáneres, impresión	Cámaras digitales

El **audio** ofrece un tipo de contenido que puede ser utilizado en el momento que receptor considere más oportuno, lo que aporta un valor añadido a la hora de comunicar. Entre las distintas opciones de audio encontramos el podcast, las entrevistas y los audiolibros. Los formatos más utilizados son:

Formato	FLAC	MP3	WAV	OGG	WMA
Calidad	Alta	Media-Baja	Alta	Media-Baja	Alta-Media
Tamaño	Alto	Medio	Alto	Bajo	Medio
Compresión permitida	No	Sí	No	Sí	Sí
Admite metadatos	Sí	Sí	Sí	Sí	Sí
Situación de uso recomendable	Archivos musicales de alta calidad	Audios para webs o dispositivos portátiles	Audios para webs o dispositivos portátiles	Audios para webs o dispositivos portátiles	Audios para webs o dispositivos portátiles

Para terminar, el **vídeo** resulta un formato que unifica la imagen y el audio, constituyendo uno de los contenidos que mejor funciona en redes sociales y páginas web, pues ayuda a ilustrar un proyecto o a mostrar el lado más humano del mismo. En la red podemos encontrar revisiones de productos, tutoriales, transmisiones en directo, spots o vídeos musicales. Los formatos principales de vídeo se recogen en la siguiente tabla:

Formato	AVI	MKV	MP4	OGG	MPEG
Calidad	Variable	Alta	Media	Alta	Media
Tamaño	Variable	Alto	Medio	Bajo	Medio
Compresión permitida	Sí	Poca	Sí	Sí	Sí
Situación de uso recomendable	Almacenaje de vídeos originales capurados con cámara	Almacenaje de vídeos de alta calidad	Vídeos para webs o dispositivos portátiles	Ideal para vídeos en internet por su calidad-peso	Formato estándar para compresión y uso en internet

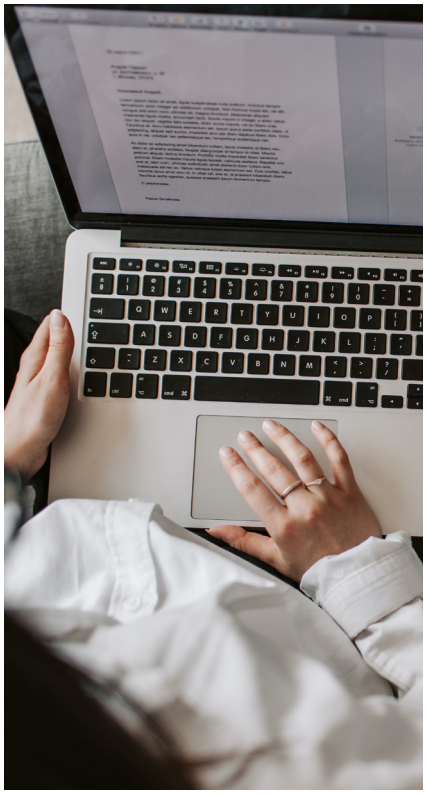
Los contenidos digitales representan herramientas muy poderosas a la hora de transmitir información, con fines docentes, comerciales o de cualquier tipo. La elección de formatos adecuados será fundamental para conseguir impulsar nuestro proyecto y alcanzar así la mayor cantidad de receptores posible.

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

Procesamiento de texto y aplicación de plantillas

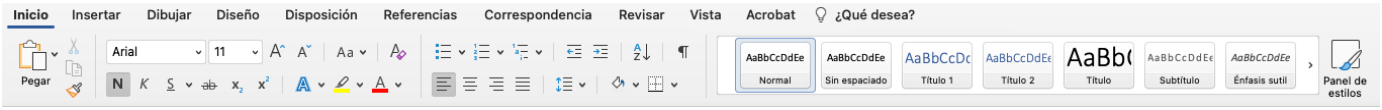
Procesamiento de texto y aplicación de plantillas



En este documento vamos a ir describiendo cuáles son las funcionalidades básicas que nos ofrece un procesador de textos. Resaltaremos el uso de plantillas de documentos proporcionadas por el procesador o la posibilidad de crear plantillas propias que se almacenen.

Las herramientas que vamos a describir en este documento se encuentran en software de fácil acceso, como **Word**, de Microsoft, aunque las podríamos encontrar en cualquier otro procesador de texto. Una de las ventajas más grandes es su capacidad de cambiar el formato del texto y documento en cualquier momento. La barra contiene botones y listas que se despliegan para todas las características que se cambian más a menudo sobre el aspecto del texto.

En la siguiente imagen vemos la barra de herramientas de Word:

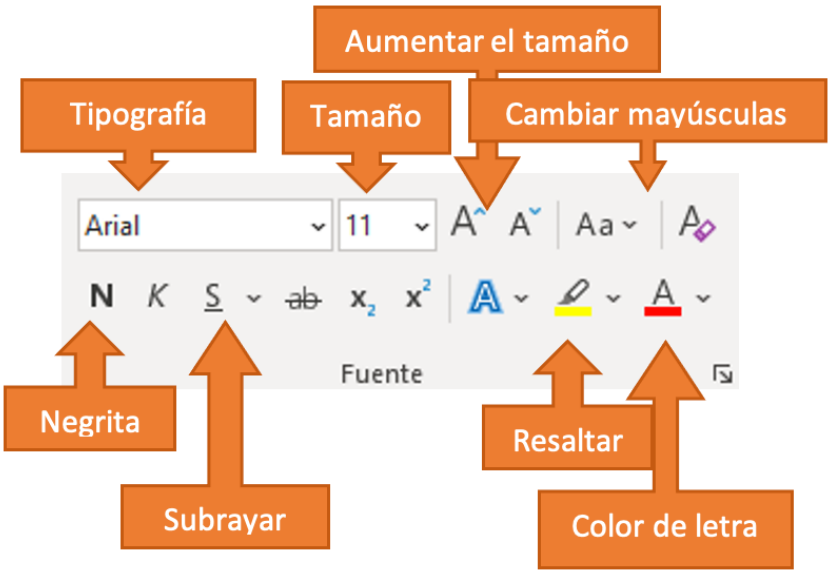


En el siguiente cuadro vemos resumidas las funciones que posteriormente vamos a ir describiendo:

Bloque	Contiene	Permite
Portapapeles	Cortar, copiar, pegar, pegar desde.	Mostrar todos los elementos almacenados.
Fuente	Seleccionar tipografía, tamaño de letra, negrita, cursiva, subrayado, tachado, subíndice, superíndice, sombra, subrayado en colores, cambiar entre mayúsculas o minúsculas.	Personalizar con opciones avanzadas de fuentes y caracteres. Agregar efectos visuales, estilos y colores.
Párrafo	Lista con viñeta, numeración, lista multinivel, sangría, alinear el texto a la izquierda, al centro, a la derecha o justificar, interlineado, sombreado del párrafo, añadir bordes, ordenar una selección alfabéticamente o numéricamente, mostrar las marcas del párrafo para edición avanzada.	Ajustar con precisión el diseño, espaciado, sangría y otros.
Estilos	Aplicar estilos predeterminados, crearlos o eliminarlos.	Administrar y personalizar la apariencia de un texto.

Y ahora vamos a ir viendo con un poco más de detalle para qué se utiliza cada una de estas herramientas siguiendo el orden de la imagen de izquierda a derecha.

- **Portapapeles:** almacena texto y gráficos que copia o corta desde cualquier lugar y permite pegar los elementos almacenados en cualquier otro archivo Office. Primero se debe seleccionar con el ratón aquello que se quiera copiar.
- **Fuente:** podemos modificar la apariencia de las fuentes usando las herramientas de esta parrilla y haciendo clic sobre ellas. Primero deberemos seleccionar el texto que se quiere modificar y luego, marcando las herramientas, modificaremos la tipografía, tamaño, grosor o color.



- **Párrafo:** igual que se puede cambiar el formato de las fuentes, es posible modificar el aspecto de los párrafos. En este caso se puede: sangrar texto, ajustar los márgenes, crear tablas, añadir viñetas y números a listas.

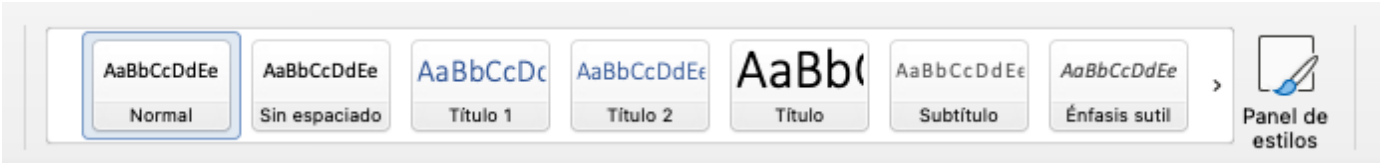
  Para añadir una sangría en la primera línea se podría usar la tecla Tabulador del teclado. No obstante, si el párrafo consta de varias líneas, es pertinente usar la herramienta que define los sangrados, aumentando o disminuyendo la sangría.

Cuando aparecen listas que se quieren numerar o poner una viñeta, se puede usar estas herramientas. Aparecerá un número, o una viñeta, delante de cada uno de los párrafos seleccionados, con los correspondientes sangrados para separar el texto de los números. Cuando el párrafo tiene una extensión superior a una línea, las líneas a partir de la segunda se alinean con la primera que contiene el número.

Con la función del interlineado se puede añadir espaciado vertical entre líneas o párrafos. También se pueden añadir espacios antes y después de los párrafos.

Con estos botones se fija la alineación en distintas posiciones con respecto a los márgenes de la página.

- **Estilos:** tiene una colección de estilos y otras características para el documento. Cada documento tiene una plantilla con estilos predeterminada en la lista desplegada.



Se puede modificar un estilo actualizándolo para que coincida con el formato del documento. En el grupo **Estilos** de la pestaña **Inicio**, se hace clic con el botón derecho en el estilo que se quiere cambiar y se marca Actualizar [nombre de estilo] para que coincida con la selección.

- **Uso de plantillas predefinidas:** al iniciar un documento en Word se puede dar formato con la barra de herramientas al texto o bien usar plantillas predeterminadas. Estas plantillas contienen información y elementos de diseño predefinidos y podremos usar directamente cualquier plantilla que se descargue o se puede personalizar una al gusto y guardarla como plantilla para el futuro. En cualquier caso, estas plantillas van a tener una estética agradable predefinida por lo que nosotros solo nos tendríamos que ocupar de trabajar el contenido de nuestro documento. Además de documentos, tenemos plantillas para calendarios, tarjetas de presentación y mucho más.

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

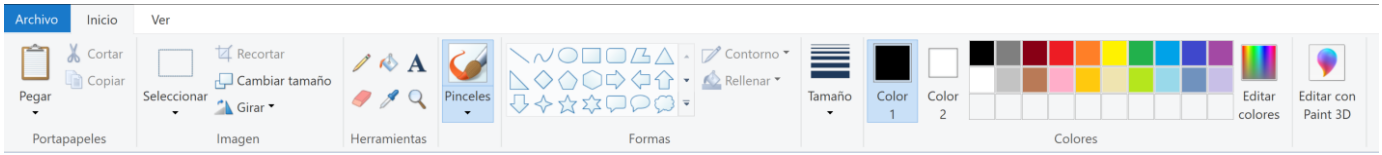
Uso de herramientas básicas de diseño de imágenes

Uso de herramientas básicas de diseño de imágenes

En este documento vamos a ir describiendo cuáles son las herramientas más básicas que nos vamos a encontrar cuando usemos programas de edición de imágenes. Este tipo de herramientas son sencillas de usar y se pueden emplear en multitud de situaciones cotidianas. No obstante, muchas veces no las usamos porque bien ignoramos sus funciones o bien no sabemos siquiera que existen.

Las herramientas que vamos a describir en este documento se encuentran en software de fácil acceso, por ejemplo, **Paint** si estamos usando un PC o bien la herramienta de edición de imágenes a la que podemos acceder desde la galería de nuestro *smartphone*. Cualquiera de estas dos opciones resultará muy intuitiva de usar y muy accesible para tener un primer contacto en el campo de la edición/diseño de imágenes.

En la siguiente imagen vemos la barra de herramientas de Paint. Mediante el uso de estas herramientas vamos a poder ir diseñando una imagen en nuestro **lienzo**, que es el espacio en blanco sobre el que podemos trabajar:



En el siguiente cuadro vemos resumidas las herramientas que posteriormente vamos a ir describiendo:

Bloque	Contiene	Permite
Portapapeles	Cortar, copiar, pegar, pegar desde.	Almacenar información.
Selección de imagen	Seleccionar todo, selección rectangular, selección libre, invertir selección, selección transparente, eliminar, recortar, cambiar tamaño, girar	Seleccionar y trabajar con parte de una imagen.
Edición de imagen	Lápiz, pincel, relleno con color, texto, borrador, cuentagotas, lupa.	Introducir dibujos a mano alzada, texto y colorear.
Formas	Base de datos de formas, contorno, relleno.	Introducir formas predefinidas.
Tamaños	Base de datos de tamaños.	Seleccionar el tamaño del lápiz y el pincel
Colores	Base de datos de colores, color 1, color 2	Seleccionar el color de trabajo

Ten en cuenta que estas son las herramientas más básicas, trabajaremos con herramientas más avanzadas en otro *software* en niveles posteriores. Y ahora vamos a ir viendo con un poco más de detalle para qué valen cada una de estas herramientas siguiendo el orden de la imagen de izquierda a derecha:

- **Portapapeles:** como su nombre indica, es un almacén de texto e imagen que nos va a permitir *cortar* o *copiar* una imagen o parte de una imagen y almacenarla en ese espacio. Después podremos utilizar esa información pegándola donde queramos. También podremos usar la opción *pegar desde* para elegir un archivo de imagen de nuestro equipo y pegarlo en el lienzo.
- **Selección de imagen:** son el grupo de herramientas que nos van a permitir ir seleccionando partes de nuestra imagen. En el punto anterior hablamos de "una parte de una imagen", ahora usaremos estas herramientas de selección para indicar con qué parte de la imagen queremos trabajar. Así podremos *seleccionar* todo o bien solo una parte usando la herramienta de *selección rectangular* (lógicamente, tiene forma de rectángulo) o la de *selección libre*, que nos va a permitir seleccionar un área de cualquier forma. Después podremos *invertir la selección* para seleccionar todo aquello que no hemos

Saber más

Copiar y cortar son dos acciones muy parecidas, pero al cortar una imagen o parte de una imagen la estamos eliminando de nuestro lienzo, mientras que, al copiar, conservamos la información original.

marcado con las herramientas anteriores, *eliminar* la parte del lienzo que hemos seleccionado, o hacer una *selección transparente*, es decir, indicamos que el fondo de nuestra selección es transparente. El resto de las herramientas de este apartado nos van a proporcionar también recursos útiles como *recortar*, para hacer que nuestra selección se convierta en el total de la imagen, *cambiar tamaño*, para proporcionar nuevas dimensiones a nuestra imagen, o *girar*, para inclinar o voltear la nuestra imagen.

- **Herramientas para editar la imagen:** son el grupo de herramientas que nos van a permitir editar una imagen a voluntad. Entre ellas nos encontramos herramientas que nos van a permitir dibujar a mano alzada, como son el *lápiz* o el *pincel*. La ventaja del pincel es que nos va a permitir seleccionar varios formatos de salida cambiando la forma y el tamaño del pincel para adecuarla al trazo que queremos hacer. Podremos pintar más rápido usando el *relleno con color* que, para que nos hagamos una idea, el icono es un cubo de pintura y nos permitirá rellenar nuestro dibujo de un color con un solo clic del ratón. También podremos añadir *texto* a nuestra imagen eligiendo entre una amplia gama de fuentes, tamaños y formatos. Si nos hemos equivocado, podremos corregir usando la herramienta *borrador*. Y, finalmente, tenemos otras dos herramientas que aunque se encuentren en este bloque no son de edición propiamente dicha: se trata de la herramienta *cuentagotas*, que nos va a permitir seleccionar un color de nuestra imagen (para poder, por ejemplo, seguir pintando con el mismo color) y la *lupa*, que nos va a permitir ir aumentando o alejando la imagen según nuestras necesidades.
- **Formas:** es un paquete de herramientas muy útil porque nos va a permitir recurrir a un montón de formas prediseñadas que no tendremos que dibujar a mano alzada y solo tendremos que seleccionarlas para añadirlas a nuestra imagen. Así tenemos desde formas geométricas sencillas hasta dibujos de polígonos libres. Además, podremos elegir el *contorno* de nuestra forma y el *relleno* entre una serie de opciones predefinidas.

⚠ ATENCIÓN

Recuerda que estas formas que introduzcas en el dibujo podrás pintarlas después muy fácilmente con la herramienta relleno con color.

- **Selección de tamaños:** esta herramienta nos va a permitir definir cómo de grueso o de estrecho va a ser el trazo que vamos a hacer con nuestro lápiz o nuestro pincel.
- **Paleta de colores:** finalmente llegamos a la zona de selección del color de trabajo. Aquí tendremos que ir seleccionando con qué color queremos, por ejemplo, pintar con nuestro pincel. Hay que tener en cuenta que siempre podremos volver a la paleta de colores a elegir un nuevo color para algo que ya hayamos coloreado antes y volverlo a colorear de nuevo (tantas veces como queramos). Aquí tenemos varias opciones: lo más obvio es que podemos seleccionar cualquier color de la paleta de colores o bien ir al apartado de *editar colores* para tener una selección mucho más amplia en la que elegir. Además, si hay un par de colores que vamos a usar mucho, podemos seleccionarlos como *color 1* y *color 2*, de forma que nuestra selección se quedará guardado y no tendremos que estar continuamente buscándolos en la paleta.

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

Diseño estructurado de imágenes basado en capas y modificaciones a través de máscaras

Diseño estructurado de imágenes basado en capas y modificaciones a través de máscaras

Al hablar de **máscaras de capa** hemos de tener en cuenta que hablamos de recursos que nos van a permitir editar una imagen de forma no destructiva, es decir, no vamos a alterar la información original de nuestra imagen, siempre podríamos después eliminar una máscara de capa y recuperar nuestra imagen original. Esto mismo es aplicable, como comentaremos más adelante para las máscaras vectoriales.

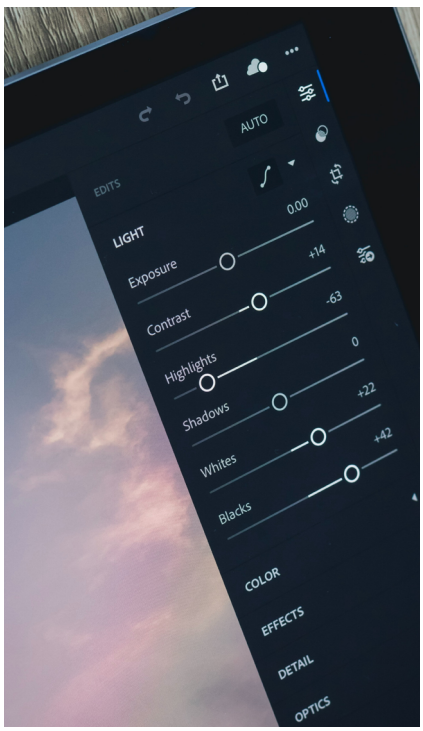
Este tipo de ediciones se pueden llevar a cabo en cualquier programa de edición de imágenes y consisten en añadir capas adicionales sobre nuestra imagen que, en su versión más sencilla, nos van a permitir **mostrar** o **ocultar** partes de una imagen (por ejemplo, un fondo). Por otro lado, las máscaras de capa se pueden aplicar sobre la totalidad de una imagen o sobre una parte muy concreta.

De esta forma, vamos a indicar qué partes de nuestra imagen queremos mostrar u ocultar pintando sobre la máscara de capa, teniendo en cuenta que las máscaras de capa trabajan en **escala de grises**, es decir, solo admiten colores entre el blanco y el negro. Así, podremos pintar la máscara de blanco, y entonces la máscara nos mostrará la totalidad de la imagen (100%) o de negro, y la máscara nos ocultará toda la imagen (0%). Pero también podemos quedarnos en un punto intermedio (gris) en el que la máscara nos va a mostrar la imagen semitransparente, es decir, nos va a ocultar una cierta parte de la información en función del tono de gris que hayamos elegido. Dicho con otras palabras, la máscara de capa nos va a permitir regular su **opacidad**.

Normalmente, los programas de edición nos van a permitir *regular* las propiedades de una máscara y controlar así su opacidad (a veces se denomina también *densidad*) mediante herramientas de tipo deslizador pasando de altos niveles de opacidad/densidad (100% de la capa en negro, todo oculto) a bajos niveles de opacidad/densidad (0% de la capa en negro, se muestra todo).

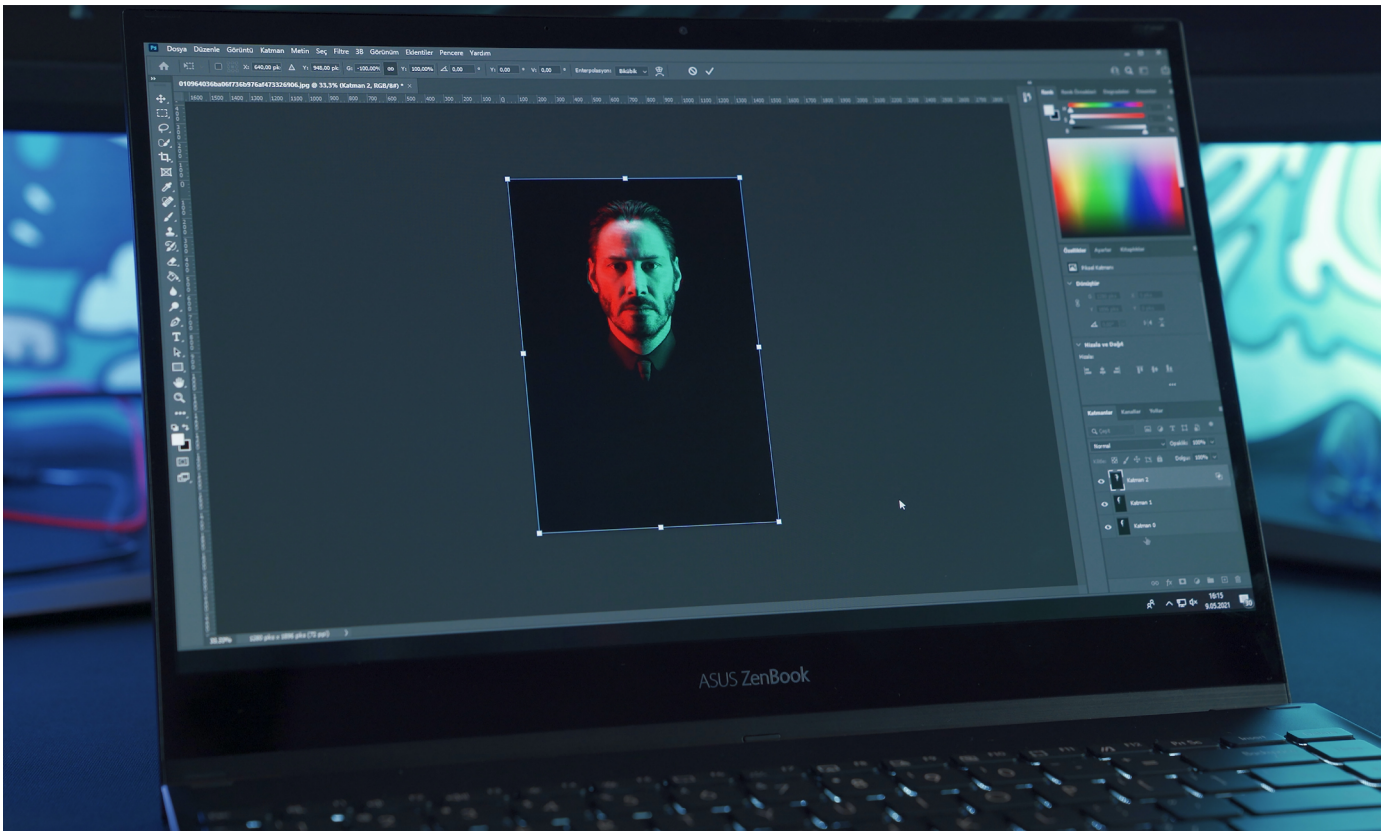
Saber más

Aunque seguramente *Photoshop* sea el software de edición de imágenes más conocido, existen alternativas en el campo del software libre, como ***GIMP*** (*gimp.org.es*), que permiten trabajar prácticamente con las mismas herramientas de edición que los programas de pago profesionales.



No obstante, estas herramientas tienen importantes diferencias con otras que nos proporcionan el mismo resultado. Por ejemplo, si usamos la herramienta *borrador* sobre una imagen, vamos a obtener el mismo efecto que si añadimos una máscara de capa y ocultamos (pintando de negro) la parte que queremos borrar, con la diferencia de que, después de aplicar el borrador, **la información de la imagen se elimina y la perdemos para siempre** mientras que en el caso de la máscara de capa podemos simplemente descartar la capa y recuperar la información que habíamos ocultado a la vista.

Por último, las máscaras de capa son herramientas de edición muy potentes que, además de permitirnos mostrar u ocultar partes de una imagen, también nos van a permitir realizar otras operaciones de edición como la aplicación de *efectos* concretos sobre parte de nuestra imagen, crear fotomontajes o composiciones o simplemente la realización de ajustes de color sobre toda nuestra imagen o parte de ella (por ejemplo, el cielo) y, como hemos comentado anteriormente, estos cambios se producen sin alterar la imagen original, por lo que podemos descartarlos y recuperamos nuestra imagen sin editar.



Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

Imágenes vectoriales vs. imágenes rasterizadas

Imágenes vectoriales VS. imágenes rasterizadas

Las **imágenes rasterizadas** son aquellas que se componen de pequeñas piezas que forman un mosaico. Estas piezas se llaman **píxeles**. A mayor cantidad de píxeles por unidad de área, mayor resolución tendrá la imagen, aumentando su **definición**. También se conocen como **mapas de bits**.

Si una imagen tiene una resolución de 800x1000 píxeles, significa que contiene 800 píxeles horizontales y 1000 verticales. Si observamos esta imagen en una pantalla o en una impresión, lo más probable es que el ojo humano no detecte los píxeles. Sin embargo, si aumentamos el tamaño de la imagen o la imprimimos ampliada, el mosaico se hará evidente.
Sin embargo, a diferencia de las imágenes rasterizadas, los **gráficos vectoriales** no se componen por píxeles, sino por **puntos de control** que determinan ángulos y líneas que forman la figura. Podemos observar las diferencias entre ambos tipos de imágenes en la figura 1.

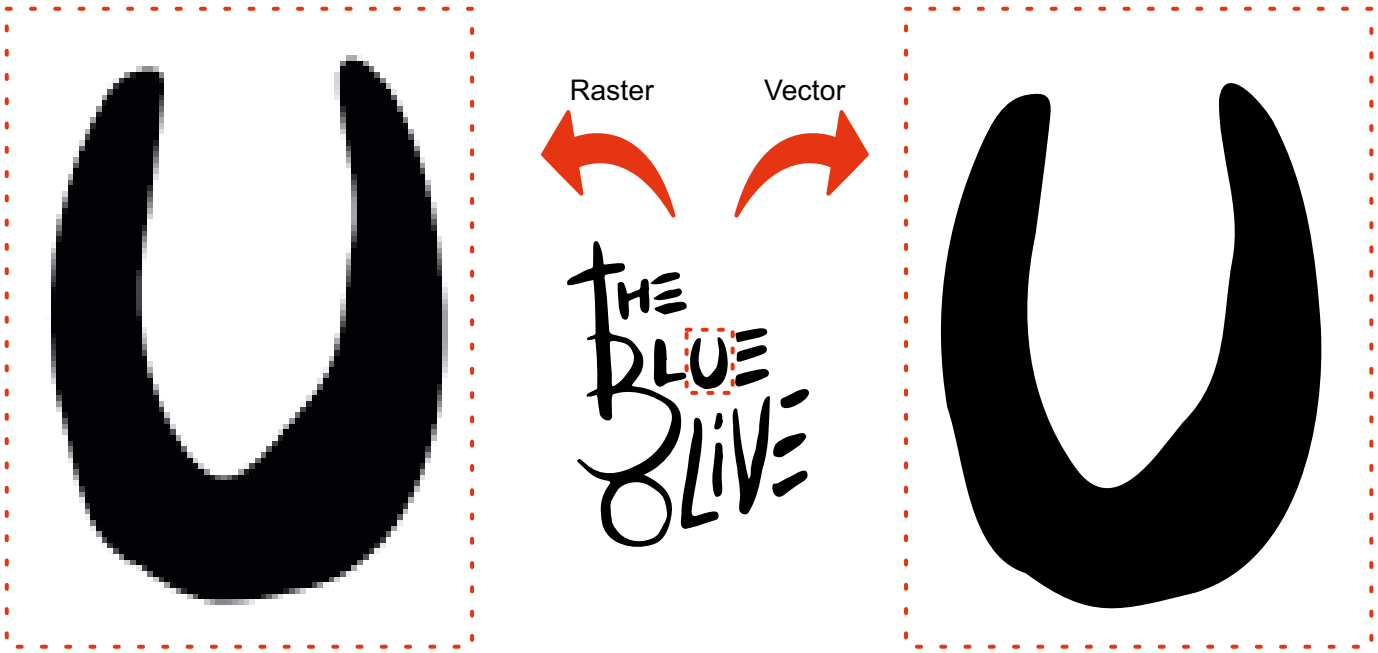


Figura 1. Diferencias entre raster y vector.

Las imágenes rasterizadas son precisas (calidad fotográfica) y ofrecen muchas posibilidades en contextos de creación gráfica y artística, pero suelen ocupar mucho espacio y no es recomendable modificar su tamaño. Sobre todo aumentarlo, ya que supone una pérdida de calidad notable. Las herramientas de software más conocidas para su manejo son *Adobe Photoshop, Gimp* y *Corel Photo-Paint* y sus formatos png, jpg, gif o tif. Por otro lado, las imágenes vectoriales no pierden calidad si son ampliadas o reducidas, escalando automáticamente su tamaño. Una misma imagen vectorial puede servirnos para imprimir una tarjeta de visita o un cartel publicitario enorme y su calidad será óptima. Además, sus archivos no suelen pesar mucho. Por el contrario, a la hora de hacer diseños con ellos, encontraremos algunas limitaciones con respecto a los ráster, y su aspecto no se acercará a la calidad fotográfica. Los programas más utilizados en vectores son *Adobe Ilustrator, Inkscape* y *CorelDraw*, y sus formatos eps, ai, pdf, svg o sketch.

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

Compresión de imágenes

Compresión de imágenes

Algunas veces las imágenes digitales de gran tamaño dificultan el uso de correos electrónicos o sitios web ya que requieren mucho tiempo para ser cargadas y el usuario debe esperar mucho tiempo para obtener la imagen/información. Esto es debido a que las imágenes utilizadas contienen muchos datos o, dicho de otra forma, son de gran tamaño, por lo que ocupan mucho espacio de almacenamiento y son difíciles de transferir o procesar.

La **compresión de imágenes** es la aplicación de procedimientos para reducir la imagen original en unos bytes. El objetivo de este proceso es reducir la redundancia de la imagen y almacenar y transferir los datos de manera eficiente. Es decir, comprimir una imagen es **reducir** los datos redundantes e irrelevantes de la imagen, con **la menor pérdida posible**, para permitir su almacenamiento o transmisión de forma eficiente. Los datos que están duplicados son prescindibles y se guardan en un formato reducido o son eliminados, disminuyendo el tamaño del archivo donde se guarda la información digital. Lo que se busca con este proceso es reducir los bytes de información sin degradar la imagen o persiguiendo una calidad de imagen suficientemente aceptable.

El proceso de compresión puede ser “con pérdidas” o “sin pérdidas”. La **compresión sin pérdidas** reduce una imagen y su calidad coincide con la original, virtualmente no tiene ninguna información perdida. Aunque parece el modo de compresión ideal, muchas veces no resuelve el problema del tamaño y la imagen sigue siendo demasiado grande. Este tipo de reducción de información es usado generalmente en situaciones en las que la calidad de la imagen es más importante que el espacio de almacenamiento o la velocidad de carga de una web.

En la **compresión con pérdidas**, la imagen comprimida no tiene la misma calidad, tamaño o información que la original. Es decir, la imagen se degrada. Lo peor de esto es que la compresión es irreversible y una vez que se ha aplicado a una imagen no puede devolverse al estado original. Si una imagen se comprime repetidas veces se va incrementando la distorsión sucesivamente.

Existen diferentes formatos de compresión y cada uno de ellos usa un algoritmo diferente para producir las imágenes comprimidas. Los más comunes se describen a continuación:

- **BMP** (del inglés "*Bit Map Picture*"), formato de compresión sin pérdidas de Microsoft. Puede guardar 16,7 millones de colores y 256 tonos de gris.

- **PNG** (del inglés "*Portable Network Graphics*"), es un formato que reduce el tamaño mediante identificación de patrones y comprime aquellos que son comunes juntos. La compresión PNG produce archivos de mayor tamaño que JPEG, pero es usada mayoritariamente cuando la calidad de la imagen es lo más relevante, como en webs con logos, iconos, imágenes con texto, dibujos y esquemas.

- **RAW** (del inglés traducido como "crudo"), formato que contiene todos los datos obtenidos por el sensor de la cámara fotográfica, tal y como ha sido tomada la imagen. Para una fotografía profesional, en la que se quiere obtener mucha información (por tanto, un tamaño de imagen grande) es el ideal. Se necesitan programas específicos para su manejo. Es un formato de compresión de datos sin pérdida de información. La mayor dificultad es que cada fabricante de cámaras usa una versión del formato por lo que falta estandarización. Este hecho reduce su utilidad y aumenta las incompatibilidades.

- **TIFF** (del inglés "*Tagged Image File Format*"), es un formato de compresión que puede ser considerado del grupo sin pérdidas. Se utiliza en casos de imágenes con alta resolución y calidad. No es adecuado para webs. Se usa en la industria gráfica.

- **WEBP**, formato de compresión de Google desarrollado solo para su uso en la web. Es un formato muy versátil ya que permite seleccionar entre compresión con o sin pérdidas. Pretende ser un sistema de compresión que compita con JPEG en la compresión de imágenes para webs.

- **GIF** (del inglés "*Graphic Exchange Format*"), formato de compresión que puede ser considerado del grupo de compresión con pérdidas o sin pérdidas. Esta clasificación ambigua depende de la imagen que se quiere reducir. GIF tiene un límite de 256 colores. Si el original tiene 256 colores o menos, la compresión será sin pérdidas. El formato GIF se

clasifica en el grupo de pérdidas si el original tuviese más de 256 colores. Es un formato para vídeos y animaciones simples.

- **JPEG** (del inglés "*Joint Group of Photographic Experts*"), es un formato de uso extensivo en webs y fotografía digital, produce compresión con pérdidas, pero es ampliamente utilizado en numerosas herramientas y aplicaciones. Permite elegir el grado de compresión (y, por tanto, el grado de pérdida).

Formato	BMP	PNG	RAW	TIFF	WEBP	GIF	JPEG
Tipo compresión	Sin pérdida	Sin pérdida	Sin pérdida	Sin pérdida/ opcional con pérdida	Con/sin pérdida	Con/sin pérdida	Con pérdida
Tamaño	Muy grande	Grande	Muy grande	Muy grande	Muy grande	Muy pequeño	Pequeño
Colores	Muy bueno	Muy bueno	Muy bueno	Muy bueno	Muy bueno	Suficiente	Muy bueno
Recomendado para	Fotografías	Imágenes con fondo transparente	Fotografías	Fotografías	Web	Gráficos , ilustraciones, animaciones	Fotografías

Los consejos para dar formato a una imagen fotográfica son:

1 | Si es posible, toma la imagen en formato RAW, quizás debas comprar tarjetas de datos para poder almacenar más información en el dispositivo/cámara.

2 | Guarda el trabajo o la imagen final en TIFF.

3 | Usa JPEG para compartir la imagen original con otros.

4 | No almacenes la imagen en JPEG.

5 | Guarda el trabajo en WEBP para usar las imágenes en páginas web.

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

Creación de video

Creación de video

Todos hicimos de pequeños dibujos en páginas de un cuaderno que al pasarlas rápidamente nuestro dibujo se animaba y adquiría movimiento. Esto, que se llama filoscopio, representa lo que es en realidad el vídeo, pues podemos entender una secuencia de vídeo como una **sucesión de imágenes** que se superponen lo suficientemente rápido para que nuestro cerebro aprecie un movimiento donde no lo hay.

De forma digital, podremos obtener vídeo por varias vías: bien usando esta forma clásica de animación, es decir, a través de imágenes que un programa nos transformará en vídeo (por ejemplo, ***Blender*** (***blender.org***)) o bien grabándolas nosotros mismos.

Para realizar nuestras propias grabaciones de vídeo necesitamos disponer de un **equipo de grabación**. Estos sistemas pueden variar entre sí enormemente en función del resultado final que busquemos, pero todos tienen en común que no solo son capaces de grabar *vídeo*, sino que, además, también nos van a registrar e integrar el *sonido*. Es decir, van a estar compuestos, al menos, de una **cámara** y un **micrófono**.

Lógicamente, estos equipos van a variar enormemente entre sí y de ello depende la calidad del vídeo resultante. En el caso de la imagen, hablaremos de su **resolución**. Así, ya son habituales los teléfonos móviles con los que podemos grabar vídeo, pero estos sistemas están limitados por un lado por las dimensiones del móvil y, por otro lado, por la capacidad de almacenamiento. Es decir, están preparados para grabar vídeos de baja calidad (o baja resolución) ya que, para que los teléfonos sigan siendo pequeños, no pueden llevar sistemas de grabación potentes, pero, si los llevaran, la calidad del vídeo superaría su capacidad de almacenamiento

Por otro lado, sistemas de grabación más profesionales, como las videocámaras o las cámaras réflex, nos van a permitir grabar vídeo de elevada calidad/resolución, ya que disponen de mejores sistemas para capturar vídeo y audio, pero además pueden presentar funcionalidades adicionales como, por ejemplo, *estabilizadores de imagen* para evitar temblores o *sistemas de infrarrojo* para grabar por la noche.

Saber más

La película de culto Pesadilla antes de Navidad (1993) fue creada mediante una técnica conocida como *stop-motion*, que no es más que ir tomando imágenes secuenciales de objetos estáticos para aparentar movimiento.

Después de grabar un vídeo, podemos recurrir a diversos programas que nos van a permitir **editar** el resultado obtenido de forma digital y obtener así un vídeo más acorde con lo que buscamos. Como ocurría durante la grabación, también vamos a encontrar gran variabilidad en los programas de edición de vídeo. En este punto, hemos de tener en cuenta que uno de los factores limitantes en la edición de vídeo es la resolución a la que se ha grabado, por tanto, si un vídeo se ha grabado a alta resolución, siempre podremos reducir su calidad, pero no podremos aumentar su resolución si el vídeo fue grabado a baja resolución.

Así, los editores más sencillos, los que podemos encontrar en teléfonos móviles o en los propios equipos portátiles de grabación, no nos van a dar muchas posibilidades. Por ejemplo, es común que estos editores incorporen un sistema que permita, al menos, *recortar* las partes del vídeo que no nos interesan.

No obstante, si queremos realizar una edición algo más completa, como *variar el ángulo de grabación*, *ajustar* los colores de la imagen, usar *filtros* o *insertar texto*, deberemos hacerla desde un PC, donde podremos recurrir a programas de edición (como ***Kdenlive*** (***kdenlive.org/es***) que nos permitirán prácticamente modificar el vídeo a voluntad.

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

Compresión de video

Compresión de video

A la hora de hablar de vídeo, es conveniente diferenciar entre calidad, resolución y tamaño del mismo, tres conceptos que están muy relacionados entre sí.

La **resolución de un vídeo** determina la cantidad de detalles que tiene, así como su nitidez. Se mide en función del número de píxeles contenidos en la relación de aspecto estándar de 16:9, la más habitual en pantallas, monitores y televisores. La resolución, determina la **calidad** del vídeo, pues cuanto mayor sea la relación de píxeles, mejor se verá el vídeo. En los últimos años, la resolución ha ido aumentando desde los 240p (baja calidad) hasta los 4K, formato de vídeo de alta definición. Independientemente del dispositivo donde se reproduzca el vídeo, cuanto mayor sea su resolución, más nítido se verá. Siempre se puede hacer la imagen más pequeña sin perder calidad, pero hacer las imágenes más grandes se traducirá en pérdidas de calidad. La relación entre las distintas resoluciones se puede observar en la siguiente figura:

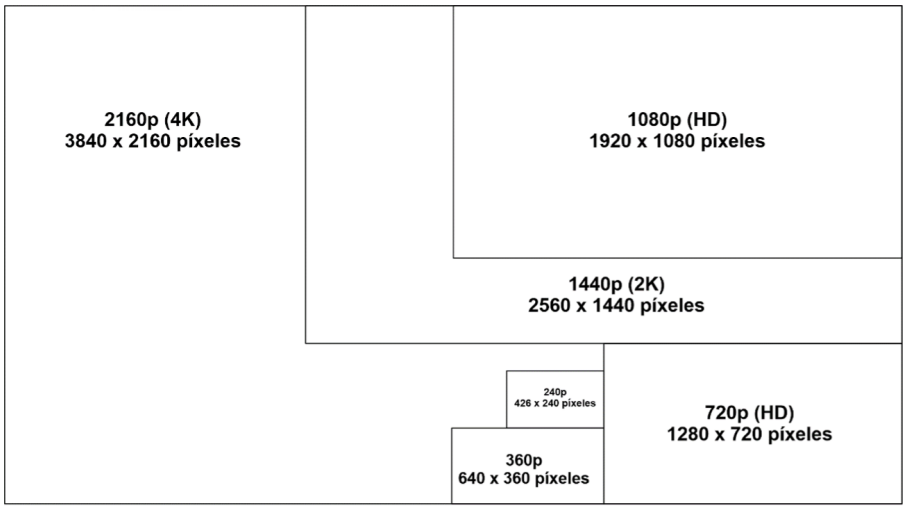


Figura 1. Diferentes resoluciones de vídeo.

Al hablar de **tamaño**, podemos referirnos a la cantidad de píxeles contenidos en el mismo (resolución) o al tamaño en bites que el archivo de vídeo ocupa al guardarlo en el disco duro del ordenador. Entra en juego así el concepto de **compresión**, para conseguir optimizar los archivos de vídeo ocupando la mínima información posible. Pero comprimir conlleva una serie de riesgos en lo que se refiere a calidad del archivo, por lo que habrá que alcanzar un compromiso.

Al aumentar la compresión, la calidad de la imagen disminuye. Una compresión elevada significa que píxeles de colores similares se convierten en un solo color, por lo que la nitidez se pierde y la imagen queda menos detallada. Además, la compresión de las imágenes debe ir compensada en el movimiento temporal, para mantener una tasa de bits (flujo de imagen) que cumpla con los requisitos deseados. La compresión de vídeo permitirá también mejorar la transmisión de datos en *streaming*, ya que la cantidad de datos implicados en un vídeo suele exceder las capacidades de procesamiento de *hardware* actuales. Se consigue así una transmisión más eficiente.

Para poder garantizar que el vídeo sea accesible para el emisor, habrá que elegir el formato de archivo de vídeo correcto. Y esto no es tan sencillo como definir el tipo de formato en el ordenador, sino que habrá que valorar los tipos de codificación (códec), contenedores de archivo de vídeo y tipos de formato de vídeo. El **formato del archivo** jugará un papel determinante, pues un archivo de vídeo no solo incluye imágenes sino también audio y es posible que metadatos (como subtítulos, estructuras de menús o información del archivo). Por lo tanto, en el formato de vídeo se considera por un lado el **códec**, que comprime y descomprime el vídeo, y un **contenedor** que agrupa todo y lo hace funcionar en cualquier software de reproducción.

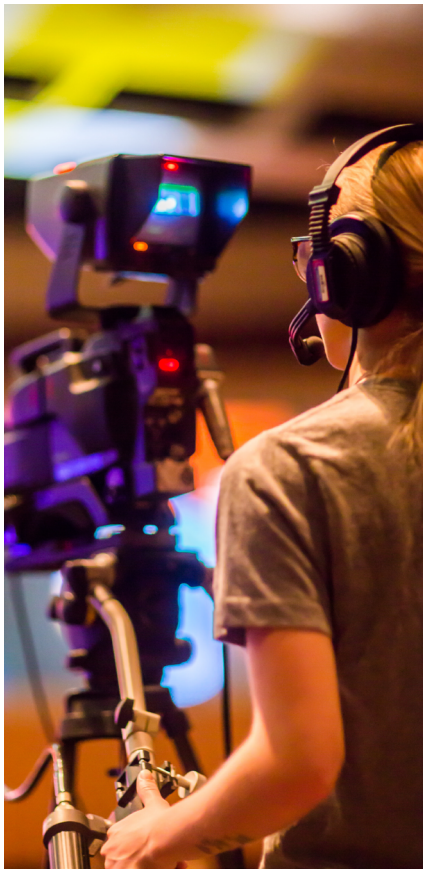
El **códec de vídeo** es un protocolo para codificar y decodificar vídeo. Define el orden que se utiliza para diseñar los datos de un archivo de audio o vídeo, de modo que se pueda reproducir y/o editar. Organiza los datos de los medios que se mantienen en el contenedor. Existen diversos códecs de audio y vídeo, cada uno con sus ventajas e inconvenientes. Los más comunes son el **H.264**, el **MPEG-4** y el **DivX**. Son códecs eficientes con una buena capacidad de preservar la calidad y reducir el tamaño. Los formatos contenedores más utilizados son el **AVI**, el **MP4** y el **MOV**. Se pueden utilizar con distintos códecs en función de los dispositivos y *softwares* donde se vayan a reproducir.

La función última del códec es la de comprimir el vídeo (para que ocupe menos) y descomprimirlo al reproducirlo (para que se vea bien). Y esta compresión puede ser con o sin pérdida. La **compresión con pérdida** permite obtener archivos de menor tamaño, omitiendo datos que dan lugar a archivos de peor

calidad de vídeo. Esto se hace muy evidente en compresiones acumulativas (cuando se comprime algo ya comprimido previamente). La **compresión sin pérdida** mantiene los datos del archivo original, ofreciendo mejor calidad de vídeo y evitando la degradación del archivo. Como desventaja, ocupan un mayor tamaño.

Algunos de los formatos de vídeo más utilizados son:

- **MP4** resulta especialmente útil para compartir contenido en línea. Es el más recomendado para subir vídeos a YouTube con buena calidad. Suele combinarse con el códec H.264 o H.265. Suelen tener tamaños relativamente pequeños y conservar una buena calidad.
- **AVI** es un formato estándar bastante antiguo y universalmente aceptado. Compatible con una enorme variedad de códecs, permite muchas configuraciones de archivo diferentes. Reproducibles en la mayoría de los reproductores, los tamaños de archivo suelen ser grandes, menos recomendables para su transmisión o descarga.
- **MKV** permite almacenar varios canales o pistas de audio y de subtítulos, con muy buena calidad y en muy poco espacio. Además, es un formato abierto por el que no hay que pagar derechos.
- **FLV** es un formato que se hizo muy común debido su tamaño de archivo pequeño y su compatibilidad con la mayoría de los exploradores y reproductores Flash. Su uso ha disminuido en la actualidad.
- **MOV** es un formato de vídeo para Apple, desarrollado para su reproductor Quicktime. Tienen una calidad muy alta y ocupan tamaños de archivo considerables. Presenta menor compatibilidad que los anteriores.
- **WMV** es el formato de Video de Windows Media. Con tamaños pequeños, son una buena opción para envíos, aunque no destaca por ninguna característica especial.



Para cambiar entre formatos existen diversos convertidores. Algunos de ellos son *MiniTool Movie Maker*, *iMovie* (Mac), *Video Converter* (Android), *VLC Media Player* y *Online Videoconverter*.

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

Audio y compresión

Audio y compresión

En el proceso de creación de un contenido audiovisual no cabe duda de que la música, o el audio en general, presenta una relevancia que en la mayoría de las ocasiones supera a la de la propia imagen. El proceso de *storytelling*, o el arte de cómo contar, desarrollar y adaptar una historia en este formato, requiere el uso de las herramientas de audio para conseguir construir un mensaje robusto que impacte al receptor. La música, los efectos de sonido o la narración, son elementos fundamentales para apelar a las emociones del público. Utilizar estos recursos de manera eficaz, facilitarán que el oyente mantenga la atención en nuestro contenido.

Cuando hablamos de compresión de audio, tenemos que especificar que nos referimos a reducir el tamaño del archivo, reducir la tasa de bits de una señal digital de audio volviéndolo más pequeño, conservando casi toda la información original. En función del grado de compresión la pérdida de calidad podrá ser imperceptible o muy notoria.

Será importante controlar el tamaño del audio, para que ocupe menos espacio en los dispositivos y sea, por tanto, más sencillo de transmitir a menor ancho de banda.

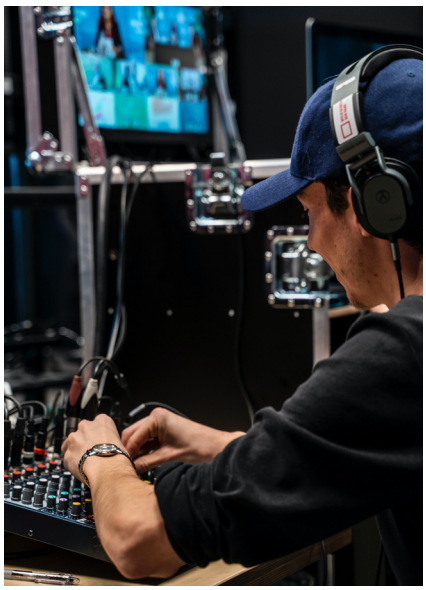
NOTA

En el ámbito de la producción musical, cuando se habla de compresión de audio, no se hace referencia a la reducción del tamaño del archivo, sino al procesado que permite controlar el rango dinámico de una señal de audio. En el contexto musical, la dinámica es la diferencia entre las partes fuertes y suaves de una canción. En el contexto del audio, en lugar de partes fuertes y suaves, existen picos individuales y depresiones de la señal. El rango dinámico será alto si hay mucha diferencia entre estos picos y valles y bajo si la diferencia es pequeña. Y estos niveles pueden regularse con el uso de un compresor.

Existen varios tipos de formatos en función de la compresión del audio. Para hablar de ellos, distinguiremos entre formatos de audio sin comprimir y formatos de audio comprimidos (con y sin pérdida).

NOTA

Prueba a ver un vídeo que te haya impactado, sin audio. Un ejemplo clásico es la ***escena de la ducha de Psicosis*** (e.digitall.org.es/psicosis)(Alfred Hitchcock, 1960) con y sin música.



Formatos de audio sin comprimir

También conocidos como archivos *Hi-Res* o de alta resolución, son aquellos que preservan toda la información del audio original, procesada y almacenada de manera digital. Estos archivos proporcionan la más alta calidad y fidelidad de audio, ocupando grandes cantidades de espacio de almacenamiento. Entre los más habituales encontramos:

- **WAV** (*Waveform Audio Format*), es el formato sin comprimir más habitual y utilizado, tanto por la industria profesional como a nivel de usuario. Suele contener audio sin comprimir en formato PCM, siendo un formato adecuado para Windows (aunque Mac también lo reproduce).
- **AIFF** (*Audio Interchange File Format*), es otro formato sin compresión de Apple que también se puede reproducir en PC. La mayoría de sus archivos contienen audio sin comprimir en formato PCM.

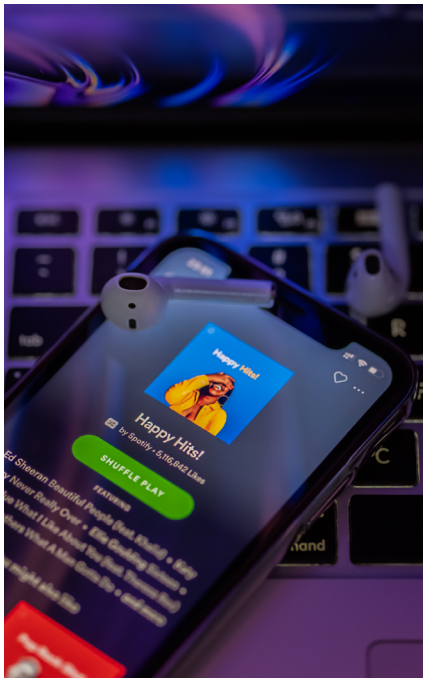
NOTA

El formato PCM es la codificación más utilizada para convertir una onda analógica en una onda digital (secuencia de bits) reproducible en medios digitales.

Formatos de audio comprimido sin pérdidas

Logran comprimir el tamaño de los archivos sin que se pierda calidad general de audio. El nivel de compresión o ahorro de espacio digital no es tan grande como el que nos ofrece la compresión con pérdidas, pero sí que se observa una reducción en comparación con los formatos sin compresión. Los más habituales son:

- **FLAC** (*Free Lossless Audio Codec*), formato de compresión sin pérdida más común en el ámbito musical. Es de código abierto e incluye metadatos incrustados (información del álbum, etc.). Ocupan aproximadamente la mitad del archivo original.
- **ALAC** (*Apple Lossness Audio Codec*), es muy similar a FLAC, pero desarrollado por Apple. También es de código abierto, a pesar de ser de Apple.



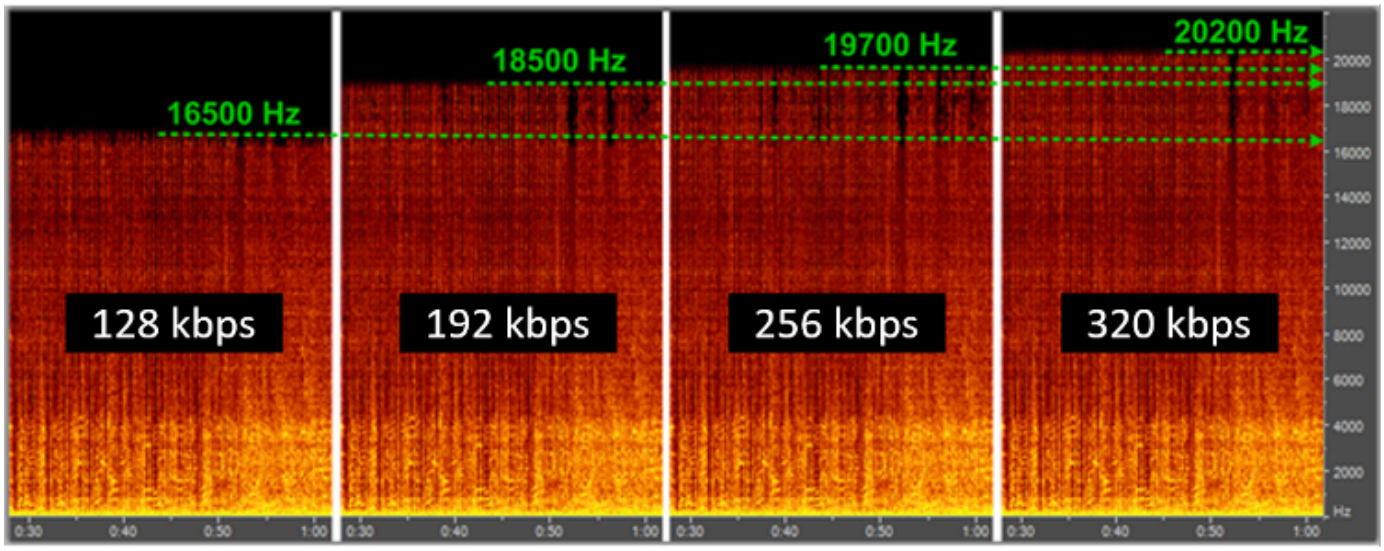
Formatos de audio comprimido con pérdidas

Estos formatos sacrifican la calidad para minimizar el tamaño. Pesan poco y son reproducibles por cualquier dispositivo, pero generan un sonido pobre y sin brillo. No se utilizan en ámbitos profesionales ya que se pierde la fidelidad del sonido. Entre los más habituales podemos encontrar:

- **MP3** (*Moving Picture Experts Group*), es el más popular de los formatos con pérdida. Su algoritmo elimina partes o frecuencias que son fácilmente audibles por el oído humano. Su compresión puede realizarse a distintos niveles de kbps (kilo bites por segundo), siendo el de mayor calidad el de 320 kbps.
- **AAC** (*Advanced Audio Coding*), también conocido como MPEG-4. Es el formato comprimido con pérdidas utilizado por *iTunes* en descargas y por *YouTube* para gestionar el *streaming* de su audio.
- **OGG Vorbis**, es un formato comprimido con pérdidas de código abierto muy utilizado en plataformas de *streaming* como *Spotify* para ahorrar ancho de banda (para que te hagas una idea, la versión gratuita de Spotify reproduce a 128 kbps, mientras que la versión premium lo hace a 320 kbps).
- **WMA** (*Windows Media Audio*), formato creado por Microsoft. Al igual que los dos anteriores, pretende abordar algunos de los fallos del método de compresión del MP3 con un enfoque similar. Aunque en términos objetivos WMA presenta una compresión de más calidad que MP3, es un formato no admitido por muchos dispositivos. Tampoco ofrece ningún beneficio sobre ACC u OGG.

¿Qué formato de audio puede ser el más adecuado?

Esto dependerá de para qué necesitemos nuestro audio. En cualquier caso, si estamos capturando y editando audio sin procesar, la mejor opción será que utilicemos un formato sin comprimir. Así nos aseguraremos estar trabajando con la mejor calidad de audio posible. Al terminar, siempre podremos comprimir (pero esto no podemos hacerlo al revés). Si nos interesa conservar una representación de audio fiel, será conveniente utilizar compresión de audio sin pérdidas. Esta es la razón por la que los melómanos prefieren el formato FLAC por encima del MP3, a pesar de que requiere disponer de más espacio de almacenamiento. Por último, si el archivo no contiene música, es necesario ahorrar espacio y la calidad es un parámetro sacrificable, se puede recurrir a la compresión de audio con pérdida. La realidad es que mayoría de las personas no logra distinguir entre compresión con y sin pérdida. Es extremadamente difícil percibir diferencias, escuchando con equipos o dispositivos de calidad, ya que las compresiones suelen estar al límite de la audición humana. Sin embargo, con equipos de alta gama o auriculares profesionales y con un oído algo educado, se pueden discernir diferencias en matices del audio. Y, desde luego, midiendo con un espectrómetro las longitudes de onda, observaremos diferencias notables.



Ejemplo de compresión de audio. Se observa cómo, de izquierda a derecha, en un archivo en MP3, el extremo de frecuencias agudas (medido en Hz) es mayor cuanto menor es el nivel de compresión (mayor cantidad de kbps).

Creación de
contenidos digitales

Nivel A1 3.1 Desarrollo de contenidos

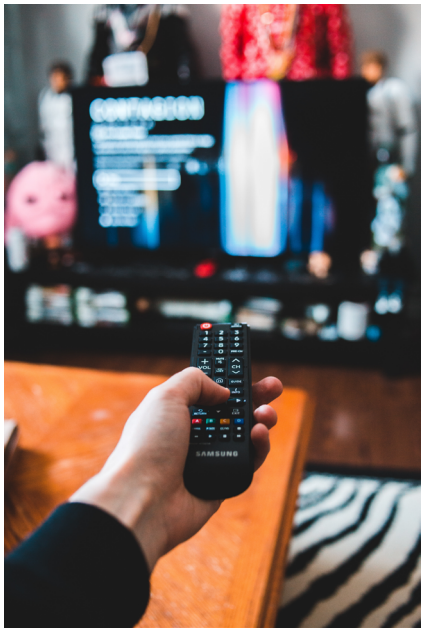
Contenidos digitales en Internet

Contenidos digitales en Internet

Los **contenidos digitales** son fundamentales a la hora de desarrollar cualquier estrategia de comunicación en los tiempos modernos, bien el enfoque sea docente, de marketing o con cualquier otro tipo de objetivo. Sin contenidos digitales, internet sería un erial sin valor ni interés para nadie. Generar contenidos de calidad en diferentes formatos será clave para desarrollar cualquier proyecto, pues será el modo de atraer al receptor, fidelizarlo y aportarle algo. Cualquier material informativo que se pueda incluir en un medio digital, constituye un contenido digital. Estos pueden estar compuestos por texto, imagen, vídeo, audio, aplicación, *software* o cualquier otro medio de interacción que podamos imaginar. Entre los más conocidos, podemos encontrar los siguientes:

- **Blogs personales:** consisten en una plataforma web que permite expresar con personalidad el mensaje del emisor. Es una forma de contactar directamente con el receptor generando interés sobre los contenidos a transmitir de manera creativa, incluyendo material educativo, informativo o relevante para el público objetivo. Habitualmente los blogs cuentan con una sección de comentarios para cada entrada en la que el receptor puede interactuar con el emisor y el resto de los receptores.
- **Redes sociales:** son estructuras formadas por personas y organizaciones conectadas a partir de interés o valores comunes. A través de estas redes se construyen relaciones entre individuos o empresas de manera horizontal, superando las limitaciones físicas. Independientemente de su uso personal o profesional, permiten compartir contenidos de manera sencilla, rápida y gratuita. Será importante definir los contenidos de cada perfil para que el receptor tenga claro que encuentra lo que busca. Las principales redes sociales actualmente son *Facebook, Youtube, Instagram, TikTok, WhatsApp, Linkedin, Twitter, Pinterest, Telegram, Tumblr* y un largo etcétera.
- **Plataformas de vídeo:** los vídeos son herramientas útiles para ilustrar un concepto o resolver dudas, que pueden ayudar a mostrar el lado más creativo o humano de un proyecto. Suponen una forma muy cercana de transmitir un mensaje. A la hora de incluirlos en un correo electrónico,

pueden suponer un problema debido al tamaño que éstos suelen ocupar. Será recomendable utilizar plataformas de *streaming* para subir estos contenidos, e incluir un enlace a los mismos en el correo electrónico. Se puede utilizar *Youtube* o *Vimeo*, dos de las plataformas gratuitas más conocidas que permiten compartir este tipo de material.



- **Plataformas de audio o podcast:** supone una forma cómoda e interesante de desarrollar la presentación y discusión de un tema o evento particular por medio de un archivo de audio, disponible en un archivo para descargar o en *streaming*. Es un formato de baja demanda, ya que el receptor puede escucharlo cuando desee en cualquier dispositivo. Del mismo modo que con el vídeo, puede suponer un problema incluir un archivo de audio como un adjunto en un correo electrónico. Para solventar esta situación, se puede recurrir a *SoundCloud* o *Spreaker*, plataformas que permiten subir archivos de audio en distintos formatos y compartirlos en función de las necesidades. Algunas de las plataformas de gestión de podcast más conocidas son *iVox, SoundCloud, Anchor, Spotify for Podcasters, Google Podcasts, Spreaker, MixCloud* o *Apple Podcasts*, entre muchas otras.

Aprovechar estos formatos permite acortar distancias, conocer nuevos contenidos y apostar por espacios de reflexión y crecimiento personal. El tiempo y el esfuerzo de preparación de los contenidos tendrá que ser optimizado para conseguir crear material de calidad. Con los contenidos digitales se puede conseguir atraer a más público objetivo, mejorar el posicionamiento y la visibilidad de un proyecto, y humanizarlo, dotándolo de autoridad.

DigitAll

Creación de contenidos digitales

3.2 INTEGRACIÓN Y REELABORACIÓN DE CONTENIDO DIGITAL

Creación de
contenidos digitales

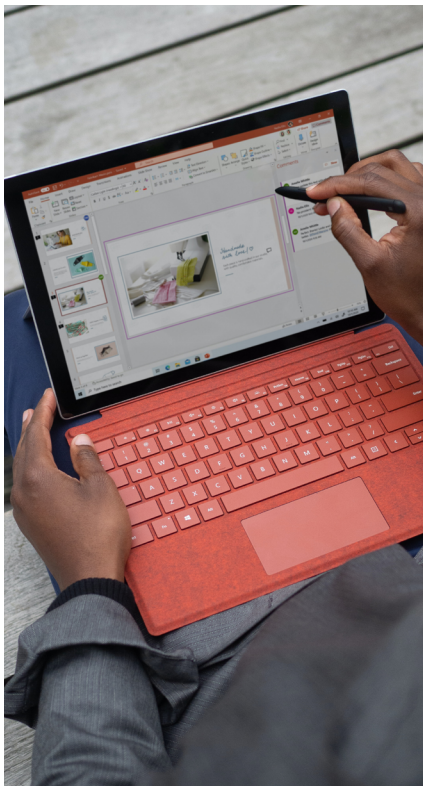
Nivel A1 3.2 Integración y reelaboración de contenido digital

Integración de texto, imágenes, audio y vídeo en presentaciones

Integración de texto, imágenes, audio y vídeo en presentaciones

Las herramientas de creación y edición de presentaciones cuentan con una serie de elementos comunes que el usuario utilizará según una serie de criterios de diseño y guías de estilo para crear sus propias presentaciones. Los más importantes son:

- **Diapositiva (slide):** unidad básica de presentación de la información. Una presentación constará de un conjunto de diapositivas, cada una de las cuales puede incluir información textual (tablas, enlaces a sitios web), gráfica (imágenes y gráficos) e incluso multimedia (audio, imágenes animadas, vídeos).
- **Animación/es:** son efectos que pueden o no incluirse en los elementos que contiene una diapositiva. Así pues, no todos los elementos dentro de una diapositiva deben contener una animación, sino que será el usuario quien establezca para cada elemento la/s animaciones que considere más adecuadas. Las animaciones permitirán mostrar, enfatizar u ocultar elementos de una diapositiva para ayudar al presentador a transmitir el mensaje.
- **Transición:** espacio entre una diapositiva y la siguiente en la presentación. Se trata de un elemento opcional en una presentación de diapositivas, ya que el usuario puede optar porque no haya ningún efecto de transición entre dos diapositivas. Elegir una transición adecuada para una presentación no es tarea fácil ni tampoco un aspecto trivial, pues la elección de la transición afectará al ritmo de la presentación.



Utilizando estos tres elementos principales, cualquier usuario podrá crear una presentación con diapositivas para un propósito concreto. Sea cual sea la herramienta de creación y edición de diapositivas utilizada, ésta permitirá al usuario incluir cualquier tipo de contenido en sus diapositivas y hacer una presentación ante la audiencia, ya que todas ellas incorporan un modo de presentación o pase de diapositivas a tal efecto.

El modo de presentación o pase de diapositivas es el que ofrece cualquier herramienta de creación y edición de diapositivas para impartir la presentación una vez realizada.

A través de él, el presentador puede compartir con la audiencia las diapositivas utilizadas (ocultando los aspectos de diseño) e interactuar con ellas a través de las animaciones y transiciones definidas para que la presentación sea lo más efectiva posible.

A continuación, se muestran algunas de las herramientas más populares en la actualidad:

Herramienta	Logo	Descripción
PowerPoint (privativa)		PowerPoint es el software de creación y edición de presentaciones de ***Microsoft*** (*e.digitall.org.es/powerpoint*) que permite a los usuarios crear atractivas presentaciones que constan de páginas individuales, o diapositivas. Es una herramienta muy completa y su uso está ampliamente extendido. Es compatible con Windows y macOS. No obstante, es una herramienta de pago y se debe adquirir una licencia o pagar una subscripción mensual o anual.
Keynote (privativa)		Es una herramienta gratuita integrada en la suite ofimática de ***Apple*** (*apple.com/keynote*) que permite crear presentaciones efectivas de una manera sencilla. En este software es muy fácil crear animaciones y secuenciarlas si fuera necesario, ya que dispone de multitud de efectos de animación muy originales. Keynote permite trabajar sin problemas entre dispositivos macOS y iOS, además de poder editar presentaciones realizadas con PowerPoint.
Presentaciones de Google (privativa)		Es una herramienta gratuita de Google. ***Presentaciones de Google*** (*e.digitall.org.es/slides*) está disponible como aplicación web, aplicación móvil para Android, iOS, Windows, BlackBerry y como aplicación de escritorio en ChromeOS de Google. Los usuarios pueden acceder a las presentaciones, a través del sitio web de Google Drive. Las presentaciones pueden ser editadas y compartidas por múltiples usuarios simultáneamente y los usuarios pueden ver los cambios diapositiva por diapositiva a medida que otros colaboradores realizan ediciones. También es compatible con los formatos de archivo de PowerPoint.
LibreOffice Impress (libre)	*Galdam Jitsu (Geeko), CC BY-SA 4.0 e.digitall.org.es/cc-licenses via Wikimedia Commons*	Es una herramienta gratuita y con licencia de software libre desarrollada por The Document Foundation para la suite ofimática de ***LibreOffice*** (*e.digitall.org.es/impress*). Su uso está muy extendido principalmente dentro de sistemas operativos libres, como los basados en GNU/Linux. Su interfaz es similar a PowerPoint y además es compatible con este formato.
Beamer (libre)		Beamer es un complemento empleado por la comunidad de usuarios del editor de textos LaTeX. Este permite la reutilización de documentos en ***LaTeX*** (*latex-project.org*) para crear presentaciones de alta calidad tipográfica. No es propiamente una herramienta de presentación, lo que limita sus posibilidades. Además, su uso es imposible para usuarios que no conocen LaTex. Está disponible para los sistemas operativos Windows, macOS, Unix, GNU/Linux, etc.
Prezi (privativa)	Prezi	Es una herramienta de ***Prezi*** (*prezi.com*). Estas presentaciones parten de una imagen o grafismo que muestra la totalidad de la presentación para, utilizando la metáfora de una lupa, ir haciendo zoom sobre distintas áreas para mostrar el contenido. Las presentaciones de Prezi son muy efectistas y dinámicas, e incluyen animaciones que permiten a la audiencia ver, entender y recordar ideas. Este tipo de herramientas está destinada a elaborar presentaciones en un contexto más informal o familiar.
Canva (privativa)	Canva	***Canva*** (*canva.com*) ofrece servicios no solo para la creación de presentaciones, sino también de infografías y otros tipos de material publicitario. Su interfaz con diseño drag & drop (arrastrar y soltar) hace que sea muy fácil para el usuario diseñar y crear una presentación o cualquier otro tipo de material. Además, Canva ofrece una enorme biblioteca de plantillas con miles de elementos incorporados y que el usuario puede utilizar según su criterio. Esta herramienta está disponible como aplicación web, aplicación móvil para Android y iOS y de escritorio para Windows y macOS.

Creación de
contenidos digitales

Nivel A1 3.2 Integración y reelaboración de contenido digital

Hojas de Cálculo: representación y cálculo de datos

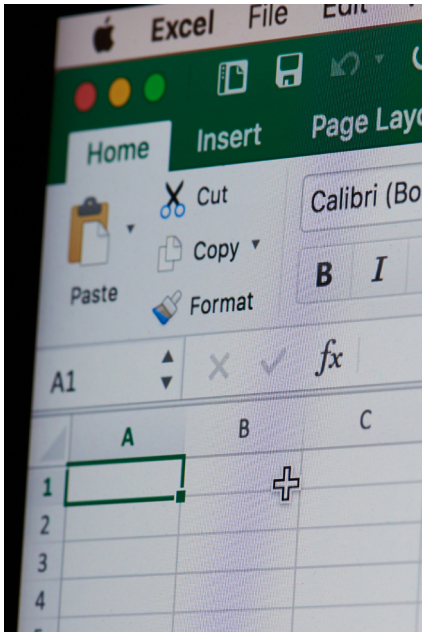
Hojas de Cálculo: representación y cálculo con datos

Una hoja de cálculo es una herramienta informática que permite trabajar con datos numéricos y alfanuméricos. Estas herramientas permiten realizar operaciones matemáticas con los datos y obtener representaciones gráficas de los mismos. Estas figuras pueden ser reutilizadas en presentaciones o en informes. Por ejemplo, un profesor puede emplear una hoja de cálculo para gestionar las calificaciones de sus estudiantes. Esta herramienta le permitiría calcular las calificaciones medias, realizar estadísticas o crear listados de alumnos que incluyesen gráficos estadísticos.

Las herramientas de hojas de cálculo cuentan con una serie de elementos comunes, destacando:

- **Celda:** la unidad básica de presentación de la información es la celda. En ella se almacena un dato (numérico o alfanumérico).
- **Hoja:** el conjunto de celdas de una tabla se denomina hoja. El conjunto de celdas horizontales de una tabla constituye una fila y el conjunto de celdas verticales se denominan columna. Las filas se identifican con números y las columnas con letras. Por tanto, cada celda queda definida por la fila y la columna que ocupa dentro de la tabla. Por ejemplo, A3 se refiere a la celda de la primera fila y tercera columna.
- **Rango:** son todas las celdas contenidas en un rectángulo definida por la celda superior izquierda y por la celda inferior derecha. Por ejemplo, A1:B2 son las celdas A1, A2, B1 y B2.
- **Libro:** es un archivo de una hoja de cálculo y está conformado por varias hojas. Cada hoja tiene un nombre para poder ser referenciado los datos contenidos en ella.
- **Plantilla:** libro que utiliza como base para crear otros libros similares. Se pueden crear plantillas para libros y hojas.
- **Operaciones y funciones:** Las hojas de cálculo permiten realizar operaciones aritméticas (sumas, restas, multiplicaciones y divisiones) empleando filas o columnas. También existen cálculos más complejos que se recogen

en las denominadas funciones. Las funciones permiten realizar cálculos sobre un conjunto de datos situados en una o en varias hojas. Por ejemplo, se puede calcular la media de una columna de datos, sumar todos los valores de una fila o contar el número de apariciones de un determinado número o texto en una región seleccionada. Se puede hacer que el contenido de una celda sea el resultado de aplicar una determinada función.



- **Gráficos:** el usuario puede seleccionar una región de la hoja e incrustar un tipo de gráfico que es elaborado a partir de un conjunto de datos seleccionados. Estas herramientas disponen de una amplia variedad de tipos de gráficos predefinidos como los de líneas, áreas, columnas, barras, dispersión, circulares, histogramas, u organigramas entre otros.

⚠ ATENCIÓN

Los usos más habituales de las hojas de cálculo son:

- **Organizar datos.** La posibilidad de disponer de múltiples hojas permite clasificar y organizar los datos, además de realizar diferentes tipos de búsquedas.
- **Compartir datos.** Los libros se pueden compartir entre usuarios, permitiendo de forma sencilla y ágil el acceso a los datos a diferentes usuarios.
- **Depurar y filtrar datos.** Las hojas contienen todos los datos, pero disponen de filtros que permiten visualizar versiones simplificadas de los mismos que permiten el análisis y depuración de estos.
- **Graficar datos.** Permite realizar diversos tipos de gráficos y diagramas con el objetivo de facilitar la compresión de la información, analizar patrones, etc. Estos gráficos se pueden posteriormente incluirse en presentaciones o en informes.
- **Manipulación matemática de los datos.** Se pueden realizar de forma sencilla cálculos matemáticos con los datos de un libro. Estos cálculos pueden ir de operaciones simples hasta problemas matemáticos complejos.

A continuación, se muestran algunas de las herramientas más populares en la actualidad:

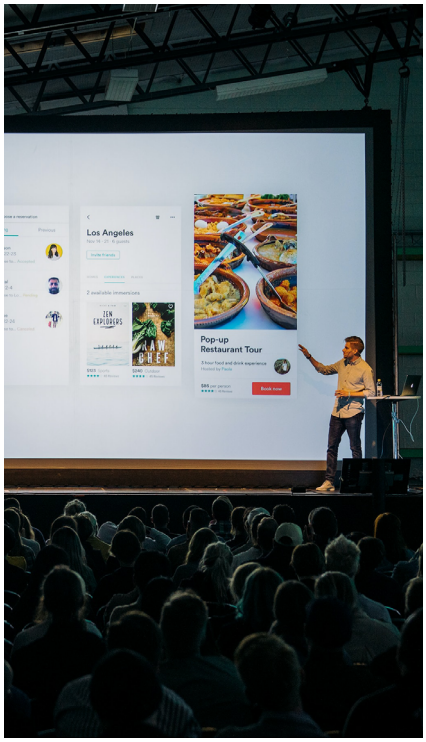
Herramienta	Logo	Descripción
Excel (privativa)		Excel es el software de hojas de cálculo de ***Microsoft*** (e.digitall.org.es/excel) que permite a los usuarios trabajar con múltiples funciones. Es una herramienta muy completa y su uso está ampliamente extendido. Es compatible con Windows y macOS. No obstante, es una herramienta de pago y se debe adquirir una licencia o pagar una subscripción mensual o anual
Numbers (privativa)		Es una herramienta gratuita integrada en la suite ofimática de ***Apple*** (apple.com/numbers) que permite crear hojas de cálculo de una manera sencilla. Destaca por su simplicidad y elegancia. Numbers permite trabajar sin problemas entre dispositivos macOS y iOS, además de poder añadir diagramas utilizando un Apple Pencil en iPad. Además, incorpora una gran variedad de plantillas predefinidas para una gran cantidad de escenarios de uso.
Hojas de cálculo de Google (privativa)		Es una herramienta gratuita de Google. ***Hojas de cálculo de Google*** (e.digitall.org.es/sheets) está disponible como aplicación web, lo que permite a los usuarios trabajar con diferentes sistemas operativos. Los usuarios pueden acceder a las hojas de cálculo, a través del sitio web de Google Drive. Las hojas de cálculo se guardan en la nube y pueden ser usadas por múltiples usuarios simultáneamente. Los usuarios pueden ver los cambios a medida que otros colaboradores realizan ediciones. También es compatible con otros formatos externos, incluido el archivo de Excel.
LibreOffice Calc (libre)	***Galdam Jitsu (Geeko), CC BY-SA 4.0*** e.digitall.org.es/cc-licenses via Wikimedia Commons	Es una herramienta gratuita y con licencia de software libre desarrollada por The Document Foundation para la suite ofimática de ***LibreOffice*** (e.digitall.org.es/calc). Su uso está muy extendido principalmente dentro de sistemas operativos libres, como los basados en GNU/Linux. Su interfaz es similar a Excel y además es compatible con este formato.
Zoho Sheet (privativa)		***Zoho Sheet*** (zoho.com/sheet) es una aplicación que forma parte de la suite ofimática de este desarrollador. Cuenta con las funciones habituales, tablas dinámicas y gráficos. Realiza el trabajo en la nube y por tanto permite el trabajo colaborativo y acceder a los proyectos desde cualquier ordenador con conexión a internet. Es gratuito hasta 25 usuarios por lo que es una alternativa para pequeñas empresas y usuarios finales.

Creación de
contenidos digitales

Nivel A1 3.2 Integración y reelaboración de contenido digital

Composición de contenidos digitales existentes

Composición de contenidos digitales existentes



Cuando preparamos o elaboramos contenido digital para cualquier propósito (presentar un producto, anunciar un servicio, elaborar contenido sobre un acontecimiento o viaje personal), a menudo disponemos de elementos como imágenes, audios o vídeos que queremos **combinar** para generar el nuevo contenido. En estos casos, nuestro objetivo es generar un contenido digital de tipo ***slideshow***, es decir, una **composición** de imágenes, vídeos y sonidos con un propósito específico.

En la actualidad existen multitud de **herramientas software gratuitas** e intuitivas para que cualquier usuario sin un conocimiento experto pueda generar su propio contenido digital. Algunas de estas herramientas incluso son accesibles directamente desde la **web**, sin necesidad de instalar ningún software específico. Todas ellas cuentan con un sistema de edición común, aunque haya ciertas diferencias y particularidades en los menús y botones de cada una de ellas. Para ello, utilizan un sistema de tipo ***drag and drop***, que permite arrastrar y soltar en una línea temporal los elementos que queremos que formen parte del nuevo contenido. Estos elementos se mostrarán tal y como los hayamos ordenado en la línea de tiempo.

A la hora de utilizar **imágenes**, debemos saber que es posible realizar múltiples transformaciones y procesamientos sobre ellas para lograr el efecto deseado. Así, es posible utilizar programas y herramientas web que permiten **incluir texto** en una foto (para crear, por ejemplo, un meme), **combinar varias** fotos existentes (creando montajes en los que, por ejemplo, la cabeza de una persona aparece en el cuerpo de otra, se fusionan dos escenas o dos paisajes…), **crear animaciones** a partir de varias imágenes, **realizar efectos** sobre ellas para conseguir un resultado original y **crear vídeos** a partir de un conjunto de imágenes.

Por su parte, el **audio** es un elemento muy útil para dar **ritmo** a la presentación y generar **ambiente**. Para ello, es posible introducir piezas de audio que acompañen a una imagen para aportarle más **acción** (por ejemplo, un audio de bullicio sobre la imagen de un atasco) o insertar audio de fondo a lo largo de

distintas imágenes con el objetivo de crear ambiente y generar **emociones** (por ejemplo, incluir una melodía de una nana sobre las fotos de un bebé). En un vídeo, también es posible **sustituir** todo o parte del audio que contiene el vídeo por otro de nuestra elección.

Con respecto al **vídeo**, es el elemento multimedia con mayor capacidad de transmisión de contenidos. Es especialmente importante que tanto las imágenes como los audios y vídeos que se incluyan en el nuevo contenido sean de **buena calidad**. Además, en el caso del audio y el vídeo es especialmente importante asegurarse que los archivos de este tipo que se quieren incluir son **compatibles** con el software que se está utilizando para generar el contenido digital. En caso de no serlo, existen **herramientas gratuitas de conversión de archivos** que se pueden utilizar, tanto como aplicación de escritorio como en versión web. Una vez que el contenido ya ha sido elaborado, lo más común es exportarlo en formato de vídeo. Este proceso se conoce con el nombre de **renderizado**. El renderizado llevará más o menos tiempo y generará un archivo de menor o mayor peso en función de la cantidad de elementos introducidos, la duración del vídeo y la resolución deseada.

Para **combinar** todos los elementos anteriores es posible también **utilizar transiciones y animaciones o efectos especiales**. Las transiciones permiten combinar distintos elementos a través de efectos que permiten dar continuidad al contenido. Las animaciones o efectos especiales, a su vez, aportan dinamismo al contenido generado para mejorar su impacto.

De todas estas formas es posible crear contenidos digitales a partir de otros existentes. Estos nuevos contenidos van desde un **meme**, una **infografía** o una **presentación** hasta contenidos más complejos de tipo **slideshow** que combinen todos los elementos multimedia disponibles.

Saber más

Herramientas para conversión de archivos:

• Conversor online gratuito de audio que soporta hasta 300 formatos.
online-audio-converter.com/es

• Herramienta que permite convertir cualquier formato de texto, imagen, vídeo o audio, a otro del mismo tipo.
online-convert.com

DigitAll

Creación de contenidos digitales

3.3 DERECHOS DE AUTOR Y LICENCIAS DE PROPIEDAD INTELECTUAL

Creación de
contenidos digitales

Nivel A1 3.3 Derechos de autor y licencias de propiedad intelectual

Derechos de autor y licencias de propiedad intelectual

Derechos de autor y licencias de propiedad intelectual

Propiedad intelectual: conceptos fundamentales

Son objeto de propiedad intelectual todas las creaciones originales literarias, artísticas o científicas expresadas por cualquier medio o soporte, tangible o intangible, actualmente conocido o que se invente en el futuro, comprendiéndose entre ellas:

- Los programas de ordenado.
- Las colecciones de obras ajenas, como las bases de datos que por la selección o disposición de sus contenidos constituyan creaciones intelectuales. La protección reconocida a estas colecciones se refiere únicamente a su estructura en cuanto forma de expresión de la selección o disposición de sus contenidos, no siendo extensiva a éstos.

Derechos de autor en creaciones intelectuales

La propiedad intelectual está integrada por derechos de carácter personal y patrimonial, que atribuyen al autor la plena disposición y el derecho exclusivo a la explotación de la obra, sin más limitaciones que las establecidas en la Ley.

Se considera autor a la persona natural que crea alguna obra literaria, artística o científica. Se presumirá autor, salvo prueba en contrario, a quien aparezca como tal en la obra, mediante su nombre, firma o signo que lo identifique. La condición de autor tiene un carácter irrenunciable; no puede transmitirse “inter vivos” ni “mortis causa”, no se extingue con el transcurso del tiempo así como tampoco entra en el dominio público ni es susceptible de prescripción.

La propiedad intelectual de una obra literaria, artística o científica corresponde al autor por el solo hecho de su creación. Incluye el conjunto de derechos que corresponden a los autores y a otros titulares (artistas, productores, organismos de radiodifusión...) respecto de las obras y prestaciones fruto de su creación.

En el sistema anglosajón, a diferencia del continental, el derecho de autor se conoce como “copyright”.

- Propiedad Intelectual = Derechos de Autor ➡ Sistema Continental e Hispano hablante
- Copyright = Derechos Explotación ➡ Sistema Anglosajón

Definición de copyright

El símbolo del Copyright © informa al público que una obra es original y que su uso, reproducción, transformación, publicación, etc. está sujeta a derechos de autor. El titular o cesionario en exclusiva de un derecho de explotación sobre una obra o producción protegidas por la ley podrá anteponer a su nombre el símbolo © con precisión del lugar y año de la divulgación de aquéllas. No se requiere solicitar la inscripción de los derechos de propiedad intelectual de la obra en cuestión para incluir el símbolo ©.

Los símbolos y referencias mencionados deberán hacerse constar en modo y colocación tales que muestren claramente que los derechos de explotación están reservados.

Public Domain (Domino Público)

Los derechos de explotación de una obra, en cualquier forma y, en especial, los derechos de reproducción, distribución, comunicación pública y transformación durarán toda la vida del autor y 70 años después de su muerte o declaración de fallecimiento. La extinción de los derechos de explotación de las obras determinará su paso al dominio público. Las obras de dominio público podrán ser utilizadas por cualquiera, siempre que se respete la autoría y la integridad de la obra e impidiendo cualquier deformación, modificación, alteración o atentado contra ella que suponga perjuicio a sus legítimos intereses o menoscabo a su reputación.

Las obras de Dominio Público pueden estar reguladas por Creative Commons con dos marcas:

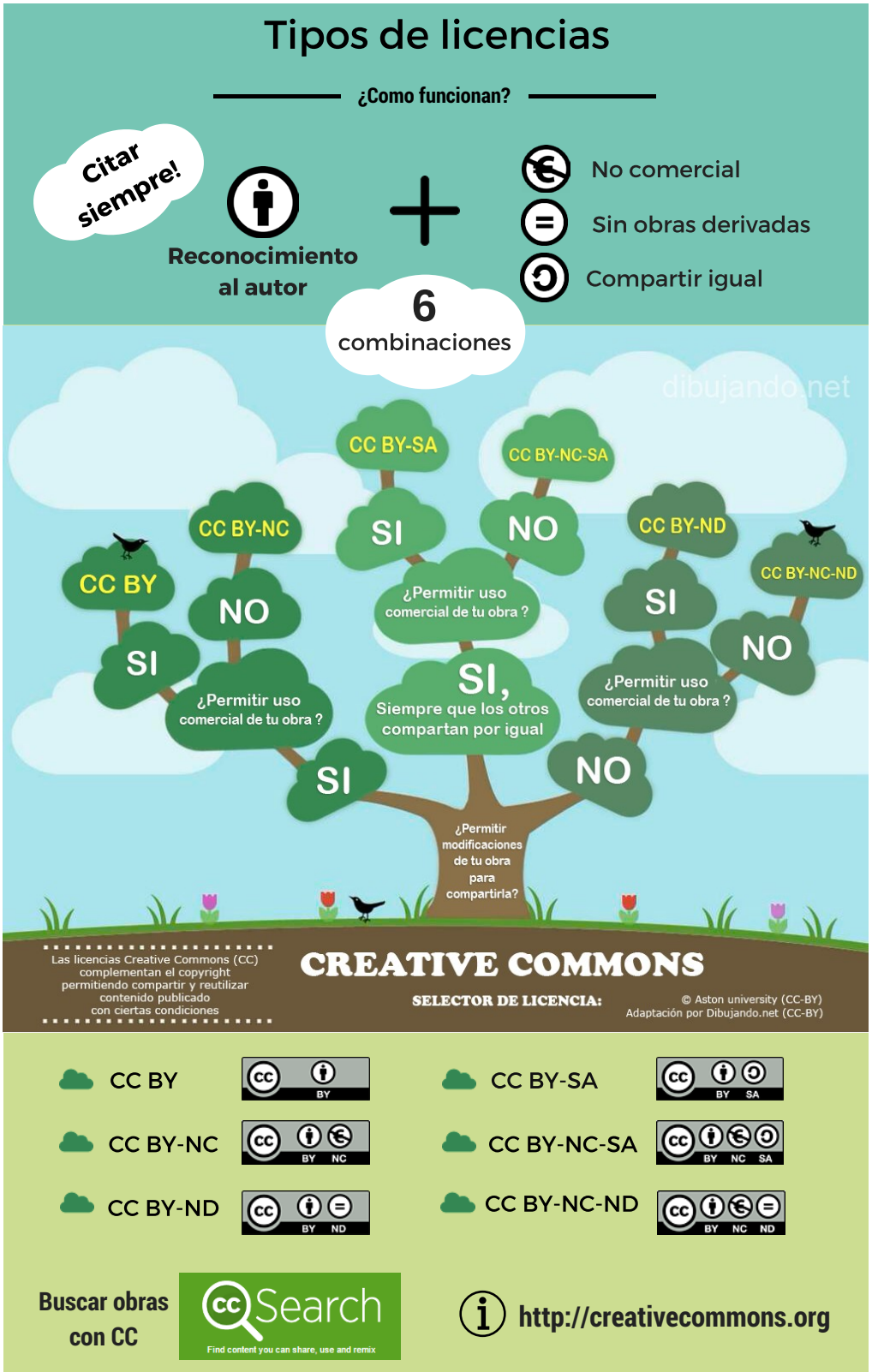
- **© Public Domain:** indica las obras que no están protegidas por derechos de autor (de explotación) por haber expirado el plazo de protección y que, por tanto, pertenecen al Dominio Público.
- **0 Public Domain:** es una marca que los autores de nuevas creaciones pueden otorgar a sus obras y que indica que renuncian a cualquier derecho sobre la misma, teniendo igual tratamiento que si se tratase de una obra en Dominio Público.

Derechos de autor en creaciones intelectuales

Las licencias CC permiten a los autores, u otros titulares de derechos, autorizar y/o ceder, bajo ciertas condiciones, algunos de los derechos sobre sus obras y mantener otra parte de éstos. Las CC se pueden aplicar a cualquier tipo de contenidos creativos, incluidas las bases de datos.

Se gestionan a través de ***Creative Commons*** (***creativecommons.org***), organización no gubernamental y sin ánimo de lucro que ofrece a autores y creadores, licencias y herramientas libres.

Cada licencia CC detalla qué derechos cede el autor, bajo qué condiciones y qué pueden hacer los usuarios con la obra, sobre todo en el caso de que quieran volver a publicarla o modificarla. Existen 4 condiciones básicas que se combinan en las licencias CC: Reconocimiento (obligatorio e imprescindible), Uso No comercial, Sin Obras Derivadas, y Compartir Igual. Su combinación genera 6 licencias CC diferentes, tal y como se esquematiza en el siguiente esquema:



Iconos creados por Freepik desde www.flaticon.com (CC BY 3.0)

crue Universidades Españolas Red de Bibliotecas REBIUN

Fuente: ***Creative commonESP*** (*e.digitall.org.es/rebiun*)

Creación de
contenidos digitales

Nivel A1 3.3 Derechos de autor y licencias de propiedad intelectual

Plagio

Plagio

Como ya hemos explicado en anteriores unidades, todos los derechos respecto a las obras literarias, artísticas y científicas pertenecen a los autores, sean estos artistas, escritores, productores o programadores informáticos. Y ellos son quienes deben autorizar el uso y explotación de sus creaciones, ya que la Propiedad Intelectual ampara todos sus derechos.

En los últimos 20 años se está produciendo un incremento exponencial de los plagios, debido en gran medida a la irrupción de Internet y a la masificación de su uso.

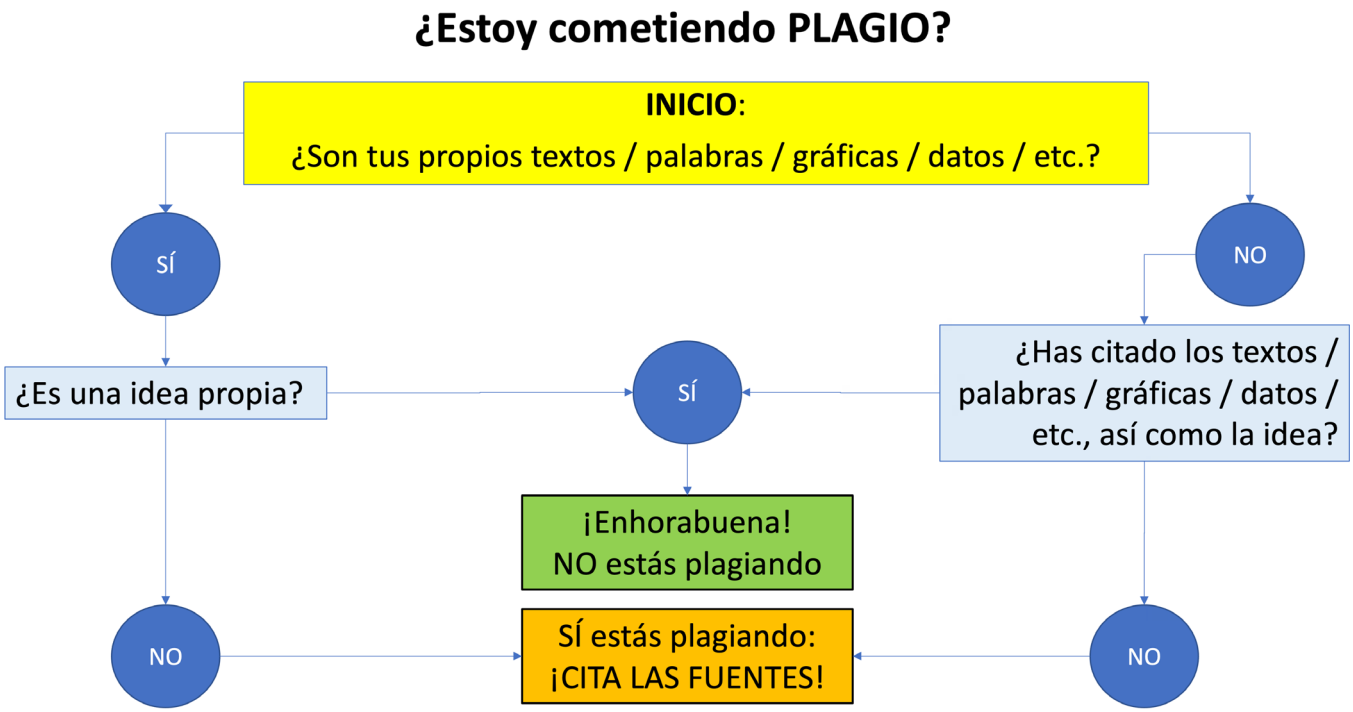
Plagiar consiste en copiar en lo **sustancial** obras ajenas, presentándolas como propias. No obstante, es importante señalar que no toda coincidencia entre dos o más creaciones supone un plagio. Solo cuando hablamos de **coincidencias estructurales básicas y fundamentales**, y no cuando son accesorias o añadidas, es decir, no transcendentales.

Muchas personas, de manera voluntaria o por desconocimiento, descargan de la red textos, fotografías, programas informáticos, artículos académicos o científicos, etc. **Esta información NO se puede usar libremente y como si fuera propia**, ya que están protegidos mediante la Propiedad Intelectual, excepto en los casos en que el autor lo autorice expresamente. Al igual que el resto de información, un blog, una página web o un documento electrónico son obras de algún autor y están sujetos a las mismas condiciones de uso y cita, ya que están amparadas por los Derechos de Autor.

No debes copiar y pegar sin más. **Siempre debemos citar la fuente de procedencia**. Dependiendo de la gravedad de lo copiado, se puede cometer una infracción grave regulada en el Código Penal (***artículos 270 a 272*** (***e.digitall.org.es/legislacion***)) y que conlleva graves consecuencias para el infractor, como son penas de prisión de 6 meses a 4 años. El artículo 270.2 es el responsable de que, durante los últimos años, hayan tenido que cerrar webs como RojaDirecta, Series.ly o EliteTorrent, de gran fama entre los internautas. Aunque únicamente mostraban enlaces a contenidos cargados en la web por sus usuarios, los Tribunales les hizo responsables de delitos contra la Propiedad Intelectual.

En el entorno universitario, más del 60% de los estudiantes ha incurrido en alguna forma de **plagio académico** para hacer sus trabajos. Para evitarlo, el estudiante no debe copiar el texto, las gráficas, los datos, etc., tal y como aparecen en los artículos originales, a menos que se cite el autor y la obra de donde proceden y se transcriban entre comillas.

El siguiente diagrama te permitirá determinar si estás o no plagiando:



DigitAll

Creación de
contenidos digitales

3.4
PROGRAMACIÓN

Creación de
contenidos digitales

Nivel A1 3.4 Programación

Características de los algoritmos y resolución de problemas

Características de los algoritmos y resolución de problemas

Existen multitud de problemas que se pueden resolver mediante un ordenador, pero todos ellos tienen en común el que parten de unos datos de entrada y se procesan mediante una serie de instrucciones con el fin de proporcionar unos resultados; es decir, siguen el esquema que se muestra en la Figura 1. Los programas son líneas de código escritas por personas, llamadas *programadores*, en algún lenguaje de programación que entiende el ordenador y que resuelven un problema. Podría pensarse que para solucionar un problema basta con aprender un lenguaje de programación y sentarse directamente a escribir código. Sin embargo, eso sería equivalente a construir un coche sin antes haberlo diseñado o a rodar una película sin un guion previo. En el contexto de la programación, los *algoritmos* constituyen una de las herramientas básicas para el diseño de los programas.

Figura 1. Partes de un problema resoluble con un ordenador.

Un algoritmo es la descripción de la solución de un problema mediante la **secuencia ordenada de pasos** que lo resuelven. Se caracteriza por ser:

- **Preciso,** es decir, no debe poseer ambigüedades para ser interpretado siempre de la misma manera.
- **Robusto,** esto es, no debe contener errores.
- **Definido**, respondiendo siempre de la misma manera ante las mismas circunstancias.
- **Ordenado,** dejando claro cuál es la secuencia de instrucciones que hay que realizar.
- **Finito,** lo que significa que acaba en algún momento.
- **Legible**, es decir, comprensible para cualquier persona que lo lea.

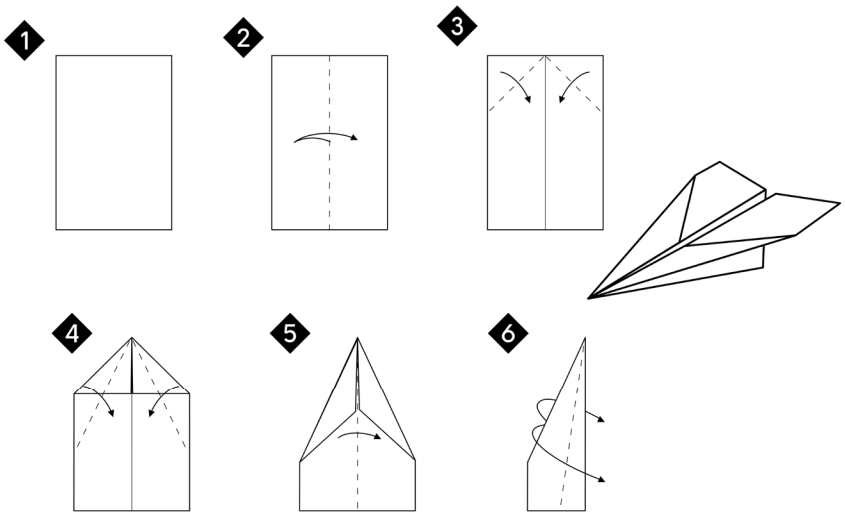


Figura 2. Instrucciones para realizar un avión de papel. (*e.digitall.org.es/aviones-papel*)

La Figura 2 ilustra un algoritmo para solucionar el problema consistente en la realización de un avión de papel. En este ejemplo, la entrada es la hoja de papel; la salida, el avión construido; y las instrucciones están representadas por las imágenes de la 1 a la 6. Observa que el algoritmo está descrito de forma gráfica y que cumple todas las propiedades: preciso, robusto, definido, ordenado, finito y legible.

En el contexto de la programación, los algoritmos se pueden escribir de muchas formas, aunque se suele usar el *pseudocódigo* por ser un lenguaje parecido al que utilizamos para comunicarnos, pero sin las ambigüedades e imprecisiones propias del lenguaje natural. Aprenderás todo sobre el pseudocódigo en otro tema posterior. Ahora lo más importante es que asimiles el proceso previo relacionado con la resolución de problemas. Porque para diseñar el algoritmo que describe la solución de un problema, antes es imprescindible **analizar el problema** de forma detallada, con el fin de identificar cada una de sus tres partes (Figura 1): datos de entrada, instrucciones para procesarlas y datos de salida. Para ello, puedes tratar de dar respuesta a las siguientes preguntas:

- ¿Qué quiero obtener? (*Salida*)
- ¿Con qué datos cuento? (*Entrada*)
- ¿Qué instrumentos tengo? y,
- ¿Qué instrucciones hay que seguir y en qué orden?

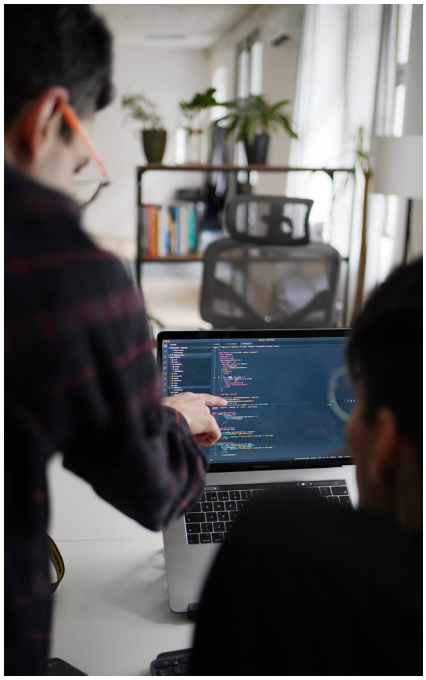
} (*Procesamiento*)

Lo vas a entender enseguida con un ejemplo:

Imagina que te piden que escribas un programa que calcule el área de un círculo, de radio un número real positivo. Para abordarlo, debes responder a las preguntas anteriores:

- **¿Qué quiero obtener?**
  Un número que represente el área del círculo.
- **¿Con qué datos cuento?**
  Con el número que representa el radio.
- **¿Qué instrumentos tengo?**
  La operación producto la proporcionan todos los lenguajes de programación.
- **¿Qué instrucciones hay que seguir y en qué orden?**
  Veamos:

  P1. Llamamos *radio*, por ejemplo, al número que representa el radio del círculo
  P2. Hay que obtener el valor del *radio* (supongamos que se le da el valor 2)
  P3. Llamamos *PI*, por ejemplo, al valor 3,14
  P4. Llamamos *área*, por ejemplo, al número que representa el resultado de la expresión *PI*radio*radio* (es decir, de 3,14*2*2)
  P5. Se proporciona como resultado el valor de *área* (el número 12,56)



Observa que no se puede mostrar el valor de área si aún no se ha calculado su valor; es decir, el paso P5 debe ir siempre después del paso P4. O no se puede hacer un producto de dos números si no se conoce el valor de uno de ellos; es decir, que el paso P4 debe ir siempre después del paso P2.

Sin embargo, el paso P3 puede ir antes o después del paso P1, y antes o después del paso P2. Además, el nombre de los números y las expresiones podría haber sido otro. Por ejemplo, al número que representa el radio se le podría haber puesto *r*, *R*, *radius*, *X*, o cualquier otro que nos guste. Lo mismo para el nombre del valor 3,14159 y para el valor de la expresión que calcula el área. Por tanto, se pueden crear distintos algoritmos igual de válidos que el que hemos escrito.

⚠ ATENCIÓN

El orden de las instrucciones es tan importante como las propias instrucciones.

⚠ ATENCIÓN

El algoritmo que describe la solución de un problema no tiene por qué ser único.

Para que te vayas familiarizando con el pseudocódigo, una posible descripción del algoritmo del ejemplo sería el siguiente:

```
Algoritmo AreaCirculo
Datos
  radio, área, PI: Números
Instrucciones
  Obtener el valor de radio
  PI ⬅ 3,14
  área ⬅ PI * radio * radio
  Proporcionar el valor de área
Fin AreaCirculo
```

Finalmente, puedes comprobar que el ejemplo cumple todas las propiedades de los algoritmos: es preciso, robusto, definido, ordenado, finito y legible. Además, el algoritmo es correcto: hace lo que debe hacer. Por tanto, es el momento de traducirlo a un lenguaje de programación, lo que se conoce como *implementación*. La Figura 3 contiene varias implementaciones del algoritmo del ejemplo en Pascal, Python y Java. Verás que todas son muy parecidas al algoritmo en pseudocódigo y, además, entre ellas, diferenciándose únicamente en detalles específicos de sintaxis.

```
program AreaCirculo;
var radio, PI, area:Real;
begin
  radio:=2;
  PI:=3.14;
  area:=PI*radio*radio;
  writeln (area);
end.
```

```
radio=2
PI=3.14
area=PI*radio*radio
print(area)
```

```
public class AreaCirculo {
        public static void main(String[] args) {
                double radio=2;
                double PI=3.14;
                double area=PI*radio*radio;
                System.out.println(area);
        }
}
```

Figura 3. Posibles implementaciones del algoritmo del ejemplo en Pascal, Python y Java.

En definitiva, una vez definido el algoritmo, la implementación es casi inmediata. Por tanto, para programar la solución de un problema, el proceso a seguir, de forma resumida, es el siguiente:

1 | Análisis del problema
Para identificar los datos de entrada, los pasos a seguir para su procesamiento y la salida que debe proporcionar.

2 | Diseño del algoritmo
Debe comprobarse que cumple todas las propiedades y que es correcto
Se escribirá en pseudocódigo.

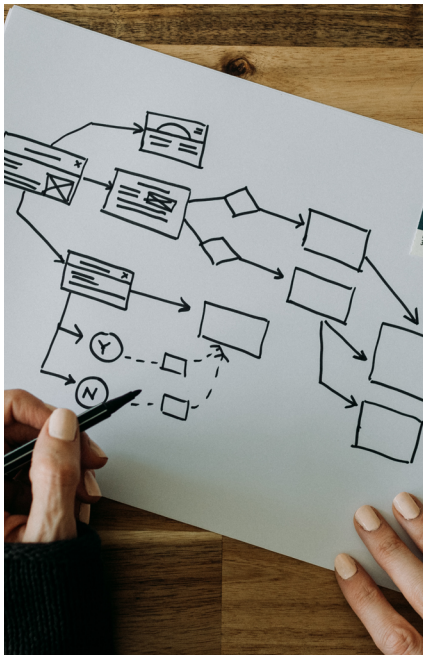
3 | Implementación
Traducción del algoritmo al lenguaje de programación elegido.

Creación de
contenidos digitales

Nivel A1 3.4 Programación

Diagramas de flujo

Diagramas de flujo



Tal como se ha comentado anteriormente, los diagramas de flujo permiten representar visualmente el flujo de cualquier algoritmo, facilitando así su comprensión. Se trata de una herramienta ampliamente utilizada en el ámbito de la programación de computadores, especialmente entre equipos de desarrollo software para la documentación de proyectos, y el intercambio de ideas y conceptos.

Veamos un ejemplo sencillo de uso. Para ello supongamos el siguiente problema, en el que se desea diseñar un algoritmo para controlar la venta de entradas a un concierto, limitada a mayores de edad. El algoritmo debe determinar si la venta está o no permitida. El algoritmo que resolvería la problemática mencionada anteriormente podría ser el siguiente:

1 | **Inicio del programa**
2 | **Solicitar al usuario el año de nacimiento**
3 | **Solicitar al usuario el año actual**
4 | **Calcular la edad en base a los dos datos anteriores como la diferencia entre el año actual y el año de nacimiento**
5 | **Consultar la edad y comprobar si es igual o superior a los 18 años**
   - En caso afirmativo, es decir, que la edad sea igual o superior a los 18 años, el programa debe mostrar por pantalla un mensaje informando sobre que la venta es permitida.
   - En caso negativo, es decir, la edad es inferior a los 18 años, se debe informar por pantalla de que la venta no está permitida.
6 | **Fin del programa**

Aunque el cálculo de la edad requiere una mayor precisión teniendo en cuenta el día de nacimiento y el mes, para este primer ejemplo se ha decidido simplificar únicamente al año de nacimiento. El algoritmo anterior tiene su representación equivalente en un diagrama de flujo, tal como muestra la siguiente figura.

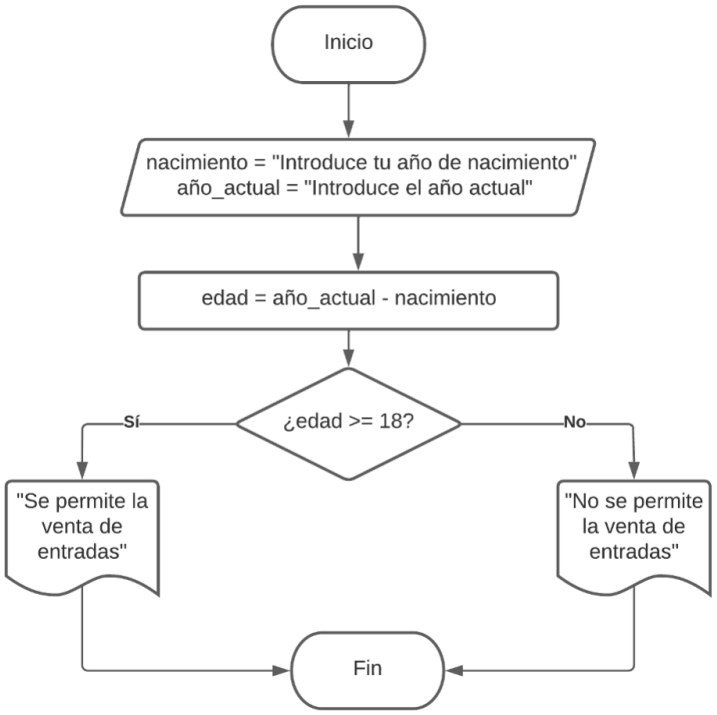
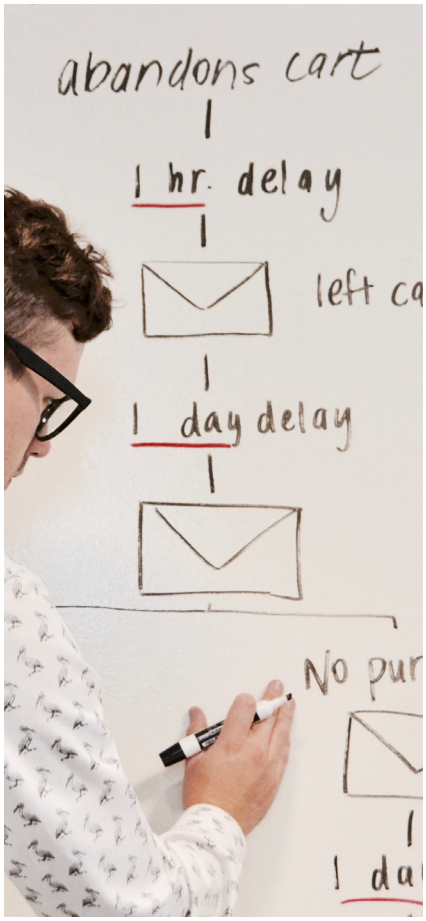


Figura 1. Diagrama de flujo asociado a un algoritmo para el control de venta de entradas en base a la edad.

Inicialmente se marca el inicio del programa. Las dos primeras instrucciones que debe ejecutar el programa corresponden a la solicitud de datos de entrada, representadas mediante un paralelogramo. Para que el usuario sea consciente de que el programa solicita datos y está a la espera de entrada, es necesario mostrar mensajes por pantalla. En primer lugar, se muestra por pantalla la frase "*Introduce tu año de nacimiento*", y el dato introducido por teclado se almacena en la una variable llamada *nacimiento*. En segundo lugar, se muestra por pantalla la frase "*Introduce el año actual*" y el dato introducido se almacena en la variable *año_actual*. Posteriormente se procede al cálculo de la *edad* restando el contenido almacenado en las dos variables anteriores. Al tratarse de un procedimiento, se representa mediante un rectángulo. Calculada la *edad* es posible consultar su valor (representado mediante un rombo). En función de si la evaluación deriva en un valor verdadero o falso, el flujo del algoritmo continúa por un camino distinto. Camino de la izquierda en el caso de que la edad sea mayor o igual a dieciocho, y el camino de la derecha en caso contrario. En ambos casos se muestra un mensaje al usuario informando de la decisión tomada. Finalmente se marca la finalización del programa.



Creación de
contenidos digitales

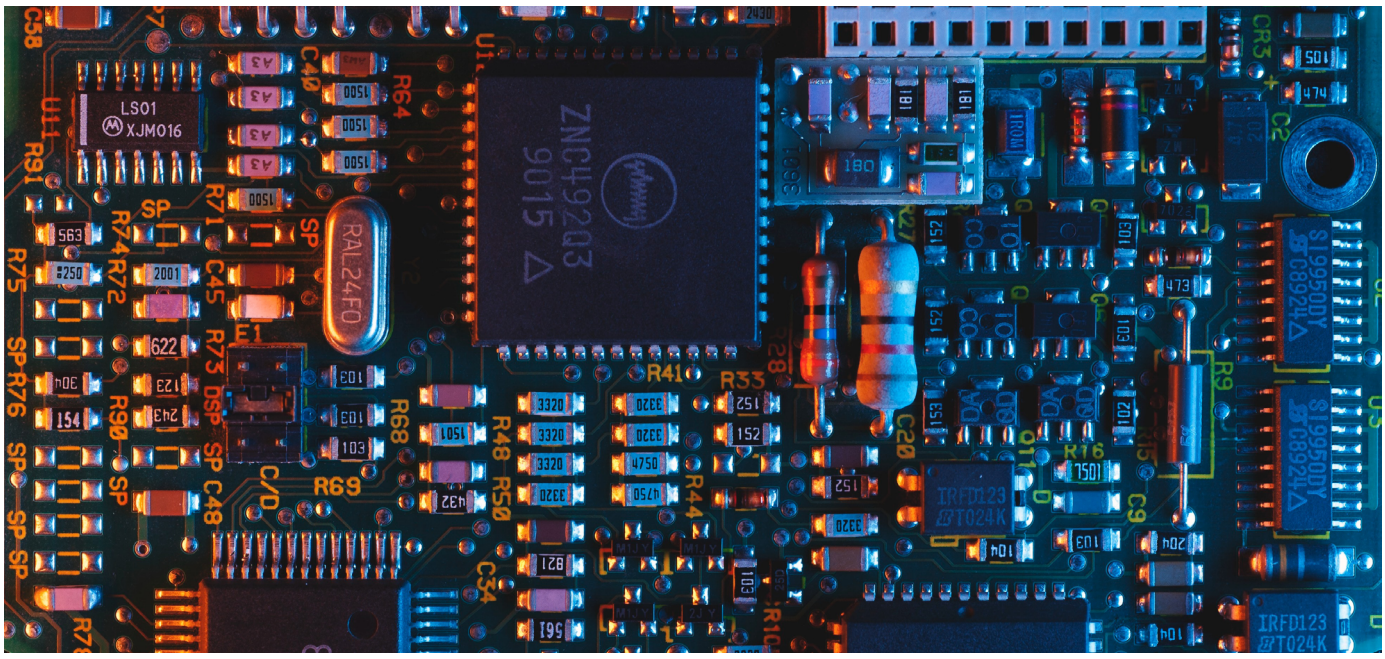
Nivel A1 3.4 Programación

Máquinas programables. El concepto de programa

Máquinas programables. El concepto de programa

Los móviles, tabletas, portátiles y ordenadores o computadores personales que usamos en nuestro día a día, y a los cuales nos podemos referir con carácter general como **máquinas**, no dejan de ser un conjunto de circuitos integrados con multitud de chips conectados entre sí, que nos permiten hacer una gran variedad de tareas en nuestras actividades cotidianas. En el ámbito laboral y escolar son un instrumento de trabajo y en el personal son un instrumento de relación, desarrollo personal y de entretenimiento. Sin duda alguna, estas máquinas o dispositivos informáticos son, hoy en día, unas herramientas fundamentales en nuestra vida diaria.

Podemos reflexionar sobre un ejercicio muy sencillo, y pensar en la cantidad de dispositivos electrónicos que integran algún tipo de dispositivo de procesamiento. Las televisiones, los equipos de cine en casa, las videoconsolas, los asistentes personales como Alexa, los dispensadores automáticos de comida para mascotas, los frigoríficos, las lavadoras, las calderas o los equipos de aire acondicionado son algunos ejemplos de elementos que nos rodean y que están basados en tecnología, materializada por una gran variedad de pequeñas máquinas.



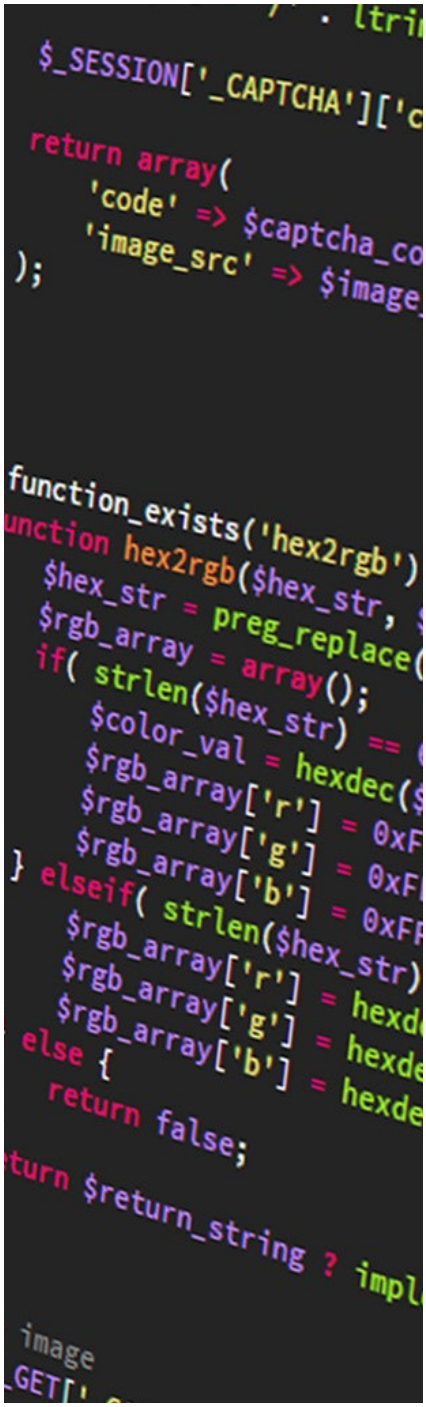
La placa base es uno de los componentes fundamentales de cualquier ordenador.

La generalidad de uso de los dispositivos informáticos o máquinas, necesario por otra parte si se considera la diversidad de problemas o áreas en los cuales son aplicables, implica un trabajo adicional cuando se quiere que resuelvan problemas particulares. Este trabajo consiste en decirle a la máquina o dispositivo qué es lo que debe hacer. En otras palabras, nos tenemos que comunicar con ellas, para ordenarles qué es lo que deben hacer. Desafortunadamente, y a pesar de los recientes avances en técnicas de procesamiento de lenguaje natural, todavía no hablamos el mismo idioma, especialmente en lo que se refiere a indicarle a una máquina el problema que debe resolver. En este punto, no hablamos de indicarle a un asistente virtual que nos reserve entradas para el estreno de una película en nuestro cine favorito. Nos referimos a transmitir las instrucciones que una máquina debe ejecutar para realizar una determinada tarea.

Así, **programar** se puede entender como el acto de convertir la funcionalidad que posee una aplicación en instrucciones u órdenes para la máquina que, eventualmente, las ejecutará. Por medio de la programación ordenamos a la máquina cómo se debe de comportar.

Una vez está claro qué es programar una máquina, es conveniente definir el concepto de **programa**, entendido como el conjunto de instrucciones y órdenes que contienen la funcionalidad deseada para la aplicación que se está desarrollando. Cada aplicación que ejecutamos en nuestra máquina es un programa con instrucciones que debe ejecutar. Instrucciones que serán ejecutadas de manera secuencial, con cambios en el flujo de ejecución causados por las propias instrucciones o por eventos que se produzcan de manera externa, como por ejemplo la interacción del usuario a través del teclado o el ratón.

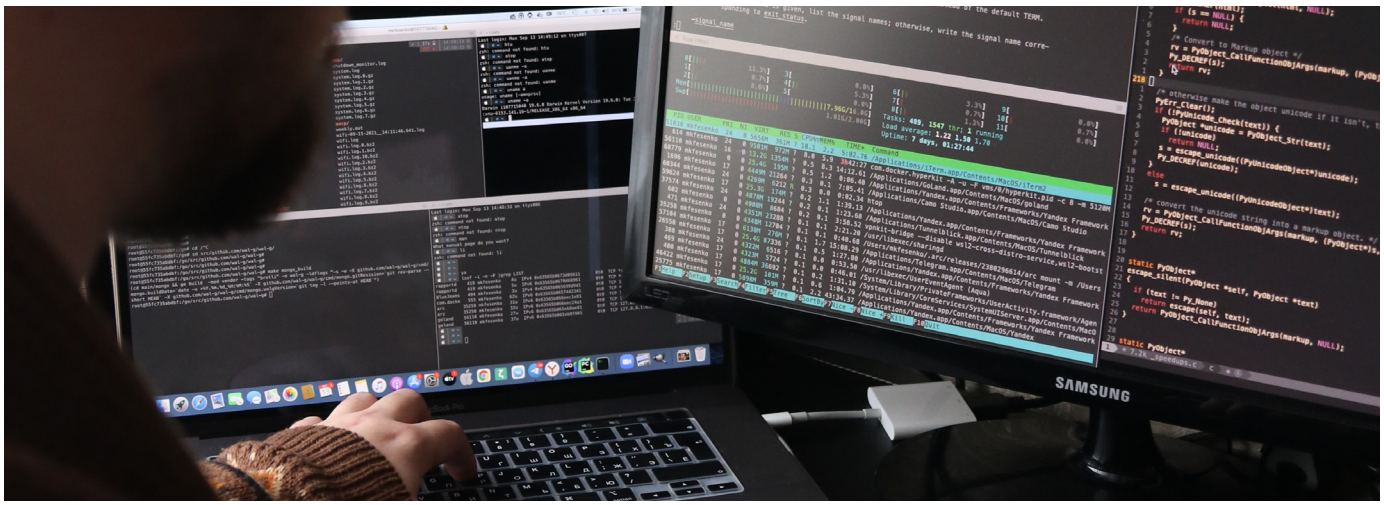
Recuerda que la máquina u ordenador está compuesta por componentes físicos. Algunos de los más relevantes son la placa base, que es el dispositivo que conecta a todos los elementos, procesador o CPU, responsable de ejecutar las instrucciones de los programas, memoria RAM, que permite almacenar información para proporcionar un rápido acceso a la misma, disco duro, que ofrece un mecanismo de almacenamiento de información más permanente, tarjeta gráfica, que optimiza el tratamiento de imágenes y vídeo, o



Fragmento de código fuente.

la tarjeta de sonido, responsable de gestionar los elementos sonoros. Sin embargo, el programa es un conjunto de instrucciones que alguien habrá escrito y almacenado en un archivo o fichero. La pregunta que nos podemos hacer en este punto es ¿cómo podemos comunicarnos o dar instrucciones a ese conjunto de elementos físicos electrónicos que conforman la máquina? Resulta evidente la necesidad de establecer un nexo entre esos dos mundos: i) el mundo físico, representado por los componentes de la máquina, y ii) el mundo lógico, representado por los programas.

Finalmente, resulta interesante que conozcas que la programación se engloba en lo que denominamos formalmente construcción de software, definido como un proceso complejo y sofisticado que va más allá de lo que comúnmente conocemos como programación. En este sentido, la construcción de software se puede dividir en varias etapas generales: i) definición del problema, para definir claramente, y de forma precisa, el problema que se quiere resolver, ii) especificación o definición de requisitos, que permite generar una descripción detallada acerca de lo que el sistema software debe hacer, iii) planificación del desarrollo, que tiene como objetivo detallar la planificación del resto de fases de la construcción de software, iv) diseño, que persigue organizar cómo se estructurará el código del programa, v) programación o codificación, entendida como la escritura de un conjunto de sentencias que persiguen la generación de un resultado, vi) pruebas, que pone el foco en comprobar que el programa creado funciona como debe y, finalmente, vi) mantenimiento, fase relacionada con la actualización del código.



Creación de
contenidos digitales

Nivel A1 3.4 Programación

Lenguajes de programación. Definición y evolución

Lenguajes de programación. Definición y evolución

Los circuitos que componen los ordenadores trabajan con dos niveles de tensión. Son circuitos digitales, y estos dos niveles se asocian con los números 0 y 1. ¿Te resulta familiar ahora el término *sistema binario*? Las acciones que puede realizar un dispositivo informático se representarán a través de un conjunto de secuencias de 0 y 1.

Si antes hemos definido un programa como "el conjunto de instrucciones que contienen la funcionalidad deseada para la aplicación que se está desarrollando", la recreación de ese programa en la memoria principal de un ordenador consistirá en la *traducción* de ese conjunto de instrucciones en secuencias de 0 y 1. A este sistema de códigos, directamente interpretable por la máquina, se le denomina **lenguaje máquina**.

Cada máquina tiene su propio *lenguaje máquina* con el que se puede programar. Este lenguaje es específico para la arquitectura interna de dicha máquina. En otras palabras, una máquina solo puede reconocer estas instrucciones o códigos de operaciones programada para ella. Todos los lenguajes máquina disponen de un conjunto de instrucciones similares: operaciones simples, para, por ejemplo, copiar información, operaciones aritméticas, operaciones lógicas, que manejan valores booleanos (verdadero y falso), y operaciones de entrada/salida, asociadas a las conexiones de la máquina con respecto al exterior.

Los programas en lenguaje máquina se ejecutan muy eficientemente, ya que se redactan específicamente para los circuitos que los han de interpretar y ejecutar. Este código es directamente interpretable por la CPU y no requiere de transformaciones previas para ser ejecutado. Sin embargo, la programación en lenguaje máquina es un trabajo difícil y extremadamente engorroso para el programador, siendo necesario que este conozca la arquitectura física de la máquina con un gran nivel de detalle.

Saber más

La programación en lenguaje máquina, en palabras de John Backus, era *un arte oscuro, una materia arcana, solo al alcance de unos pocos que comenzaron a considerarse como miembros de una clase sacerdotal guardiana de ciertas habilidades y misterios demasiado complejos para los mortales normales, que se oponían a ningún cambio revolucionario que pudiese hacer la programación tan simple que cualquiera pudiera realizarla.*

Precisamente, esta dificultad para programar fue lo que propició el objetivo de "democratizar la tarea de programar los ordenadores", reduciendo el periodo de formación de los programadores y facilitando la tarea de programar máquinas. Se pretendía, así, incrementar la distancia entre los programas y el frío acero de la máquina para, al mismo tiempo, reducirla entre el código y los programadores. Para ello, sería imprescindible, como discutiremos más adelante, establecer un nivel de independencia entre el programa y la arquitectura en la que este se ejecutará.

Uno de los primeros avances significativos en esta dirección fue el uso de una notación simbólica o *mnemónica*, empleada para representar cada instrucción o código de operación de la máquina. Estas claves mnemotécnicas eran más fáciles de recordar que los códigos numéricos, pero, sin embargo, requerían que, una vez establecida la secuencia de instrucciones en mnemónico que solucionaban el problema, fueran traducidas a lenguaje máquina. A este lenguaje se le denominó **lenguaje ensamblador**.

El siguiente listado de código muestra un ejemplo de suma de dos números en código ensamblador para una arquitectura 8086, almacenados en las direcciones de memoria 4000 y 4002. El resultado se almacena en las direcciones 5000 y 5002. El objetivo de mostrar este ejemplo es ilustrar el salto que existe entre llevar a cabo la programación en lenguaje máquina, directamente a través de ceros y unos, y la programación mediante una notación con un nivel semántico más elevado.

```
2000    MOV    CX, 0000        # Carga 0000 en el registro CX
2003    MOV    AX, [4000]      # Carga el número 3000 en el acumulador AX
2007    MOV    BX, [4002]      # Carga 3002 en el registro BX
200B    ADD    AX, BX [AX]     # Suma el contenido de AX y BX
200D    JNC    2010            # Salta a dirección 2010 si la suma no se lleva una
200F    INC    CX              # Incrementa el contenido del acumulador AX
2010    MOV    [5000], AX      # Mueve el contenido de AX a la posición 3004
2014    MOV    [5002], CX      # Mueve el contenido de CX a la posición 3004
2018    HLT                    # Para el programa
```

Sin embargo, y al existir una correspondencia estrecha, generalmente uno a uno, entre las claves del lenguaje ensamblador y los códigos de las operaciones de las máquinas, la programación continúa estando próxima a la máquina y sigue siendo un proceso minucioso y complicado. En este contexto, se hace necesaria la creación de un lenguaje lo más cercano posible al programador que permita expresar las distintas acciones a realizar por la máquina. Aquí aparece el concepto de **lenguaje de programación de alto nivel**.

Los lenguajes de programación de alto nivel suelen asociarse a la noción de *abstracción*. En otras palabras, cuando usas un lenguaje de este tipo te puedes centrar en crear un programa sin preocuparte de los detalles de la máquina que lo ejecutará. La principal ventaja de este enfoque es que la programación se simplifica y se hace más entendible, en parte porque las sentencias de programación se parecen más al lenguaje humano de lo que ocurriría con un lenguaje de programación de bajo nivel, o incluso con respecto al lenguaje ensamblador. Como abordaremos más adelante, el uso de un lenguaje de alto nivel implica la necesidad de herramientas que traduzcan sus sentencias a lenguaje máquina.

Finalmente, y como puedes imaginar, en la actualidad no existe un único lenguaje de programación de alto nivel. Por el contrario, existe toda una variedad de lenguajes que varían en función de su objetivo de diseño y de sus características. Algunos de los más populares en los últimos años, o incluso décadas, son C, Java, Python o JavaScript. La figura 1 muestra los lenguajes de programación de alto nivel más populares desde 2002.

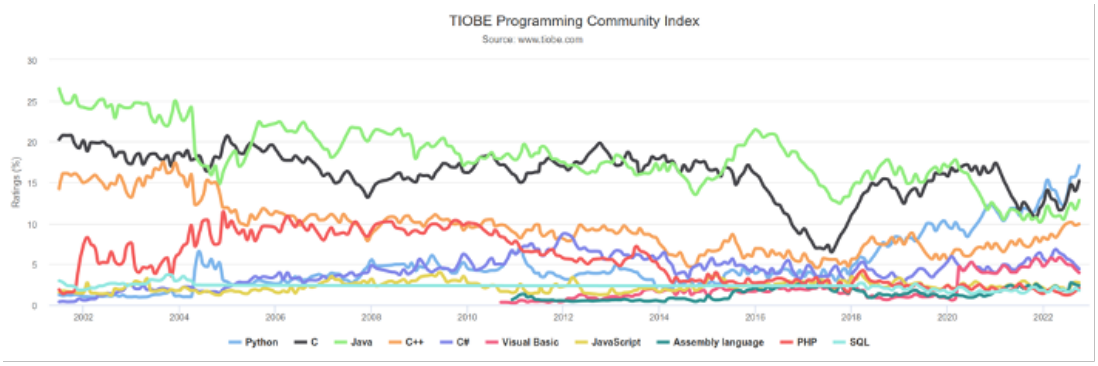


Figura 1. Índice TIOBE, en el que se muestra la evolución de los lenguajes de alto nivel más populares. Fecha de octubre de 2022.

Creación de
contenidos digitales

Nivel A1 3.4 Programación

Intérpretes VS Compiladores

Intérpretes VS. Compiladores

Para facilitar la programación de las máquinas se emplean los lenguajes de programación de alto nivel, como por ejemplo C++ o Python. Estos lenguajes permiten al programador indicarle qué es lo que quieren que haga en un lenguaje cercano a él pero lejano de lo que entiende la máquina, que son los lenguajes máquina.

De este modo surge la necesidad de algún tipo de traductor o procesador de lenguajes que sea capaz de convertir las instrucciones dadas por el programador en instrucciones ejecutables por la máquina.

Sobre esta base es posible construir unos programas especiales, diseñados para transformar los programas escritos en lenguajes de programación de alto nivel, entendibles por los humanos, en lenguaje máquina o código binario, entendible por las máquinas. Estos programas son **los compiladores y los intérpretes**. Por ejemplo, la existencia de un compilador para el lenguaje de programación C++ hace posible que un programa escrito en C++ se traduzca en lenguaje máquina asociado a una determinada arquitectura hardware. Del mismo modo, la existencia de un intérprete para el lenguaje de programación Python hace posible que un programa escrito en Python se pueda ejecutar sobre la misma u otra máquina o arquitectura física.

Si bien los compiladores y los intérpretes persiguen el mismo objetivo, esto es, traducir código escrito en un lenguaje de alto nivel en lenguaje máquina, existen **diferencias** sustanciales entre ambos. Tres de las más relevantes son las siguientes:

- Los compiladores transforman el código de un programa en código binario antes de ejecutarlo en una determinada máquina. En otra palabras, antes de ejecutar el programa es necesario compilarlo. Este proceso asegura, antes de usar el programa, que el código ha sido escrito de acuerdo a la especificación del lenguaje. Sin embargo, los intérpretes no generan código máquina ya que directamente las entienden y las convierten en instrucciones ejecutables en la máquina.
- El código compilado se ejecuta de manera más rápida que el código interpretado. Esto implica que haya ámbitos donde sea preferible usar un lenguaje compilado en lugar

de usar un lenguaje interpretado. Un ejemplo sería el mundo de los videojuegos, donde el programa, es decir, el videojuego, debe funcionar con una tasa de imágenes por segundo elevada para dar esa sensación de animación, al mismo tiempo que se encarga de interactuar con el jugador.

- Un compilador analizará todo el código fuente de un programa para encontrar errores antes de ejecutarlo. Por el contrario, el intérprete lo hará línea a línea.

En este contexto de compiladores e intérpretes, se suelen utilizar los términos *lenguajes de programación compilados* y *lenguajes de programación interpretados*, dependiendo de si la generación de código máquina se realiza a través de compiladores o intérpretes. A la hora de elegir un lenguaje de programación para crear un programa, se deberán tener en cuenta diferentes variables. Algunas de las más relevantes son la experiencia del equipo de trabajo con un determinado lenguaje de programación, el rendimiento requerido por el programa (los lenguajes interpretados están asociados a unos tiempos de ejecución más altos) o la capacidad del lenguaje para que el programador cree programas de manera ágil (los lenguajes interpretados suelen tener ventaja con respecto a los compilados).

Creación de
contenidos digitales

Nivel A1 3.4 Programación

Expresiones y asignación

Expresiones y asignación

Las **expresiones** son el mecanismo fundamental de cómputo de los programas, y se trata de la instrucción más básica que podemos utilizar cuando desarrollamos un programa. Las expresiones están formadas por **valores** (p. ej., 4 o 546) y **operadores** (p. ej., + o -), y se pueden **evaluar** por el computador para obtener un único valor o provocar algún efecto deseado.

Por ejemplo, podríamos construir una expresión que sume dos números dados utilizando una expresión como `2+3`. Esta expresión estaría formada por los valores 2 y 3, y el operador de suma (+). La evaluación de dicha expresión resultaría en el valor `5`. Así, podemos crear expresiones de diferentes formas mediante la combinación de operandos (números, variables, etc.) y operadores (matemáticos, lógicos, etc.).

Veamos a continuación los diferentes tipos de expresiones que existen en la mayoría de los lenguajes de programación.

Expresiones de asignación de variables

Como ya hemos visto anteriormente, las variables permiten al programa almacenar valores para trabajar con ellos posteriormente. En este sentido, las variables resultan ideales para guardar el valor resultante de evaluar una expresión y poder encadenar así expresiones a lo largo del programa.

Utilizando esta técnica podríamos conseguir almacenar, por ejemplo, el valor `23` en una variable llamada `número` utilizando la siguiente expresión:

```
numero = 23
```

Este tipo de expresiones se denomina **expresiones de asignación de variables** y están formadas por el operador de asignación (=) que divide la expresión de asignación de variable en una parte izquierda y una parte derecha. En la parte derecha del operador se encontrará la expresión cuya evaluación queremos guardar, mientras que en la parte izquierda se encontrará la variable que servirá almacenar el resultado de dicha evaluación. Como apunte adicional,

almacenar un valor en una variable por primera vez antes de que se utilice en el programa se denomina «inicializar una variable». Esto es importante porque una variable puede tener un valor inesperado si no se inicializa, lo que puede causar errores en el programa. Además, inicializar una variable asegura que siempre tenga un valor válido y conocido, lo que puede facilitar la corrección de errores y el mantenimiento del código.

Veamos algunos ejemplos de este tipo de expresiones y el resultado de sus evaluaciones:

```
1 | numero_impar = 23
2 | numero_par = 4
3 | resultado = numero_impar + numero_par
4 | numero_impar = resultado
```

En la línea 1 se inicializa una variable con el resultado de evaluar la expresión a la derecha del operador de asignación (=), guardando así el valor 23 en la variable `numero_impar`. De igual forma, en la línea 2, se inicializa otra variable (`numero_par`) con el valor 4. Después, en la línea 3 se ejecuta una expresión en la que se evalúa la suma de las variables creadas con el valor 27, asignando el resultado a la variable `resultado`. Finalmente, en la línea 4 se asigna el valor de la variable `resultado` (27) a la variable `numero_impar`, sustituyendo el valor de esta última (23) por el de la primera (27).

⚠ ATENCIÓN

Como se acaba de mencionar en el texto, una variable se puede inicializar *(crear)* la primera vez que se le asigna un valor. Tras esto, dicha variable puede ser usada en cualquier expresión posterior con otras variables y valores. Las siguientes asignaciones que sufra dicha variable simplemente sustituirán el valor que tuviera anteriormente por el resultado obtenido tras la nueva evaluación, pero sin crear una nueva variable con el mismo nombre.

Además de utilizar expresiones de asignación de variables a partir de otras expresiones definidas en el código del programa, existen expresiones cuya evaluación resulta en obtener datos de otras fuentes de datos como, por ejemplo, otros dispositivos de entrada. Estaríamos hablando, en este caso, de ejecutar una expresión de asignación de variables evaluando alguna otra

expresión que sea capaz de leer los valores desde el propio teclado del computador, por ejemplo:

```
1 | saludo = leer_teclado()
```

En la línea 1, la expresión de la derecha del operador de asignación detendrá la ejecución del programa hasta que el usuario introduzca algún texto utilizando el teclado y pulse la tecla [ENTER]. Al terminar la evaluación de la expresión, el texto introducido por teclado será el resultado que se le asignará a la variable saludo.

Expresiones que provocan efectos colaterales

Las expresiones que hemos visto hasta ahora pueden ser utilizadas para componer expresiones de asignación de variables, ya que el resultado de su evaluación puede ser almacenado en una variable. Sin embargo, existen otras expresiones cuya evaluación no devuelve ningún resultado, sino que **provoca un efecto colateral** en el estado del ecosistema de ejecución (p. ej., intérprete, sistema operativo, *hardware*,...). Por lo tanto, no podemos utilizar dichas expresiones en expresiones de asignación.

Por ejemplo, una expresión que muestre texto por la pantalla del computador no devolvería ningún valor, sino que utilizaría dicha pantalla para mostrar el resultado de evaluar esa expresión:

```
1 | mostrar_pantalla(‘¡Hola, mundo!’)
```

En este ejemplo, la evaluación de la expresión haría aparecer por pantalla el texto “¡Hola, mundo!”. Este sería el comportamiento esperado de ejecutar dicha expresión. Sin embargo, nótese como el resultado de la expresión no se asigna a ninguna variable, sino que simplemente se evalúa y provoca el efecto colateral de mostrar texto en pantalla.

Expresiones aritméticas

Las expresiones aritméticas nos permiten realizar operaciones matemáticas sobre valores numéricos. Los lenguajes de programación suelen proporcionar un conjunto de operadores aritméticos común que podemos utilizar para construir expresiones más complejas:

Operador	Operación	Expresión de ejemplo	Resultado de la evaluación de ejemplo
**	Exponente	3 ** 3	27
%	Resto de la división	55 % 7	6
//	División entera (truncada)	55 // 7	7
/	Division con decimales	55 / 7	7.857...
*	Multiplicación	4 * 5	20
-	Resta	6 - 2	4
+	Suma	1 + 2	3

Por ejemplo, utilizando la tabla anterior, podríamos construir expresiones como las siguientes:

```
1 |  2 + 2 * 4
2 | 15 + 3 * 24 ** 2 % 3
3 | ((15 + 3) * 24) ** (2 % 3)
```

El resultado de evaluar la expresión de la línea 2 puede ser confuso, por lo que resulta necesario conocer el orden en el que se evalúan los operadores aritméticos en las expresiones. Normalmente, el orden de precedencia de los operadores resulta muy similar al de los operadores en el ámbito de las matemáticas: primero se evalúa el operador de exponente (**); después los operadores de multiplicación (*), división entera (//), división con decimales (/) y resto de la división (%), respetando el orden de aparición de izquierda a derecha; por último, los operadores de suma (+) y resta (-). En cualquier caso, resulta conveniente utilizar paréntesis para facilitar la legibilidad de expresiones más complejas (línea 3).

Expresiones alfanuméricas

Las **expresiones alfanuméricas** son aquellas compuestas por valores textuales y numéricos. En los lenguajes de programación, existen algunas operaciones que podemos realizar entre cadenas de texto y valores numéricos.

Por una parte, podemos **concatenar cadenas de texto** utilizando el operador de suma (+):

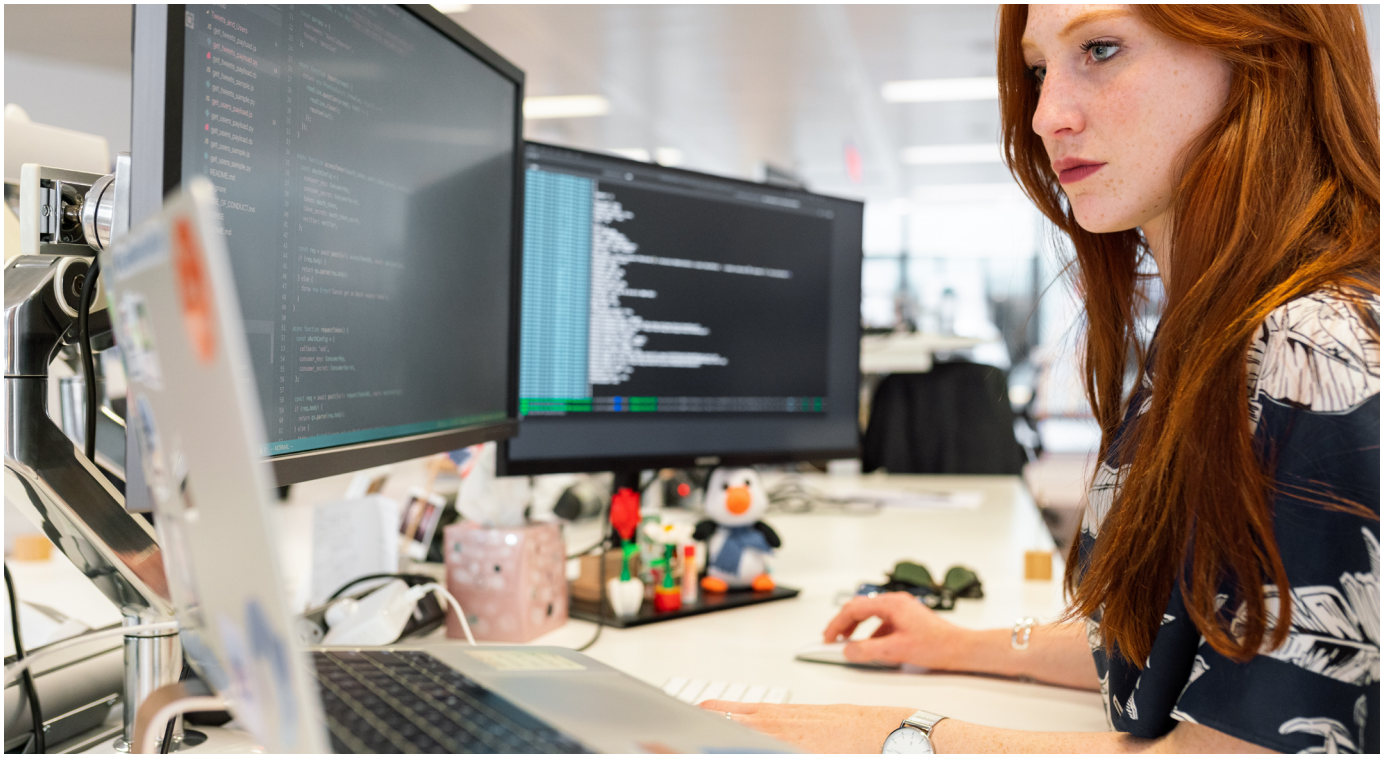
```
1 | 'Hola' + 'Adios'
```

En este ejemplo, el resultado de evaluar dicha expresión sería el texto 'HolaAdios'.

Por otra parte, podemos **replicar cadenas de texto** tantas veces como deseemos utilizando el operador de multiplicación (*):

```
1 | 'Hola' * 3
```

Esto daría como resultado el texto 'HolaHolaHola', ya que se ha replicado el texto original 3 veces.



Expresiones relacionales y lógicas

Las **expresiones relacionales** producen resultados de verdadero o falso a partir de su evaluación. Este tipo de expresiones se utilizan para construir sentencias condicionales, permitir a los programas tomar decisiones y alterar su flujo de ejecución en consecuencia.

Los lenguajes de programación proporcionan un conjunto de operadores relacionales que podemos utilizar para construir este tipo de expresiones, normalmente:

Operador	Operación
==	Igual a
!=	No igual a
<	Menor que
>	Mayor que
<=	Menor que o igual a
>=	Mayor que o igual a

A continuación, se muestran algunos ejemplos simples de expresiones relacionales:

```
1 | 'Hola' == 'Hola'
2 | 'Hola' == 'Adios'
3 | 'Hola' != 'Adios'
4 | 7 < 8
5 | 7 < 7
6 | 7 <= 7
```

Estas expresiones se evaluarían a valores de `True` (líneas 1, 3, 4 y 6) y `False` (líneas 2 y 5).

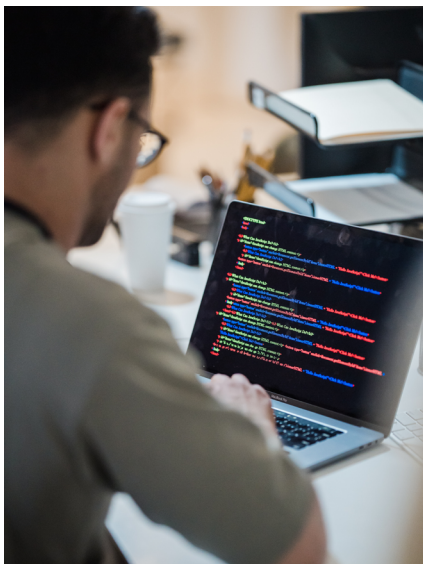
Por otra parte, existe una serie de operadores lógicos (`and`, `or` y `not`) que podemos utilizar para combinar varias expresiones relacionales y construir así las denominadas **expresiones lógicas**:

Expresión	Resultado de la evaluación
True and True	True
True and False	False
False and True	False
False and False	False
True or True	True
True or False	True
False or True	True
False or False	False
not True	False
not False	True

Por ejemplo, a continuación, se muestran algunas expresiones de este tipo:

```
1 | (4 < 5) and (5 < 6)
2 | (4 < 5) and (9 < 6)
3 | (1 == 2) or (2 == 2)
```

En la línea 1 primero se evaluarían las expresiones relacionales entre paréntesis, obteniendo una expresión intermedia tal que `True and True`. Después, se volvería a evaluar dicha expresión obteniendo como resultado final el valor `True`. La evaluación del resto de expresiones del ejemplo se realizaría de manera similar, obteniendo los resultados `False` (línea 2) y `True` (línea 3).

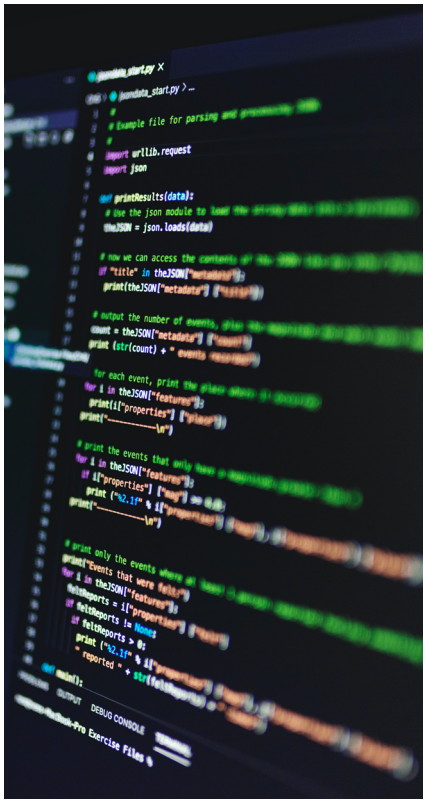


Creación de
contenidos digitales

Nivel A1 3.4 Programación

Control del flujo de ejecución

Control del flujo de ejecución



El término *flujo de ejecución de un programa* involucra tres conceptos que se relacionan entre sí: i) flujo, ii) ejecución y iii) programa. Aunque los vamos a definir a continuación, debes saber que existe una relación muy cercana, a nivel semántico, entre el flujo de ejecución de un programa y la idea de diagrama de flujo, previamente abordado desde una perspectiva visual a la hora de introducir el concepto de algoritmo.

El concepto de *programa*, previamente discutido, se puede definir como una secuencia de instrucciones que el ordenador es capaz de entender y que tienen un objetivo bien definido. En segundo lugar, la palabra *ejecución* hace referencia, precisamente, a esa capacidad de la máquina para hacer que el programa haga lo que debe hacer, empleando para ello los recursos físicos o hardware disponibles. Por ejemplo, uno de estos recursos sería un registro del procesador del ordenador que se emplea para almacenar o guardar el valor de una variable. Finalmente, el concepto *flujo* refleja esa noción de dinamismo de la que gozan los programas en ejecución, y que establece el camino que el programa recorrerá desde que se arranca hasta que termina. Cada paso de este camino estará representado por una instrucción de código. En este punto, es importante destacar que, aunque un programa esté compuesto de una secuencia de instrucciones, no tiene por qué ejecutarlas todas y cada una de ellas en un determinado orden. Por el contrario, el programa ejecutará un subconjunto de dichas instrucciones en función de una serie de condiciones.

Estas condiciones las establece el programador, de forma directa, a través de instrucciones que modifican o controlan el flujo de ejecución del programa. Por otra parte, el flujo de ejecución del programa puede verse alterado dependiendo del valor puntual de las variables que se usan en el programa. Como veremos en ejemplos posteriores, las instrucciones que se usan para controlar el flujo de un programa suelen combinarse con el uso de variables.

Pero antes de discutir cómo se puede controlar el flujo de un programa mediante sentencias más complejas, veamos un ejemplo más sencillo. El siguiente listado muestra la estructura general de un programa, compuesto por 7 instrucciones. De acuerdo a dicha estructura, un ordenador que ejecutara este programa procesaría primero `sentencia 1`. Después, procesaría `sentencia 2`. A continuación, procesaría la instrucción `goto etiqueta`. En este punto, el lector podría pensar que el flujo de ejecución del programa continuaría ejecutando, en orden secuencial, `sentencia 4`. Sin embargo, la instrucción `goto`, que representa el ejemplo más sencillo de control de flujo en un programa, provocaría que este "saltara" a la instrucción `etiqueta:`. Precisamente, la traducción literal de `goto` podría ser "ir a", en el contexto de ir a otro punto del código del programa. En otras palabras, la instrucción `goto` provoca un cambio o salto incondicional en el flujo de ejecución del programa, de forma que la siguiente instrucción a ejecutar, `sentencia 5`, sea la que está escrita justo después de la instrucción `etiqueta:`.

```
sentencia 1
sentencia 2
goto etiqueta
sentencia 3
sentencia 4
etiqueta:
sentencia 5
```

Si bien los saltos incondicionales no requieren de condición alguna para ejecutarse, las sentencias de control condicional, por otra parte, se basan en la idea de evaluar una condición o expresión para *disparar* posibles cambios en el flujo de ejecución de un programa. En este sentido, recuerda que las sentencias condicionales son las estructuras más básicas que permiten a un computador actuar tomando decisiones. En el ejemplo de sentencia condicional que ya conoces, "si el perro ladra, entonces encender la iluminación del jardín", la palabra clave `si` introduce una condición a modo de pregunta (¿ladra el perro?). Si la respuesta a esta pregunta es afirmativa, la parte inmediatamente posterior a la palabra clave `entonces` será la acción que se ejecutará.

⚠ ATENCIÓN

El código que abusa del uso de sentencias de salto incondicional puede convertirse en el denominado *código espagueti*, ya que estas sentencias complican la estructura del flujo de ejecución de un programa. Este término deriva de la comparación con un plato de espaguetis, donde el propio código se visualiza como un conjunto de hilos enredados. El espagueti anudado representaría la metáfora del flujo de ejecución del programa. Así, y con carácter general, se desaconseja el uso de sentencias de salto incondicional.

La condición a evaluar en una sentencia condicional suele estar relacionada con la consulta del valor de una determinada variable, o incluso de varias. El siguiente listado de código muestra un fragmento escrito en el lenguaje de programación Python que ejemplifica, de manera sencilla, esta situación. En las líneas 1 y 2 se declaran las variables `a` y `b`, a las que se le asignan, respectivamente, los valores 13 y 7. En la línea 4 aparece una sentencia condicional, representada en Python por la palabra clave `if`. A continuación, aparece la condición a evaluar, `a > b`. Así, si el valor de a es mayor que el valor de b, entonces se ejecutará la sentencia de la línea 5. Si no lo es, es decir, si el valor de a es menor o igual al valor de b, entonces se ejecutará la sentencia de la línea 7. Fíjate cómo en la línea 6 se utiliza otra palabra clave en Python, `else`, para agrupar las sentencias que se ejecutarán cuando la evaluación de la condición equivale a un valor falso.

```
a = 13
b = 7

if a > b:
   print('El valor de a es mayor que el valor de b')
else:
   print('El valor de a es menor o igual que el valor de b')
```

Como puedes imaginar, si ejecutaras este código el resultado sería el siguiente:

```
El valor de a es menor o igual que el valor de b
```

Las sentencias condicionales se pueden complicar, encadenando unas con otras para así cubrir más situaciones.

⚠ ATENCIÓN

El término utilizado para agrupar sentencias condicionales se denomina *anidamiento*. Así, puedes anidar o encadenar varias sentencias condicionales para controlar diferentes situaciones o condiciones, asociando diferentes conjuntos de acciones en función del resultado de la evaluación de la condición que gobierna la sentencia.

En este sentido, el fragmento de código anterior se puede extender tal y como se muestra a continuación. Concéntrate en la sentencia elif de la línea 6. En esencia, lo que le estaríamos indicando al programa es que si la evaluación de la condición anterior, es decir, la de la línea 4, fue falsa, entonces prueba esta nueva condición, definida en la línea 6. Finalmente, y al igual que en el anterior ejemplo, la sentencia else de la línea 8 se utiliza para indicar la acción que se debe ejecutar (línea 9) en caso de que todas las condiciones anteriores se hayan evaluado a falso.

```
a = 13
b = 7

if a > b:
   print('El valor de a es mayor que el valor de b')
elif a == b:
   print('El valor de a es igual al valor de b')
else:
   print('El valor de a es menor que el valor de b')
```

Hay otros lenguajes de programación que agrupan varias sentencias del tipo if-then-else en una única sentencia. Por ejemplo, el lenguaje de programación Java proporciona la sentencia switch, tal y como se muestra en el siguiente ejemplo de código.

```
int dia = 3;
switch (dia) {
  case 1:
    System.out.println("Lunes");
    break;
  case 2:
    System.out.println("Martes");
    break;
  case 3:
    System.out.println("Miércoles");
    break;
  /* Resto de días aquí: Jueves, Viernes, Sábado */
  case 7:
    System.out.println("Domingo");
    break;
}
```

La expresión de la sentencia switch de la línea 2 solo se evalúa una vez. Posteriormente, el valor de dicha expresión se compara con cada uno de los valores de la sentencia switch, vinculados a las diferentes sentencias case que la componen. En este ejemplo, se compara el valor de la variable dia con cada uno de los valores de las sentencias case. Así, el valor de dia se comparará primero con el valor 1, después con el valor 2, etc. Cuando exista una coincidencia, es decir, cuando la comparación sea verdadera, entonces se ejecuta el código que existe a continuación de la sentencia case correspondiente. En este ejemplo, se ejecutará el código de las líneas 10 y 11, de forma que el resultado de ejecutar este programa sería Miércoles.

Por otra parte, y como ya conoces, los lenguajes de programación ofrecen sentencias para repetir fragmentos de código. Normalmente, esta repetición se llevará a cabo mientras una determinada condición sea verdadera. Por ejemplo, y volviendo al lenguaje de programación Python, la sentencia while permite la ejecución de un conjunto de sentencias en tanto en cuanto la condición que la gobierna sea verdadera. El siguiente listado de código muestra un caso concreto.

```
i = 1;

while i <= 5:
    print(i)
    i = i + 1
```

Como puedes apreciar, es relativamente fácil identificar que el código contenido en el bucle while, es decir, el código de las líneas 4 y 5, se ejecutará 5 veces. Esto es equivalente a afirmar que el bucle while realizará 5 iteraciones. La condición que gobierna el bucle equivale a preguntarse en cada iteración si el valor de la variable i es menor o igual que 5. La primera vez que se realiza esta comparación, el valor de la variable i es 1. Como 1 es menor o igual que 5, la condición se evalúa a verdadero y, por lo tanto, el código del bucle (líneas 4 y 5) se ejecuta. Este código muestra el valor de la variable i por pantalla y, aquí viene la parte importante, le suma una unidad al contenido de la variable i. Posteriormente, el flujo de control del programa

vuelve a la línea 3, evaluando, de nuevo, la condición. Ahora, el valor de la variable `i` es 2, que se compara de nuevo con 5. Este proceso se repetirá hasta que la variable `i` contenga el valor 6. En este punto, el bucle `while` no ejecutará más iteraciones.

Finalmente, la inmensa mayoría de los lenguajes de programación incluyen sentencias que permiten escribir bucles con mayor flexibilidad. Una de las más clásicas es la sentencia `for`. Típicamente, este tipo de sentencias permitirá incluir más funcionalidad a la hora de declarar el bucle, como por ejemplo la asignación de valor a variables o incluso la propia actualización de la variable que gobierna la ejecución del bucle. Así, el programador tiene más flexibilidad a la hora de escribir código. Discutamos otro ejemplo en Python.

```
for i in range(1, 6):
    print(i)
```

Este fragmento de código produce exactamente el mismo resultado que el fragmento anterior, en el que se utilizó la sentencia `while`. En otras palabras, el código de la línea 2 se ejecutará 5 veces (el bucle realizará 5 iteraciones). A diferencia del fragmento anterior, la variable `i` se declara en la propia sentencia del bucle `for` de la línea 1. Además, y mediante la sentencia `range`, estamos indicando que el bucle iterará mientras el valor de la variable `i` se encuentre en el rango de 1 a 6, es decir, mientras que el valor de la variable `i` sea 1, 2, 3, 4 o 5. Como puedes apreciar, hemos sido capaces de recrear, con solo 2 líneas de código, la misma funcionalidad que en el listado anterior, que contenía 4 líneas de código (omitiendo la línea en blanco).

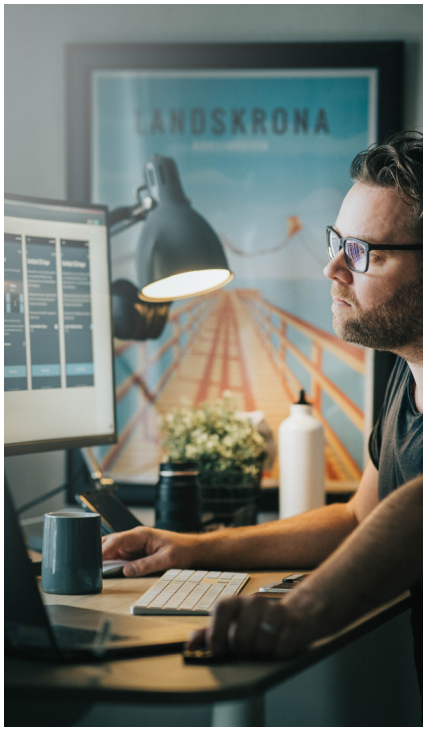
```
var nodeRequire = typeof require === 'function' && require;

function newRequire(name, jumped) {
  if (!cache[name]) {
    if (!modules[name]) {
      // if we cannot find the module within our internal map or
      // cache jump to the current global require ie. the last bundle
      // that was added to the page.
      var currentRequire = typeof parcelRequire === 'function' && parcelRequi
      if (!jumped && currentRequire) {
        return currentRequire(name, true);
```

Creación de
contenidos digitales

Nivel A1 3.4 Programación

Guías de estilo

Guías de estilo



Las **guías de estilo** son documentos que definen un conjunto de convenciones de programación, es decir, un conjunto de buenas prácticas, métodos de programación y otras reglas que garantizan el desarrollo de programas de manera consistente. Por ejemplo, las guías de estilo se encargan de documentar aspectos de los programas relativos a cómo deberíamos nombrar nuestras variables, o si deberíamos utilizar espacios o tabulaciones para indentar el código, entre otros aspectos.

Aunque las guías de estilo suelen utilizarse principalmente durante el desarrollo de proyectos software en equipo, su efectividad en proyectos individuales está más que asegurada; es mucho más sencillo leer y entender el código de un programa que sigue una guía de estilo de manera consistente que el de otro que no lo hace.

Existen multitud de guías de estilos, tantas como lenguajes de programación hay o incluso más de una por lenguaje. Como programadores, podemos elegir la guía de estilo que prefiramos en función del soporte existente, la documentación y la facilidad de adopción. Lo importante es que seamos consistentes con la guía de estilo que elijamos para el desarrollo de nuestro programa y no mezclemos diferentes guías de estilo para un mismo proyecto.

Entre los beneficios que nos aportan las guías de estilo, podemos destacar los siguientes:

- Mejoran la legibilidad del código, facilitando su lectura y comprensión.
- Facilita que otros desarrolladores puedan realizar contribuciones a nuestros programas. Del mismo modo, nos permiten implicarnos rápidamente en el desarrollo de otros programas que usen guías de estilo con las que ya estemos familiarizados.
- Nos ayudan a centrarnos en lo que realmente importa, es decir, desarrollar nuestros programas.

Un caso de guía de estilo sería la **PEP 8** o **Guía de Estilo para Código Python**, creada por el propio autor del lenguaje Python, Guido van Rossum. Esta guía de estilo proporciona indicaciones sobre la organización del código, nombrado de variables y otros elementos del lenguaje, uso de comas o entrecomillado de cadenas de texto, entre otros.

Por ejemplo, PEP 8 aconseja nombrar las variables y funciones en minúsculas y separando las distintas palabras con guiones bajos. Algunos ejemplos válidos de nombres de variables que se ajustan a lo que indica esta regla serían `numero_de_aprobados` o `contador`, mientras que algunos ejemplos incorrectos serían `numeroDeAprobados` o `CONTADOR`.

Además de las guías de estilo, podemos también mejorar la legibilidad de nuestros programas y su mantenibilidad futura documentando aquellas partes del código más complejas o en las que consideremos que merece la pena hacer hincapié. Añadiendo comentarios al código podemos realizar esta documentación directamente sobre el propio código del programa, de tal forma que cualquier programador pueda revisar dichos comentarios durante la etapa de desarrollo.

En este contexto, la guía de estilo PEP 8 también da algunas indicaciones sobre el uso de comentarios en el código como, por ejemplo, añadir comentarios que realmente aporten algo al lector y no expongan obviedades, o documentar utilizando comentarios en inglés, entre otras:

```
# Improves the accuracy by 5% to reduce possible detection errors
x = x * 1.05
```

DigitAll

Formación en
Competencias
Digitales

https://www.digitall.org.es/

DigitAll

Coordinación General

Universidad de Castilla-La Mancha
Carlos González Morcillo
Francisco Parreño Torres

Coordinadores de área

Área 1. Búsqueda y gestión de información y datos

Universidad de Zaragoza
Francisco Javier Fabra Caro

Área 2. Comunicación y colaboración

Universidad de Sevilla
Francisco Javier Fabra Caro
Francisco de Asís Gómez Rodríguez
José Mariano González Romano
Juan Ramón Lacalle Remigio
Julio Cabero Almenara
María Ángeles Borrueco Rosa

Área 3. Creación de contenidos digitales

Universidad de Castilla-La Mancha
David Vallejo Fernández
Javier Alonso Albusac Jiménez
José Jesús Castro Sánchez

Área 4. Seguridad

Universidade da Coruña
Ana M. Peña Cabanas
José Antonio García Naya
Manuel García Torre

Área 5. Resolución de problemas

UNED
Jesús González Boticario

Coordinadores de nivel

Nivel A1

Universidad de Zaragoza
Ana Lucía Esteban Sánchez
Francisco Javier Fabra Caro

Nivel A2

Universidad de Córdoba
Juan Antonio Romero del Castillo
Sebastián Rubio García

Nivel B1

Universidad de Sevilla
Francisco de Asís Gómez Rodríguez
José Mariano González Romano
Juan Ramón Lacalle Remigio
Montserrat Argandoña Bertran

Nivel B2

Universidad de Castilla-La Mancha
María del Carmen Carrión Espinosa
Rafael Casado González
Víctor Manuel Ruiz Penichet

Nivel C1

UNED
Antonio Galisteo del Valle

Nivel C2

UNED
Antonio Galisteo del Valle

Maquetación

Universidad de Salamanca
Fernando De la Prieta Pintado
Pilar Vega Pérez
Sara Alejandra Labrador Martín

Creadores de contenido

Área 1. Búsqueda y gestión de información y datos

1.1 Navegar, buscar y filtrar datos, información y contenidos digitales

Universidad de Huelva
Ana Duarte Hueros (coord.)
Arantxa Vizcaíno Verdú
Carmen González Castillo
Dieter R. Fuentes Cancell
Elisabetta Brandi
José Antonio Alfonso Sánchez
José Ignacio Aguaded
Mónica Bonilla del Río
Odiel Estrada Molina
Tomás de J. Mateo Sanguino (coord.)

1.2 Evaluar datos, información y contenidos digitales

Universidad de Zaragoza
Ana Belén Martínez Martínez
Ana María López Torres
Francisco Javier Fabra Caro
José Antonio Simón Lázaro
Laura Bordonaba Plou
María Sol Arqued Ribes
Raquel Trillo Lado

1.3 Gestión de datos, información y contenidos digitales

Universidad de Zaragoza
Ana Belén Martínez Martínez
Francisco Javier Fabra Caro
Gregorio de Miguel Casado
Sergio Ilarri Artigas

Área 2. Comunicación y colaboración

2.1 Interactuar a través de tecnología digitales

Iseazy

2.2 Compartir a través de tecnologías digitales

Universidad de Sevilla
Alién García Hernández
Daniel Agüera García
Jonatan Castaño Muñoz
José Candón Mena
José Luis Guisado Lizar

2.3 Participación ciudadana a través de las tecnologías digitales

Universidad de Sevilla
Ana Mancera Rueda
Félix Biscarri Triviño
Francisco de Asís Gómez Rodríguez
Jorge Ruiz Morales
José Manuel Sánchez García
Juan Pablo Mora Gutiérrez
Manuel Ortigueira Sánchez
Raúl Gómez Bizcocho

2.4 Colaboración a través de las tecnologías digitales

Universidad de Sevilla
Belén Vega Márquez
David Vila Viñas
Francisco de Asís Gómez Rodríguez
Julio Barroso Osuna
María Puig Gutiérrez
Miguel Ángel Olivero González
Óscar Manuel Gallego Pérez
Paula Marcelo Martínez

2.5 Comportamiento en la red

Universidad de Sevilla
Ana Mancera Rueda
Eva Mateos Núñez
Juan Pablo Mora Gutiérrez
Óscar Manuel Gallego Pérez

2.6 Gestión de la identidad digital

Iseazy

Área 3. Creación de contenidos digitales

3.1 Desarrollo de contenidos

Universidad de Castilla-La Mancha
Carlos Alberto Castillo Sarmiento
Diego Cordero Contreras
Inmaculada Ballesteros Yáñez
José Ramón Rodríguez Rodríguez
Rubén Grande Muñoz

3.2 Integración y reelaboración de contenido digital

Universidad de Castilla-La Mancha
José Ángel Martín Baos
Julio Alberto López Gómez
Ricardo García Ródenas

3.3 Derechos de autor (copyright) y licencias de propiedad intelectual

Universidad de Castilla-La Mancha
Gabriela Raquel Gallicchio Platino
Gerardo Alain Marquet García

3.4 Programación

Universidad de Castilla-La Mancha
Carmen Lacave Rodero
David Vallejo Fernández
Javier Alonso Albusac Jiménez
Jesús Serrano Guerrero
Santiago Sánchez Sobrino
Vanesa Herrera Tirado

Área 4. Seguridad

4.1 Protección de dispositivos

Universidade da Coruña
Antonio Daniel López Rivas
José Manuel Vázquez Naya
Martiño Rivera Dourado
Rubén Pérez Jove

4.2 Protección de datos personales y privacidad

Universidad de Córdoba
Aida Gema de Haro García
Ezequiel Herruzo Gómez
Francisco José Madrid Cuevas
José Manuel Palomares Muñoz
Juan Antonio Romero del Castillo
Manuel Izquierdo Carrasco

4.3 Protección de la salud y del bienestar

Universidade da Coruña
Javier Pereira Loureiro
Laura Nieto Riveiro
Laura Rodríguez Gesto
Manuel Lagos Rodríguez
María Betania Groba González
María del Carmen Miranda Duro
Nereida María Canosa Domínguez
Patricia Concheiro Moscoso
Thais Pousada García

4.4 Protección medioambiental

Universidad de Córdoba
Alberto Membrillo del Pozo
Alicia Jurado López
Luis Sánchez Vázquez
María Victoria Gil Cerezo

Área 5. Resolución de problemas

5.1 Resolución de problemas técnicos

Iseazy

5.2 Identificación de necesidades y respuestas tecnológicas

Iseazy

5.3 Uso creativo de la tecnología digital

Iseazy

5.4 Identificar lagunas en las competencias digitales

Iseazy

Financiado por la Unión Europea NextGenerationEU
Plan de Recuperación, Transformación y Resiliencia