

Python

TechGuide - Alura, FIAP e PM3

Nivel 1

Python - Fundamentos:

- Python es un lenguaje de programación de alto nivel de uso general, ampliamente utilizado en aplicaciones web, desarrollo de software, ciencia de datos y aprendizaje automático. Su filosofía de diseño se centra en la legibilidad del código mediante el uso de sangría significativa. Python es de tipado dinámico y cuenta con un recolector de basura.
- Los tipos de datos primitivos.
- Declarar variables, teniendo en cuenta los diferentes tipos.
- Utilizar estructuras condicionales ('if', 'else').
- Conocer los operadores de asignación y comparación.
- Usar estructuras de repetición y bucles ('while', 'for').
- Utilizar funciones, pasando parámetros y argumentos.
- Manipular métodos.
- Manipular arrays y listas.
- Obtener datos de una API.
- Crear constructores.
- Utilizar funciones anónimas.

Contenidos

- **Web** Documentación Python (<https://docs.python.org/es/3/tutorial/>)
- **Web** ¿Qué es Python? - AWS (<https://aws.amazon.com/es/what-is/python/>)
- **Artículo** El Manual de Python (<https://www.freecodecamp.org/espanol/news/el-manual-de-python/>)

- **Artículo** Variables y tipos de datos básicos en Python
(<https://soka.gitlab.io/blog/post/2022-07-19-python-vars-intro-numbers/>)
- **Artículo** Sentencia If Else de Python: Explicación de las sentencias condicionales
(<https://www.freecodecamp.org/espanol/news/sentencia-if-else-de-python-explicacion-de-las-sentencias-condiciones/>)
- **Artículo** Pasos iniciales para utilizar la biblioteca Requests de Python
(<https://www.digitalocean.com/community/tutorials/how-to-get-started-with-the-requests-library-in-python-es>)
- **YouTube** Manejo de entrada y salida de datos | Curso de Python desde cero 🐍
(<https://www.youtube.com/watch?v=Sei1sltfocw>)
- **YouTube** CURSO de PYTHON 2020 🐍 CONSTRUCTORES
(<https://www.youtube.com/watch?v=ElObT5rOx2M>)
- **YouTube** Curso Python 🐍 | Tipos de datos en Python, Enteros, Float, String, Booleanos
  (https://www.youtube.com/watch?v=R3eQs-AsB_w)

Contenidos Alura:

- **Artículo** Python - Una introducción al Lenguaje
(<https://www.aluracursos.com/blog/python-una-introduccion-al-lenguaje>)
- **Artículo** ¿Qué es Python? Historia, sintaxis y una guía para iniciarse en el lenguaje
(<https://www.aluracursos.com/blog/que-es-python-historia-guia-para-iniciar>)
- **Artículo** Trabajando con precisión en números decimales en Python
(<https://www.aluracursos.com/blog/precision-numeros-decimales-python>)
- **Artículo** Python datetime: trabajando con fechas
(<https://www.aluracursos.com/blog/python-datetime-trabajando-con-fechas>)
- **Artículo** Listas en Python: operaciones básicas
(<https://www.aluracursos.com/blog/listas-de-python-operaciones-basicas>)
- **Artículo** ¿Cómo comparar objetos en Python?
(<https://www.aluracursos.com/blog/como-comparar-objetos-en-python>)
- **YouTube** Descubre el poder de Python: El lenguaje de programación que revoluciona la industria (<https://www.youtube.com/watch?v=BxcMMgmLKTU>)
- **Curso** Curso de Python: comenzando con el lenguaje
(<https://www.aluracursos.com/curso-online-python-comenzando-con-lenguaje>)
- **Curso** Curso de Python: funciones incorporadas
(<https://app.aluracursos.com/course/python-funciones-incorporadas>)



- **Curso** Formación Aprenda a programar en Python con orientación a objetos (<https://www.aluracursos.com/formacion-lenguaje-python>)

Conceptos de Orientación a Objetos:



- La Programación Orientada a Objetos es un paradigma de programación de software basado en la composición e interacción entre diversas unidades llamadas 'objetos' y las clases, que contienen una identidad, propiedades y métodos. Se basa en cuatro componentes de la programación - abstracción digital, encapsulación, herencia y polimorfismo.
- Cómo funcionan los objetos
- Crear y utilizar constructores
- Qué son las clases
- Crear y utilizar métodos
- Cómo funciona la encapsulación
- Qué es la herencia
- Qué es el polimorfismo
- Cómo funcionan las interfaces
- Qué son las abstracciones

Contenidos

- **Web** Programación orientada a objetos (C#) (<https://learn.microsoft.com/es-es/dotnet/csharp/fundamentals/tutorials/oop>)
- **Web** Programación orientada a objetos - IBM (<https://www.ibm.com/docs/es/spss-modeler/saas?topic=language-object-oriented-programming>)
- **Web** Introducción a los objetos JavaScript (<https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects>)
- **Artículo** Introducción a POO con JavaScript ES6 (<https://medium.com/academia-hack/introducci%C3%B3n-a-poo-con-javascript-es6-80074fde0cdf>)
- **Artículo** Programación Orientada a Objetos (<https://ellibrodepython.com/programacion-orientada-a-objetos-python>)
- **YouTube** 4 Principios de la Programación Orientada a Objetos (<https://www.youtube.com/watch?v=Uk1C1NARZjU>)
- **YouTube** Mini Curso: POO con PHP (básico) (https://youtu.be/Ben_VC2rm10)

-   ¿Qué es la Programación Orientada a Objetos en PYTHON? - [Con EJEMPLOS] | Python desde CERO #13 (<https://www.youtube.com/watch?v=KwT1F7uL5rA>)

Contenidos Alura:

-  Alura Latam: ¿Qué es la Programación Orientada a Objetos? (<https://www.youtube.com/watch?v=Oigen2sjagk>)
-  POO: ¿Qué es la programación orientada a objetos? (<https://www.aluracursos.com/blog/poo-que-es-la-programacion-orientada-a-objetos>)
-  Revisando la Orientación a Objetos: encapsulación de Java (<https://www.aluracursos.com/blog/revisando-la-orientacion-a-objetos-encapsulacion-de-java>)
-  ¿Qué es encapsulamiento? (<https://www.aluracursos.com/blog/Que-es-encapsulamiento>)
-  Herencia en JavaScript (<https://www.aluracursos.com/blog/herencia-en-javascript>)
-  Interfaces Gráficas con Eclipse WindowBuilder (<https://www.aluracursos.com/blog/interfaces-graficas-con-eclipse-windowbuilder>)
-  Cómo no aprender Java y Orientación a Objetos: getters y setters (<https://www.aluracursos.com/blog/como-no-aprender-java-y-orientacion-a-objetos-getters-y-setters>)
-  Ordenar una lista de objetos en Java (<https://www.aluracursos.com/blog/ordenar-una-lista-de-objetos-en-java>)
-  Formación Java Orientado a Objetos (<https://app.aluracursos.com/formacion-javaoo>)
-  JavaScript: Introducción a la Orientación a Objetos (<https://app.aluracursos.com/course/javascript-introduccion-orientacion-objetos>)
-  JavaScript: Herencia e Interfaces en Orientación a Objetos (<https://app.aluracursos.com/course/javascript-herencia-interfaces-orientacion-objetos>)
-  C# y Orientación a Objetos (<https://app.aluracursos.com/formacion-c-sharpe-orientacion-a-objetos>)
-  Python: comprensión de la Orientación a Objetos (<https://app.aluracursos.com/course/python-comprension-orientacion-objetos>)

- **Curso** Python: avanzando en la orientación a objetos (<https://app.aluracursos.com/course/python-avanzando-orientacion-objetos>)
- **Curso** Java Polimorfismo: Entendiendo herencia e interfaces (<https://app.aluracursos.com/course/java-parte-3-entendiendo-herencia-interfaces>)
- **Curso** Curso de JavaScript: Objetos (https://app.aluracursos.com/course/javascript-objetos?utm_source=gnarus&utm_medium=timeline)
- **Curso** Formación: Aprenda a programar en Python con orientación a objetos (<https://app.aluracursos.com/formacion-lenguaje-python>)

Estructura de Datos:

- En el contexto de los ordenadores, una estructura de datos es una forma específica de almacenar y organizar los datos en la memoria del ordenador para que esos datos puedan ser fácilmente recuperados y utilizados de forma eficiente cuando sea necesario posteriormente.
- Conocer las principales estructuras de datos
- Implementar las principales estructuras de datos

Contenidos

- **Artículo** ¿Qué es una estructura de datos en programación y para qué se utiliza? (<https://blog.soyhenry.com/que-es-una-estructura-de-datos-en-programacion/>)
- **Web** Estructuras de datos - Documentación Python (<https://docs.python.org/es/3/tutorial/datastructures.html>)
- **Artículo** Estructuras de Datos y Algoritmos en Java (https://programacion.net/articulo/estructuras_de_datos_y_algoritmos_en_java_309/4)
- **Artículo** Estructuras de datos con Java: un enfoque práctico (<http://hp.fciencias.unam.mx/~alg/estructurasDeDatos/>)
- **Artículo** Colecciones: ARRAY, SET y DICTIONARY en Swift (<https://www.swiftbeta.com/colecciones-array-set-y-dictionary-en-swift/>)
- **Artículo** Colecciones en Swift y su manejo de memoria (<https://medium.com/@grago/colecciones-en-swift-y-su-manejo-de-memoria-61a03e236dd>)
- **Artículo** Estructuras de datos en .NET con C# (<https://www.genesisrrios.com/es/blog/estructuras-de-datos-en-csharp/>)
- **YouTube** Estructuras de Datos | Primeros Pasos (<https://youtu.be/Df-sgxGzyTg>)

- **YouTube** Qué son las estructuras de datos (<https://youtu.be/oQ0WklDr73E?list=PLTd5ehlj0goMTSK7RRAPBF4wP-Nj5DRvT>)
- **YouTube** ¿Qué son y cómo funcionan los árboles? | Ejemplo de implementación (<https://youtu.be/tBaQQeyXYgg>)
- **YouTube** Aprende: Estructura de Datos con Java (https://www.youtube.com/watch?v=_9ScDWpghFE)
- **YouTube** Introducción - 1 - Estructuras de Datos en C# (https://www.youtube.com/watch?v=rqaqjXbauyA&ab_channel=nicosioired)
- **YouTube** Estructuras de datos con Python en 8 minutos: Listas, Tuplas, Conjuntos y Diccionarios (<https://www.youtube.com/watch?v=v25-m1LOUiU>)
- **YouTube** Programación en Python | Colecciones | Diccionarios (<https://www.youtube.com/watch?v=vAy4IM7NLIQ>)

Contenidos Alura:

- **Artículo** Estructura de datos: introducción (<https://www.aluracursos.com/blog/estructura-de-datos-introduccion>)
- **Artículo** Conociendo tuplas en Python (<https://www.aluracursos.com/blog/conociendo-las-tuplas-en-python>)
- **Artículo** Listas en Python: operaciones básicas (<https://www.aluracursos.com/blog/listas-de-python-operaciones-basicas>)
- **Artículo** Estructura de datos: computación práctica con Java (<https://www.aluracursos.com/blog/estructura-de-datos-computacion-practica-con-java>)
- **Artículo** Python: trabajando con diccionarios (<https://www.aluracursos.com/blog/python-trabajando-con-diccionarios>)
- **Artículo** Iterando una lista en Java (<https://www.aluracursos.com/blog/iterando-una-lista-en-java>)
- **Curso** Python: avanzando en el lenguaje (<https://app.aluracursos.com/course/python-avanzando-lenguaje>)
- **Curso** C#: array y tipos genéricos (<https://app.aluracursos.com/course/csharp-array-tipos-genericos>)
- **Curso** Python Collections: listas y tuplas (https://app.aluracursos.com/course/python-collections-listas-tuplas?utm_source=gnarus&utm_medium=timeline)
- **Curso** Curso de Java: trabajar con listas y colecciones de datos (<https://www.aluracursos.com/curso-online-java-trabajar-listas-colecciones-datos>)

- **Curso** Java y java.util: Colecciones, Wrappers y Lambda expressions (<https://www.aluracursos.com/curso-online-java-util-colecciones-wrappers-lambda-expressions>)

Python - Colecciones:

- Una colección representa un grupo de objetos, conocidos como sus elementos. Son como contenedores que agrupan varios elementos en una única unidad. Algunas colecciones permiten la duplicación de elementos y otras no. Algunas están ordenadas y otras no lo están.
- Aprender a usar listas y tuplas
- Aprovechar el polimorfismo en las colecciones
- Utilizar conjuntos y diccionarios.

Contenidos

- **Artículo** Listas en Python, uso y creación (<https://www.linkedin.com/pulse/listas-en-python-uso-y-creación-alejandro-alomia/?originalSubdomain=es>)
- **Artículo** Conjuntos, sets en Python (<https://atareao.es/pyldora/conjuntos-sets-en-python/>)
- **Artículo** Diccionarios, Tuplas y Sets (<https://makeitrealcamp.gitbook.io/guias-de-make-it-real/python/diccionarios-tuplas-y-sets>)
- **YouTube** Cómo hacer LISTAS 📁 en Python 🐍 #05 [Curso Python Data Science Español] (<https://www.youtube.com/watch?v=z0xmmJ3BEPE>)
- **YouTube** Sets - 19 - Python tutorial en español (<https://www.youtube.com/watch?v=s3xbRvBaZcs>)
- **YouTube** Diccionario de funciones - 28 - Python Intermedio tutorial en español (<https://www.youtube.com/watch?v=fkqeQUiVN-Q>)
- **YouTube** Listas, Tuples, Sets, Strings y Diccionarios en PYTHON (<https://www.youtube.com/watch?v=CCUNuqqn7PQ>)

Contenidos Alura:

- **Artículo** Conociendo tuplas en Python (<https://www.aluracursos.com/blog/conociendo-las-tuplas-en-python>)
- **Artículo** Python: trabajando con diccionarios (<https://www.aluracursos.com/blog/python-trabajando-con-diccionarios>)

- **Artículo** Cómo hacer una copia de una lista en Python (<https://www.aluracursos.com/blog/como-hacer-una-copia-de-una-lista-en-python>)
- **Artículo** Comprensión de listas en Python (<https://www.aluracursos.com/blog/comension-de-listas-en-python>)
- **Artículo** Ordenar listas en Python (<https://www.aluracursos.com/blog/ordenando-listas-en-python>)
- **Artículo** ¿Append o Extend? Agregar elementos a la lista con Python (<https://www.aluracursos.com/blog/append-o-extend-agregar-elementos-a-la-lista-con-python>)
- **Curso** Curso de Python Collections: colecciones y diccionarios (<https://www.aluracursos.com/curso-online-python-collections-colecciones-diccionarios>)
- **Curso** Curso de Python Collections: listas y tuplas (<https://www.aluracursos.com/curso-online-python-collections-listas-tuplas>)

Python - Pruebas:

- Las pruebas de software son el proceso de evaluación y verificación de que un software realmente hace lo que debería hacer. Los beneficios de las pruebas incluyen la prevención de errores, la reducción de los costos de desarrollo y la mejora del rendimiento
- Pruebas unitarias
- Pruebas de integración
- Pruebas de comportamiento (behavior)
- Mocks

Contenidos

- **Web** Configuración de pruebas unitarias para código de Python (<https://learn.microsoft.com/es-es/visualstudio/python/unit-testing-python-in-visual-studio?view=vs-2022>)
- **Web** unittest — Framework de pruebas unitarias (<https://docs.python.org/es/dev/library/unittest.html>)
- **Artículo** Proyectos de prueba en Python (https://www.linkedin.com/pulse/proyectos-de-prueba-en-python-calltek/?trk=article-ssr-frontend-pulse_more-articles_related-content-card&originalSubdomain=es)

- **Artículo** Probando una aplicación web en Python - Parte 0: Introducción (<https://medium.com/contraslashsas/probando-una-aplicación-web-en-python-parte-0-9273f5a25938>)
- **Artículo** Pruebas unitarias con el módulo unittest de Python (<https://geekflare.com/es/unit-testing-with-python-unittest/>)
- **Artículo** Testing en lenguaje natural con Gherkin y Behave (<https://blog.adrianistan.eu/testing-lenguaje-natural-behave-python>)
- **Artículo** Automatización de Pruebas con Selenium y Python (<https://www.enmilocalfunciona.io/automatizacion-de-pruebas-con-selenium-y-python/>)
- **Artículo** Cómo acelerar los ciclos de prueba lentos con Selenium (<https://hackernoon.com/es/como-acelerar-ciclos-de-prueba-lentos-con-selenium>)
- **YouTube** Creación de datos mock con Python (<https://www.youtube.com/watch?v=iIZX8sxv3SU>)
- **YouTube** Probando Python con mocks (simulaciones). Monkey patching, herencia en mocks y librería Doublex (<https://www.youtube.com/watch?v=t3C6kfHwrOE>)
- **YouTube** Python Avanzado: Pruebas de Integración y Funcionales | Unittest y Pytest (https://www.youtube.com/watch?v=0t_aFxGCcEs)
- **YouTube** TDD - Test Driving Development (Pruebas Unitarias) en Python (<https://www.youtube.com/watch?v=cjN-XYHtmfI>)

Contenidos Alura:

- **Artículo** Tipos de pruebas: ¿cuáles son las principales y por qué utilizarlas? (<https://www.aluracursos.com/blog/tipos-de-pruebas-cual-utilizar>)

Python - Comunicación con APIs:

- Una API es una interfaz que los desarrolladores de software utilizan para programar la interacción con componentes o recursos de software fuera de su propio código. Una definición aún más simple es que una API es la parte de un componente de software que es accesible para otros componentes.
- Comprender qué es una API REST
- Conocer los comandos básicos de comunicación HTTP
- Entender qué es una API REST
- Saber cómo hacer solicitudes autenticadas
- Convertir objetos a JSON y viceversa

- Saber cómo utilizar las herramientas del paquete Requests.

Contenidos

- **Web** ¿Qué es una API? - AWS (<https://aws.amazon.com/es/what-is/api/>)
- **Web** ¿Cuál es la diferencia entre SDK y API? - AWS (<https://aws.amazon.com/es/compare/the-difference-between-sdk-and-api/>)
- **Web** API de correo electrónico para Python 3 (<https://cloud.google.com/appengine/docs/standard/python3/services/mail?hl=es-419>)
- **Web** ¿Qué es una API REST? (<https://www.ibm.com/es-es/topics/rest-apis>)
- **YouTube** qué son los APIs? (SOAP, REST, GraphQL) (<https://www.youtube.com/watch?v=rAylamS1Hco>)
- **YouTube** APRENDE A CONSUMIR LA API DE SPOTIFY EN 15 MINUTOS!! (<https://www.youtube.com/watch?app=desktop&v=9cz8Gvh0J7g>)
- **YouTube** Implementación de una API en Python para enviar email con Angular (https://www.youtube.com/watch?v=RSGQuH_gWNw)
- **YouTube** Automatizando API con Python Capítulo 1: Introducción (<https://www.youtube.com/watch?v=6r2M1FEMauw>)

Contenidos Alura:

- **Artículo** Buscando tweets con Python (<https://www.aluracursos.com/blog/buscando-tweets-con-python>)

Nivel 2

Flask:

- Flask es un pequeño framework web escrito en Python. Se clasifica como un microframework porque no requiere herramientas o bibliotecas particulares, manteniendo un núcleo simple pero extensible. No posee una capa de abstracción de base de datos, validación de formularios u otros componentes, ya que bibliotecas de terceros proporcionan funciones comunes. Sin embargo, Flask ofrece soporte a extensiones que pueden agregar recursos a la aplicación como si fueran implementados en Flask mismo
- Crear aplicaciones web
- Definir rutas, redirecciones y plantillas

- Validar formularios

Contenidos

- **Web** Documentación Flask (<https://flask-es.readthedocs.io>)
- **Web** Cómo crear una aplicación Web usando Flask en Python 3 (<https://www.digitalocean.com/community/tutorials/how-to-make-a-web-application-using-flask-in-python-3-es>)
- **Web** Implementación de una aplicación Flask en Elastic Beanstalk (https://docs.aws.amazon.com/es_es/elasticbeanstalk/latest/dg/create-deploy-python-flask.html)
- **Web** Tutorial: Introducción al marco web de Flask en Visual Studio (<https://learn.microsoft.com/es-es/visualstudio/python/learn-flask-visual-studio-step-01-project-solution?view=vs-2022>)
- **Web** Instructivo de la app de Flask para Python (<https://cloud.google.com/build/docs/samples/cloudbuild-python-flask?hl=es-419>)
- **Artículo** Tutorial para crear una aplicación web con Python, Flask y Angular (<https://alexmarket.medium.com/aplicaciones-web-con-python-flask-y-angular-292f889b4ba8>)
- **Artículo** Aplicación web basada en Flask y MySQL, y despliegue con Docker y docker-compose en Digital Ocean (<https://jaimesendraberenguer.medium.com/aplicación-web-basada-en-flask-y-mysql-y-despliegue-con-docker-y-docker-compose-en-digital-ocean-4754a400d4e3>)
- **Artículo** Tutorial de Flask en español: Desarrollando una aplicación web en Python (<https://j2logo.com/tutorial-flask-espanol/>)
- **Artículo** 5 maneras de ejecutar middlewares en Flask (<https://nelsoncode.medium.com/5-maneras-de-ejecutar-middlewares-en-flask-6622bb294bcb>)
- **Artículo** Flask vs Django: Elijamos Tu Próximo Framework Python (<https://kinsta.com/es/blog/flask-vs-django/>)
- **YouTube** Flask Introducción ¿Como funciona? (<https://www.youtube.com/watch?v=2eoEgs5oLxY>)
- **YouTube** Como Crear Una Página Web Muy Fácil Con Python & Flask (<https://www.youtube.com/watch?v=TK6DIBI63aI>)
- **YouTube** Crea Un API Con Python (<https://www.youtube.com/watch?v=b0ZrmhyCY4>)

- **YouTube** Creando una api flask python (<https://www.youtube.com/watch?v=6Otaw6XhXf0>)

Python - Programación Orientada a Objetos Avanzada:

- Mixin es una clase que ofrece implementación de métodos para ser reutilizados por múltiples clases hijas relacionadas.
- Sobrecarga de operador significa dar un significado extendido además de su significado operacional predefinido

Contenidos

- **Web** Clases Base Abstractas para Contenedores (<https://docs.python.org/es/3.11/library/collections.abc.html>)
- **Web** Model Clase (<https://learn.microsoft.com/es-es/python/api/msrest/msrest.serialization.model?view=azure-python-preview>)
- **Artículo** Herencia múltiple en Python (<https://www.delftstack.com/es/howto/python/multiple-inheritance-in-python/>)
- **Artículo** Herencia múltiple y «mixins» (https://fernandoruizrico.com/programacion-con-python-clases-y-objetos/#Herencia_multiple_y_«mixins»)
- **Artículo** Cómo usar la inyección de dependencias en Python (<https://medium.com/@xescuder/cómo-usar-la-inyección-de-dependencias-en-python-3de19ce7759c>)
- **Artículo** ¿Clases abstractas o duck typing? (<https://medium.com/@xescuder/implementamos-en-python-con-clases-abstractas-o-con-duck-typing-1e0af2bf1a9b>)
- **Artículo** ¿Qué son los “mixin” en Python y como debería usarlos? (<https://sergio1998.medium.com/qué-son-los-mixin-en-python-y-como-debería-usarlos-8b6e3a4a5755>)
- **Artículo** (Overloading) — Sobrecargar Operadores en Python (<https://medium.com/@LuisMBaezCo/overloading-sobrecargar-operadores-en-python-5d7a75e2bfdf>)
- **Artículo** Sobrecarga de funciones o despacho múltiple en Python (<https://recursospython.com/guias-y-manuales/sobrecarga-de-funciones-o-despacho-multiple/>)
- **Artículo** Sobrecarga de funciones en Python (<https://www.delftstack.com/es/howto/python/python-function-overloading/>)

- **YouTube** Sobrecarga de Métodos (<https://www.youtube.com/watch?v=Hj7fvkHr71s>)
- **YouTube** Sobrecarga de Operadores (<https://www.youtube.com/watch?v=emBcPZUtx3o>)

Contenidos Alura:

- **Curso** Curso de Python: avanzando en la orientación a objetos (<https://www.aluracursos.com/curso-online-python-avanzando-orientacion-objetos>)

Django:

- Django es un framework web de alto nivel en Python que permite el rápido desarrollo de sitios web seguros y de fácil mantenimiento.
- Crear una aplicación web
- Comprender la arquitectura de una aplicación desarrollada con Django
- Crear el panel de administración de una página
- Utilizar plantillas y rutas
- Crear formularios

Contenidos

- **Web** Documentación de Django (<https://docs.djangoproject.com/es/4.1/>)
- **Web** Introducción a Django (<https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Introduction>)
- **Web** Framework Web Django (<https://developer.mozilla.org/es/docs/Learn/Server-side/Django>)
- **Web** ¿Qué es Django? (<https://aws.amazon.com/es/what-is/django/>)
- **Web** Tutorial Django: El Sitio Web de La Biblioteca Local (https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Tutorial_local_library_website)
- **Artículo** Tutorial básico de Django (<https://medium.com/@crisnieromero/tutorial-básico-de-django-parte-1-introducción-y-preparación-del-entorno-c3f0031a6ca2>)
- **Artículo** ¿Para qué se utiliza Django de Python? 5 Razones claves por las que uso el Framework Django para Proyectos de Clientes (<https://www.freecodecamp.org/espanol/news/para-que-se-utiliza-django-de-python-5-razones-claves-por-las-que-uso-el-framework-django-para-proyectos/>)
- **Artículo** Flask vs Django en 2022: ¿Qué marco de Python elegir? (<https://cynoteck.com/es/blog-post/flask-vs-django/>)

- **Artículo** Comenzando con Django (<https://medium.com/@devsar/comenzando-con-django-e10b26ce306f>)
- **Artículo** Modelos en Django (<https://medium.com/@devsar/modelos-en-django-381530c5fc3c>)
- **Artículo** Forms en Django (<https://medium.com/@devsar/clase-03-forms-2d827e3ca677>)
- **Artículo** Templates y recursos estáticos (<https://medium.com/@devsar/clase-04-templates-y-recursos-estaticos-bcee8d7d1c8>)
- **Artículo** Diferencias entre default, null y blank en Django (<https://frankgalandev.com/diferencias-entre-default-null-y-blank-en-django/>)
- **Artículo** ORM de Django y QuerySets (https://tutorial.djangogirls.org/es/django_orm/)
- **YouTube** ¿Qué es Django y porque importa? - Django y Django REST Framework (<https://www.youtube.com/watch?v=B8OIRdYwNIA>)
- **YouTube** Django Python Primeros Pasos (<https://www.youtube.com/watch?v=VzIqWa4I7yU>)
- **YouTube** Cómo crear un API REST con Django (https://www.youtube.com/watch?v=XqRBb_4CLS4)

Django Rest Framework:

- Django REST Framework es un conjunto de herramientas poderosas y flexibles para la construcción de APIs.
- Desarrollar APIs
- Trabajar con modelos, serializadores y vistas
- Incluir filtros, búsquedas y ordenación
- Limitar el número de solicitudes

Contenidos

- **Web** Django REST framework (<https://www.django-rest-framework.org>)
- **Artículo** Introducción a Django REST framework (<https://www.paradigmadigital.com/dev/introduccion-django-rest-framework/>)
- **Artículo** Entendiendo Django Rest Framework (<https://4geeks.com/es/lesson/django-rest-framework-es>)

- **Artículo** Construir un API REST con Django REST Framework y APIView (<https://davidcasr.medium.com/construir-un-api-rest-con-django-rest-framework-y-apiview-5ea4b2823307>)
- **Artículo** Proyecto API con Django Rest Framework (<https://docs.hektorprofe.net/academia/django/api-rest-framework/>)
- **Artículo** Testeando endpoints con Django Rest Framework (<https://dev.to/nahuelsegovia/testeando-endpoints-con-django-rest-framework-1nbm>)
- **Artículo** Registro y autenticación con Django Rest Framework (<https://cosasdedevs.com/posts/registro-y-autenticacion-con-django-rest-framework/>)
- **Artículo** Algunos tips de Django Rest Framework que quizás no conocías (<https://jairoandres.com/algunos-tips-de-drf-que-quizas-no-conocias/>)
- **YouTube** Introducción a Django Rest Framework (<https://www.youtube.com/watch?v=MMFBD2Eoeuk>)
- **YouTube** Blog con Django Rest Framework + VueJS (<https://www.youtube.com/watch?v=A12qkUvrzK4>)
- **YouTube** Tu Propio Admin de Django (https://www.youtube.com/watch?v=RuDkWKmO9rM&list=PLMbRqrU_kvRzgD2s7JHvJxGs6FdvFjg9&index=14)
- **YouTube** Django Rest Framework | Autenticacion via Token - Login y Logout (https://www.youtube.com/watch?v=kh4YFQrvVyE&list=PLMbRqrU_kvRzgD2s7JHvJxGs6FdvFjg9&index=7)
- **YouTube** Creando una aplicación con Django REST Framework y VUE js (<https://www.youtube.com/playlist?list=PLxooeC3-xaNfS7jgZvVUM-kUcrUbgHTWJ>)

Python - MVC y MTV (o MVT):

- MVC y MTV son dos padrones de proyecto (desing patterns) utilizados para implementar interfaces y aplicaciones web.
- Comprender el patrón MVC
- Comprender el patrón MTV
- Comprender la diferencia entre los patrones MVC y MTV

Contenidos

- **Web** Glosario (<https://docs.djangoproject.com/es/4.2/glossary/>)

- **Artículo** Diferencia entre los patrones de diseño MVC y MVT (<https://barcelonageeks.com/diferencia-entre-los-patrones-de-diseno-mvc-y-mvt/>)
- **Artículo** Que es el patrón MTV (Model Template View) (<https://espifreelancer.com/mtv-django.html>)
- **Artículo** Django vs Laravel: ¿Cuál es el mejor framework en 2023? (<https://kinsta.com/es/blog/django-vs-laravel/>)
- **YouTube** Los modelos MVC y MVT o MVTU (https://www.youtube.com/watch?v=_RLCgEIYMH0)
- **YouTube** Proyecto Django Para Administrar Universidad | Aplicación Web con Base de Datos y Envío de Correos (https://www.youtube.com/watch?v=10_PpcLRaHk)

Python - Lambdas y Closures:

- Las funciones lambda son funciones anónimas. Mientras que las funciones tradicionales pueden ser creadas utilizando "def" como prefijo, las funciones lambda se crean utilizando "lambda".
- Un cierre (closure) en Python es un objeto de función interna, es decir, una función que se comporta como un objeto, que recuerda y tiene acceso a las variables en el ámbito local en el que fue creado, incluso después de que la función externa haya terminado de ejecutarse. También puede ser definido como un mecanismo para conectar datos a una función sin la necesidad de pasar parámetros.

Contenidos

- **Web** Closures (https://www.learnpython.org/es/Closures#google_vignette)
- **Artículo** Qué son y cómo utilizar expresiones Lambda en Python (<https://borjauria.es/que-son-y-como-utilizar-lambdas-en-python-4d1d168e2f90>)
- **Artículo** Expresiones Lambda en Python (<https://www.freecodecamp.org/espanol/news/expresiones-lambda-en-python/>)
- **Artículo** Funciones lambda de Python: funciones anónimas en Python (<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/funciones-lambda-de-python/>)
- **Artículo** Funciones lambda en Python. map(), filter() y reduce() (<https://j2logo.com/python/funciones-lambda-en-python/>)
- **Artículo** Que es un closure? (<https://our-academy.org/posts/closures>)
- **YouTube** Funciones Lambda o Anónimas con Python (<https://www.youtube.com/watch?v=BY0uu-ueisM>)

- **YouTube** Funciones Lambda en Python (<https://www.youtube.com/watch?v=1Ehus42VH5w>)
- **YouTube** ¿Cómo se usa la función lambda? (https://www.youtube.com/watch?v=WkJQI_crZUK)
- **YouTube** Pandas: Expresiones lambda en DataFrames (<https://www.youtube.com/watch?v=rAbp8CsDHNA>)
- **YouTube** Python Intermedio/Avanzado - Closures (<https://www.youtube.com/watch?v=dLhCJMygmAk>)

Nivel 3

Arquitectura de Microservicios:

- Los microservicios son un enfoque de arquitectura en el que el software consiste en pequeños servicios independientes que se comunican entre sí y se organizan de acuerdo con sus dominios de negocio.
- Aprender el concepto de arquitectura diseñada para microservicios
- Realizar la comunicación mediante API
- Mejorar la escalabilidad de un sistema

Contenidos

- **Web** Diseño de una aplicación orientada a microservicios (<https://learn.microsoft.com/es-es/dotnet/architecture/microservices/multi-container-microservice-net-applications/microservice-application-design>)
- **Artículo** Microservicios y arquitectura de microservicios (<https://www.intel.la/content/www/xl/es/cloud-computing/microservices.html>)
- **Artículo** Microservicios vs API: Entendiendo la diferencia (<https://kinsta.com/es/blog/microservicios-vs-api/>)
- **Artículo** Microservicios Ejemplo de Flujo (<https://medium.com/mycodebad/microservicios-ejemplo-de-flujo-f45720a9b278>)
- **Artículo** Microservicios Conceptos (<https://medium.com/mycodebad/microservicios-conceptos-55d19636873e>)
- **YouTube** Patrones fundamentales de la arquitectura microservicios (<https://www.youtube.com/watch?v=A7y23uU1NHk>)
- **YouTube** Un ejemplo de microservicios #CafeConRivas (<https://www.youtube.com/watch?v=qAcUGw7HhxM>)

Contenedores:

- Los contenedores son paquetes de software que contienen todos los elementos necesarios para ejecutarse en cualquier entorno. La gestión de contenedores es un área crucial en la computación en nube y DevOps, que implica el uso de tecnologías para automatizar el proceso de creación, implementación, escalado y monitoreo de contenedores. Los contenedores son unidades de software estandarizadas que permiten a los desarrolladores empaquetar todas las dependencias de una aplicación (código, bibliotecas, configuraciones, etc.) en un solo paquete. Esto permite que la aplicación se ejecute de forma consistente en cualquier entorno de infraestructura.
- La tecnología de contenedores, como ejemplifica Docker, proporciona un entorno coherente y portátil para el desarrollo, las pruebas y la implementación de aplicaciones, lo que es vital para el trabajo eficiente de la ingeniería de datos. Además, Kubernetes, un sistema de organización de contenedores, permite la gestión, automatización y escalabilidad de aplicaciones basadas en contenedores en entornos de producción. Dominar estos conceptos y tecnologías permite a los ingenieros de datos construir y mantener canalizaciones de datos eficientes y confiables.
- Kubernetes (también conocido como k8s o Kube) es una plataforma de orquestación de contenedores de código abierto que automatiza gran parte de los procesos manuales necesarios para implementar, gestionar y escalar aplicaciones en contenedores.
- Aislar el software para que funcione independientemente
- Implementación de software en clústeres
- Modularizar su sistema en paquetes más pequeños
- Conocer la plataforma Docker
- Conocer Kubernetes

Contenidos

- **Web** IBM - ¿Qué son los contenedores? (<https://www.ibm.com/mx-es/topics/containers>)
- **Web** Microsoft - ¿Qué es un contenedor? (<https://azure.microsoft.com/es-mx/resources/cloud-computing-dictionary/what-is-a-container>)
- **Artículo** Contenedores y la nube (<https://dev.to/roremdev/contenedores-y-la-nube-3je9>)

- **Artículo** Contenedores: cómo es el ciclo de vida de una aplicación en Kubernetes (<https://dev.to/campusmvp/contenedores-como-es-el-ciclo-de-vida-de-una-aplicacion-en-kubernetes-1ief>)
- **Artículo** ¿Qué diferencia hay entre Docker (Contenedores) y Máquinas virtuales (VMWare, VirtualBox...)? (<https://dev.to/campusmvp/qu-diferencia-hay-entre-docker-contenedores-y-mquinas-virtuales-vmware-virtualbox-4ji3>)
- **Artículo** El potencial de Kubernetes para las empresas (<https://jlcasal.medium.com/el-potencial-de-kubernetes-para-las-empresas-d36f9eece999>)
- **Artículo** ¿Qué es Docker y para que sirve? Explicación (https://dev.to/prox_sea/que-es-docker-y-para-que-sirve-explicacion-5h2n)
- **YouTube** ¿Que es un contenedor? (<https://www.youtube.com/watch?v=x5zGoICZLnU>)
- **YouTube** ¿Qué es Docker y los contenedores? (<https://www.youtube.com/watch?v=kkfZs0vJFyU>)
- **YouTube** Volúmenes y redes en Docker (https://www.youtube.com/watch?v=DIdeI70dFI&ab_channel=Programaci%C3%B3nnespa%C3%B1ol)
- **YouTube** ¿Qué es KUBERNETES? ✅ Relación con DOCKER y CONTENEDORES ▶️ (<https://www.youtube.com/watch?v=V86eTbswdQo>)

Contenidos Alura:

- **Artículo** Empezando con Docker (<https://www.aluracursos.com/blog/empezando-con-docker>)
- **Artículo** Creando volúmenes con Docker (<https://www.aluracursos.com/blog/creando-volumenes-con-docker>)

Python - Tipado Estático:

- Python es un lenguaje de tipado dinámico, lo que significa que no es necesario pensar en tipos de datos. Los lenguajes de tipado estático (como C o Java) realizan verificaciones de tipos durante la compilación. Puede parecer más seguro, ya que puede especificar inmediatamente el tipo de parámetro de cada función.
- Conocer el "type hinting"

Contenidos

- **Web** Soporte para type hints (<https://docs.python.org/es/3.10/library/typing.html>)
- **Artículo** ¿Qué son los Type hints en Python? Mejorar la calidad de tu código y hazlo más legible (<https://dev.to/rohaquinlop/que-son-los-type-hints-en-python-mejorar->

[la-calidad-de-tu-codigo-y-hazlo-mas-legible-5e99\)](#)

- **Artículo** Sugerencias de tipo o type hints (<https://www.pythonparatodo.com/?p=330>)
- **YouTube** Qué es un lenguaje compilado e interpretado y qué es el tipado estático dinámico fuerte y débil (<https://www.youtube.com/watch?v=M32Az-IRUQI>)
- **YouTube** Tipificación estática en Python 3.9 - PyConES 2020 (<https://www.youtube.com/watch?v=2vYOxbd2ioc>)
- **YouTube** Usos e integraciones del tipado estático opcional en Python (<https://www.youtube.com/watch?v=YOB14X7zL4>)
- **YouTube** PyConAr 2020 - Introducción a Type Hints (<https://www.youtube.com/watch?v=EUyp0EpVmEI>)

Python - Generadores:

- Los generadores permiten declarar una función que se comporta como un iterador, por ejemplo, se puede usar en un ciclo 'for'.
- Crear objetos iteradores
- Utilizar evaluación perezosa
- Realizar tareas simultáneas
- Uso de la palabra clave 'yield'

Contenidos

- **Web** Python Intermedio: Generadores (<https://python-intermedio.readthedocs.io/es/latest/generators.html>)
- **Web** Python Panama: Generadores (<https://pythonpanama.github.io/pythonpractico/6/>)
- **Artículo** Generadores en Python – La base del Streaming (<https://jarroba.com/generadores-en-python-la-base-del-streaming/>)
- **Artículo** Python: generadores y yield (<https://medium.com/flux-it-thoughts/python-generadores-y-yield-c7ef66d44a94>)
- **YouTube** Generators - 20 - Python Intermedio tutorial en español (https://www.youtube.com/watch?v=-_iTmR2gSyA)
- **YouTube** Yield Python - 3 ejemplos Generadores Produce sin Control (<https://www.youtube.com/watch?v=0ULwZf1Mc5E>)

Python - Asíncrono:

- En la programación asíncrona, las funciones no se ejecutan en orden. Con la asincronía, podemos interrumpir el código para obtener alguna otra información necesaria para continuar la ejecución. Esto significa que el código espera a otra parte del código y, mientras espera, ejecuta las demás partes.
- Aprender acerca de las corutinas
- Las corutinas son generalizaciones de subrutinas. Se utilizan para la multitarea cooperativa, donde un proceso cede el control de manera voluntaria, periódica o cuando está inactivo, para permitir que múltiples aplicaciones se ejecuten simultáneamente.
- Manejar la concurrencia
- Conocer el concepto de objetos esperables
- Crear tareas concurrentes
- Conocer la biblioteca 'asyncio'

Contenidos

- **Web** asyncio — E/S Asíncrona (<https://docs.python.org/es/3.8/library/asyncio.html>)
- **Web** Corrutinas y Tareas (<https://docs.python.org/es/3.8/library/asyncio-task.html>)
- **Web** Multitasking cooperativo con co-rutinas (<https://rico-schmidt.name/pymotw-3/asyncio/coroutines.html>)
- **Artículo** Una introducción a la asincronia en Python (<https://medium.com/@jmillandev/asincronia-en-python-9d3542a728f5>)
- **Artículo** Generadores y Corrutinas en Python: Una Exploración Detallada (<https://www.linkedin.com/pulse/generadores-y-corrutinas-en-python-una-exploración-palacio-gaviria/?originalSubdomain=es>)
- **Artículo** Asincronia En Python (<https://leanmind.es/es/blog/asincronia-en-python/>)
- **YouTube** Funciones asincronas con Python | asyncio (<https://www.youtube.com/watch?v=GVicp9m3s44>)
- **YouTube** Entendiendo asyncio sin usar asyncio, por Juan Pedro Fisanotti (https://www.youtube.com/watch?v=u_NDCBdHhzc)
- **YouTube** PyConAr 2021 - Concurrencia y paralelismo en Python: Multithreading vs Multiprocessing vs Async (<https://www.youtube.com/watch?v=u77Az26bFPA>)

Python - args & kwargs:

- Las variables mágicas `*args` y `**kwargs` se utilizan comúnmente en la definición de una función y sirven para pasar un número desconocido de argumentos a una función.
- Comprender la diferencia entre `*args` y `**kwargs`

Contenidos

- **Web** Args y Kwargs en Python (<https://ellibrodepython.com/args-kwargs-python>)
- **YouTube** ¿Qué es args y kwargs en python? (<https://www.youtube.com/watch?v=9iBRkmpav0Y>)
- **YouTube** ¿Qué es *ARGS y **KWARGS en Python? Funciones con ARGUMENTOS OPCIONALES |Curso Python desde CERO #11 (https://www.youtube.com/watch?v=_nAuo8JsAcM)

Python - Métodos especiales (dunder):

- Los métodos especiales, o métodos mágicos, en Python son métodos predefinidos en todos los objetos, con invocación automática en circunstancias especiales. Normalmente, no se llaman directamente por el usuario, pero pueden ser sobrecargados (sobrescritos y modificados). Sus nombres comienzan y terminan con guiones bajos dobles llamados "dunder" (una expresión derivada de double underscore).
- Comprender el concepto de métodos especiales (o mágicos)
- Conocer los principales métodos mágicos y cómo usarlos

Contenidos

- **Artículo** Clases: métodos mágicos y propiedades (<https://recursospython.com/guias-y-manuales/clases-metodos-magicos-y-propiedades/>)
- **Artículo** Python if **name** == **main** Explicado con ejemplos de código (<https://www.freecodecamp.org/espanol/news/python-if-name-main/>)
- **YouTube** POO en Python 02. Métodos especiales 🏆 💻 (<https://www.youtube.com/watch?v=goFjfPkIfcg>)
- **YouTube** Programación con Python 8 - Métodos mágicos (sobrecarga de operadores) (<https://www.youtube.com/watch?v=JDLzMNzvLis>)

Contenidos Alura:

- **Curso** Curso de String en Python: extrayendo información de una URL (<https://app.aluracursos.com/course/string-python-extrayendo-informacion-url>)
- **Curso** Curso de Python: avanzando en el lenguaje (<https://app.aluracursos.com/course/python-avanzando-lenguaje>)

Python - Metaprogramación:

- Metaprogramación es una técnica de programación en la que los programas tienen la capacidad de tratar a otros programas como sus datos. Esto significa que un programa puede estar diseñado para leer, generar, analizar o transformar otros programas, e incluso modificarse a sí mismo durante la ejecución.
- Escribir un programa que manipula otros programas
- Usar metaclasses

Contenidos

- **Web** Metaprogramación con Python 3 (<https://2013.es.pycon.org/media/metaprogramming-python3.pdf>)
- **Artículo** ¿Qué es la Metaprogramación en JavaScript? En español, por favor. (<https://www.freecodecamp.org/espanol/news/que-es-la-metaprogramacion-en-javascript-en-espanol-por-favor/>)
- **Artículo** Metaclasses en Python (<https://pythondiario.com/2018/06/metaclasses-en-python.html>)
- **YouTube** Metaprogramación en Python - Raúl Cumplido - Track Avanzado (<https://www.youtube.com/watch?v=LxUaNOq6Fbw>)
- **YouTube** Metaclasses propia - 4 - Python Avanzado tutorial en español (<https://www.youtube.com/watch?v=4rarlb1Mhz4>)

Python - Multiprocesamiento:

- En Python, el módulo de multiprocesamiento incluye una API muy simple e intuitiva para dividir el trabajo entre varios procesos.
- Ejecutar procesos en paralelo
- Conocer la clase Pool

Contenidos

- **Web** Piscinas de procesos (Process Pools) - Documentación Python (<https://docs.python.org/es/3/library/multiprocessing.html#module-multiprocessing.pool>)
- **Artículo** Comprender el multiprocessing y los subprocessos múltiples en Python (<https://hackernoon.com/es/entender-multiprocesamiento-y-multihilo-en-python>)
- **Artículo** Paralelismo y concurrencia en Python (<https://forum.huawei.com/enterprise/es/paralelismo-y-concurrencia-en-python/thread/667239418785841152-667212895836057600>)
- **Artículo** Azure Cognitive Services utilizando multiprocessing (python) (<https://medium.com/@alvarado22daniela/azure-cognitive-services-utilizando-multiprocesamiento-python-d4d9df92ca9c>)
- **YouTube** Multiprocessos - 11 - Python Avanzado tutorial en español (https://www.youtube.com/watch?v=ptiAjwygm_s)
- **YouTube** Python Multiprocessing (<https://www.youtube.com/watch?v=v9a3HkKp7jQ>)
- **YouTube** Multiprocessing Python en Español - Función Pool (<https://www.youtube.com/watch?v=LiTcHW0StzY>)

Reflection y atributos:

- Los objetos de Reflection (reflexión) se utilizan para obtener información del tipo en tiempo de ejecución. Las clases que dan acceso a los metadatos de un programa en ejecución se encuentran en el espacio de nombres System.Reflection.
- Escribir código que lee la información y metadatos de objetos en tiempo de ejecución
- Obtener nombres de clases en tiempo de ejecución y crear objetos de una clase

Contenidos

- **Web** Acceso a atributos mediante reflexión | Microsoft Learn (<https://learn.microsoft.com/es-es/dotnet/csharp/advanced-topics/reflection-and-attributes/accessing-attributes-by-using-reflection>)
- **Web** System.Reflection Espacio de nombres | Microsoft Learn (<https://learn.microsoft.com/es-es/dotnet/api/system.reflection?view=net-7.0>)
- **Web** Reflexión en .NET (<https://learn.microsoft.com/es-es/dotnet/framework/reflection-and-codedom/reflection>)
- **Web** Reflexión - Documentación Python (<https://docs.python.org/es/3.11/c-api/reflection.html>)

- **Web** Inmersión al modo interactivo - Python (<https://entrenamiento-python-basico.readthedocs.io/es/3.7/leccion2/interactivo.html#introspeccion-en-python>)
- **Artículo** ¿Qué es la Metaprogramación en JavaScript? En español, por favor. (<https://www.freecodecamp.org/espanol/news/que-es-la-metaprogramacion-en-javascript-en-espanol-por-favor/>)
- **Artículo** Reflection en Java (<https://jarroba.com/reflection-en-java/>)
- **YouTube** Ejemplo Java Reflection (<https://www.youtube.com/watch?v=3ZGYEDV9PIA>)
- **YouTube** Tutorial C# nivel Avanzado 46 - Reflection/Reflexión (<https://www.youtube.com/watch?v=9kAd4WI0hgQ>)

Habilidad Auxiliar: Infraestructura

Git y GitHub - Fundamentos:

- Git es un sistema de control de versiones distribuido gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta proyectos muy grandes, con rapidez y eficiencia.
- GitHub es un servicio de hosting para el desarrollo de software y el control de versiones mediante Git.
- Crear un repositorio
- Clonar un repositorio
- Comprometerse, empujar y tirar hacia y desde el repositorio
- Revertir un commit
- Crear de ramas y Pull requests
- Manejar fusiones y conflictos

Contenidos

- **Web** Git: Libro de Consulta (<https://git-scm.com/book/es/v2>)
- **Web** GitHub Documentación (<https://docs.github.com/es>)
- **Web** Github Pages Documentación (<https://docs.github.com/es/pages/getting-started-with-github-pages/about-github-pages>)
- **Web** W3Schools: Git Tutorial (<https://www.w3schools.com/git/default.asp?remote=github>)
- **Web** Git School - Visualizing Git (<https://git-school.github.io/visualizing-git/>)
- **Web** Dangit, Git!?! (<https://dangitgit.com/es>)

- **Artículo** Git and Github Quickstart Tutorial (<https://medium.com/@prashantramnyc/git-and-github-quickstart-tutorial-654a71594dca>)
- **YouTube** ¿Qué es Git y cómo funciona? (<https://www.youtube.com/watch?v=jGehuhFhtnE>)
- **YouTube** Git y Github | Guia Práctico de Git y Github Desde Cero (<https://www.youtube.com/watch?v=HiXLkL42tMU>)

Contenidos Alura:

- **Artículo** Git y Github: que son y primeros pasos (<https://www.aluracursos.com/blog/git-y-github-que-son-y-primeros-pasos>)
- **Artículo** Guía sobre cómo instalar Git en diferentes sistemas operativos (<https://www.aluracursos.com/blog/guia-sobre-como-instalar-git-en-diferentes-sistemas-operativos>)
- **Artículo** Iniciando un repositorio con Git (<https://www.aluracursos.com/blog/iniciando-repositorio-con-git>)
- **Artículo** Comenzando con Git: aprendiendo a versionar (<https://www.aluracursos.com/blog/comenzando-con-git>)
- **Artículo** Creando un repositorio remoto en GitHub (<https://www.aluracursos.com/blog/creando-repositorio-remoto-en-github>)
- **Artículo** Clonando un repositorio con Git y GitHub (<https://www.aluracursos.com/blog/clonando-un-repositorio-remoto>)
- **Artículo** Paso a Paso para activar tu proyecto en GitHub Pages. (<https://www.aluracursos.com/blog/github-pages>)
- **Artículo** Cómo escribir un README increíble en tu Github (<https://www.aluracursos.com/blog/como-escribir-un-readme-increible-en-tu-github>)
- **Artículo** Buenas practicas en git: evitando errores (<https://www.aluracursos.com/blog/como-evitar-errores-en-git>)
- **Artículo** GIT: Errores de comandos y directorios (<https://www.aluracursos.com/blog/git-errores-de-comandos-y-directorios>)
- **Artículo** GIT: errores de commits (<https://www.aluracursos.com/blog/git-errores-de-commits>)
- **Artículo** GIT: Errores de fusión (<https://www.aluracursos.com/blog/errores-de-fusion>)

- **Artículo** GIT: Errores con el remoto (<https://www.aluracursos.com/blog/errores-con-el-remoto>)
- **YouTube** Git y GitHub para Principiantes #AluraMás (https://www.youtube.com/watch?v=-LmFK6skG7s&ab_channel=AluraLatam)
- **YouTube** Git y GitHub: Herramientas Esenciales para el Control de Versiones y Colaboración en Desarrollo (<https://youtu.be/dw04N616Abw>)
- **YouTube** ¿Puedo subir mi proyecto a Github sin líneas de comando? - Git y Github para principiantes (https://www.youtube.com/watch?v=Yfm16Tlpcwk&ab_channel=AluraLatam)
- **YouTube** Git y GitHub: Herramientas Esenciales para el Control de Versiones y Colaboración en Desarrollo (<https://www.youtube.com/watch?v=dw04N616Abw>)
- **YouTube** ¿Puedo subir mi proyecto a Github sin líneas de comando? - Git y Github para principiantes (<https://www.youtube.com/watch?v=Yfm16Tlpcwk>)
- **Curso** Git y GitHub: repositorio, commit y versiones (<https://app.aluracursos.com/course/git-github-repositorio-commit-versiones>)
- **Curso** Git y Github: estrategias de ramificación, conflictos y Pull Requests (<https://www.aluracursos.com/curso-online-git-github-estrategias-ramificacion-conflictos-pull-requests>)

HTTP - Fundamentos:

- HTTP significa Protocolo de transferencia de hipertexto. La comunicación entre las computadoras cliente y los servidores web se realiza mediante el envío de solicitudes HTTP y la recepción de respuestas HTTP.
- Comprender la diferencia entre los verbos HTTP
- Probar solicitudes y verificar los códigos de estado en el navegador
- Aprendiendo a hacer una solicitud HTTP en la línea de comando con WGET
- Descargar una imagen con WGET
- Realización de una POST

Contenidos

- **Web** W3Schools: ¿Qué es HTTP? (https://www.w3schools.com/whatis/whatis_http.asp)
- **Web** MDN Web Docs: Una descripción general de HTTP (<https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>)

- **Web** MDN Web Docs: Métodos de solicitud HTTP (<https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>)
- **Web** MDN Web Docs: HTTP (<https://developer.mozilla.org/es/docs/Web/HTTP>)
- **Web** MDN Web Docs: Métodos de petición HTTP (<https://developer.mozilla.org/es/docs/Web/HTTP/Methods>)
- **Web** MDN Web Docs: Mensajes HTTP (<https://developer.mozilla.org/es/docs/Web/HTTP/Messages>)
- **Web** HTTP Cats (<https://http.cat/>)
- **Web** HTTP Dogs (<https://http.dog/>)
- **YouTube** Peticiones, Métodos Http y Códigos de estado. (<https://www.youtube.com/watch?v=gBK-Mfa0lw8>)
- **YouTube** SSL, TLS, HTTPS, HTTP - Explicado Fácilmente (https://www.youtube.com/watch?v=6HJAWFenYx8&ab_channel=ProfeSang)
- **YouTube** ★ Protocolo HTTP 🖨 Requests y Responses con: GET, POST, PUT, PATCH y DELETE | Desarrollo web 🌐 (<https://www.youtube.com/watch?v=l2MihYAj0lw>)
- **YouTube** REST y los Verbos de HTTP (<https://www.youtube.com/watch?v=OHBHeAPoZ8E>)

Contenidos Alura:

- **Artículo** HTTP: Desmitificando el protocolo Web (<https://www.aluracursos.com/blog/http-desmitificando-el-protocolo>)
- **Artículo** ¿Cual es la diferencia entre HTTP y HTTPS? (<https://www.aluracursos.com/blog/cual-es-la-diferencia-entre-http-y-https>)
- **Artículo** HTTP: Diferencias entre GET y POST (<https://www.aluracursos.com/blog/diferencias-entre-get-y-post>)
- **Artículo** Métodos de petición HTTP (<https://www.aluracursos.com/blog/metodos-de-peticion-http>)
- **Curso** HTTP: La base de internet (<https://app.aluracursos.com/course/http-base-internet>)

JSON:

- JSON significa Notación de objetos de JavaScript. Es un formato de texto para almacenar y transportar datos.
- Crear un objeto

- Transformar un objeto en una cadena
- Transformar una cadena en un objeto
- Manipular un objeto

Contenidos

- **Web** MDN Web Docs: JSON (https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/JSON)
- **Web** MDN Web Docs: Trabajando con JSON (<https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON>)
- **YouTube** ¿Que és JSON y cómo funciona? (https://www.youtube.com/watch?v=z8qk7T_2sWg&ab_channel=SoyDalto)
- **YouTube** ★ ¿Qué es JSON? ¿Cuál es su SINTAXIS? 🖥️ ¿Cómo crear un archivo JSON? | DESARROLLO WEB 🌐 (https://www.youtube.com/watch?v=RhxOTqFbl5Q&ab_channel=TodoCode)

Contenidos Alura:

- **Artículo** ¿Que es Json? (<https://www.aluracursos.com/blog/que-es-json>)
- **Artículo** ¿JSON y Objeto JavaScript son lo mismo? (<https://www.aluracursos.com/blog/json-y-objeto-javascript-son-lo-mismo>)
- **Artículo** Simulando una API REST con json-server (<https://www.aluracursos.com/blog/simulando-una-api-rest-con-json-server>)
- **Artículo** ¿Qué es JSON Web Token? (<https://www.aluracursos.com/blog/que-es-json-web-token>)
- **Curso** JS en la Web: CRUD con JavaScript asíncrono (<https://www.aluracursos.com/curso-online-js-web-crud-javascript-asincrono>)

Línea de Comando - Fundamentos:

- CLI es un programa de línea de comandos que acepta la entrada de texto para ejecutar funciones del sistema operativo.
- Conocer los comandos más importantes

Contenidos

- **Web** W3Schools: What is Command Line Interface (CLI)? (https://www.w3schools.com/whatis/whatis_cli.asp)

- **Web** Uso de argumentos de la línea de comandos para Terminal Windows
(<https://learn.microsoft.com/es-es/windows/terminal/command-line-arguments?tabs=windows>)
- **Artículo** Interfaz de línea de comandos o CLI
(<https://www.computerweekly.com/es/definicion/Interfaz-de-linea-de-comandos-o-CLI>)
- **Artículo** El Manual de Comandos de Linux
(<https://www.freecodecamp.org/espanol/news/comandos-de-linux/>)
- **YouTube** Consola vs Terminal vs Shell vs CLI 🖥️ ¿Qué es la terminal?
(https://www.youtube.com/watch?v=1YxHXBsVNGQ&ab_channel=ProgramadorX)
- **YouTube** Aprende la linea de comandos en un mac - bash scripting
(https://www.youtube.com/watch?v=vfwA3pUnVOg&ab_channel=Datademia)
- **YouTube** freeCodeCamp.org: Command Line Crash Course
(<https://www.youtube.com/watch?v=yz7nYlnXLfE>)
- **YouTube** Traversy Media: Command Line Crash Course For Beginners - Terminal Commands
(<https://www.youtube.com/watch?v=uwAqEzhyjtw>)
- **YouTube** Comandos Básicos e Intermedios CMD (<https://youtu.be/erKosEQaaFc>)
- **YouTube** Terminal MAC tutorial en Español - Cómo usar la terminal en MAC
(<https://www.youtube.com/watch?v=TmP3y7Z2kk4>)

Contenidos Alura:

- **Artículo** CMD: Sugerencias para trabajar en el prompt de Windows
(<https://www.aluracursos.com/blog/consejos-para-trabajar-en-el-indicador-de-windows>)
- **Artículo** Como usar el terminal integrado de Visual Studio Code
(<https://www.aluracursos.com/blog/como-usar-el-terminal-integrado-de-visual-studio-code>)
- **Curso** Linux 1: conociendo y utilizando la terminal
(<https://app.aluracursos.com/course/linux-1-conociendo-utilizando-terminal>)


Cloud - Fundamentos:

- La computación en nube, o cloud computing, es la distribución de servicios informáticos a través de Internet mediante un modelo de tarificación de pago por uso. Una nube se compone de varios recursos informatizados, desde los propios ordenadores (o instancias, en terminología de nube) hasta las redes, el almacenamiento, las bases de datos y todo lo que les rodea. En otras palabras, todo

lo que normalmente se necesita para montar el equivalente a una sala de servidores, o incluso un centro de datos completo, estará listo para usar, configurar y ejecutar.

- Conocer la diferencia entre IaaS, PaaS y SaaS
- Conocer los mayores proveedores de nube
- Especializarse en un proveedor específico de su preferencia

Contenidos

- **Web** ¿Qué es la informática en la nube? | Microsoft Azure (<https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-cloud-computing/>)
- **Web** Amazon AWS: ¿Qué es la computación en nube? (<https://aws.amazon.com/en/what-is-cloud-computing/>)
- **Web** Tipos de computación en la nube (<https://aws.amazon.com/es/types-of-cloud-computing/>)
- **Web** ¿Cómo funciona Azure? (<https://learn.microsoft.com/es-es/azure/cloud-adoption-framework/get-started/what-is-azure>)
- **Web** ¿Qué es el almacenamiento en la nube? (<https://aws.amazon.com/es/what-is/cloud-storage/>)
- **Web** ¿Qué es la seguridad en la nube? (<https://cloud.google.com/learn/what-is-cloud-security?hl=es-419>)
- **Web** Arquitectura sin servidor (<https://learn.microsoft.com/es-es/dotnet/architecture/serverless/serverless-architecture>)
- **Web** ¿Qué es Docker? (<https://aws.amazon.com/es/docker/>)
- **Artículo** Guía para principiantes sobre los fundamentos de la computación en nube (<https://scientya.com/a-beginners-guide-to-the-basics-of-what-cloud-computing-is-about-e8b3b7f25a30/>)
- **Artículo** Cloud Computing para principiantes (<https://medium.com/hackernoon/cloud-computing-for-beginners-85d168959afb/>)
- **Artículo** ¿Qué es Google Cloud y para qué sirve? (<https://www.incentro.com/es-ES/blog/que-es-google-cloud-platform>)
- **YouTube** ¿Qué es computación en la nube? | ¿Qué es cloud computing? | Explicado en 4 minutos (<https://youtu.be/MCKdaih2ISo>)
- **YouTube**  CLOUD COMPUTING ¿Qué es IaaS, PaaS y SaaS? | Modelos de Servicio Cloud (<https://youtu.be/VR8aXePkQ5M>)

- **YouTube** ¿Qué es AWS? (<https://www.youtube.com/watch?v=x2vrg7HuM6g>)
- **YouTube** ¿Cómo empiezo con Google Cloud? (Hablemos en Cloud) (<https://www.youtube.com/watch?v=OiDWqu0oQfo>)
- **YouTube** Introducción a la infraestructura de Google Cloud (<https://www.youtube.com/watch?v=209DGQCism4>)
- **YouTube** ¿Qué es la Computación en la Nube? | AWS desde cero - Parte 1: Introducción (<https://www.youtube.com/watch?v=IciVhWQ8npw>)

Contenidos Alura:

- **Artículo** ¿Qué es Cloud y sus principales servicios? (<https://www.aluracursos.com/blog/que-es-cloud-y-sus-principales-servicios>)
- **Artículo** Conociendo Terraform (<https://www.aluracursos.com/blog/conociendo-terraform>)
- **Artículo** Empezando con Docker (<https://www.aluracursos.com/blog/empezando-con-docker>)
- **Artículo** Heroku, Vercel y otras opciones de cloud como plataforma (<https://www.aluracursos.com/blog/heroku-vercel-y-otras-opciones-de-cloud-como-plataforma>)
- **YouTube** Fundamentos del OCI | Contenidos ONE (<https://youtu.be/rEgSc0UqX-g>)
- **Curso** Curso Oracle Cloud Infrastructure: implementación de una aplicación en la nube (<https://app.aluracursos.com/course/oracle-cloud-infrastructure-implementacion-aplicacion-nube>)
- **Curso** Curso Oracle Cloud Infrastructure: base de datos e infraestructura como código (<https://app.aluracursos.com/course/oracle-cloud-infrastructure-base-datos-infraestructura-codigo>)
- **Curso** Curso Deploy en Amazon EC2: Alta disponibilidad y escalabilidad de una aplicación (<https://app.aluracursos.com/course/deploy-amazon-ec2-alta-disponibilidad-escalabilidad>)
- **Curso** Curso Amazon Lightsail: Simplificando la nube (<https://app.aluracursos.com/course/amazon-lightsail-simplificando-nube>)

SQL - Fundamentos:

- Conocer los comandos más comunes de SQL
- Usar SELECT para consultar una tabla
- Usar INSERT para insertar datos en una tabla

- Usar UPDATE para actualizar una tabla
- Usar DELETE para eliminar datos de una tabla
- Usar JOIN para conectar los datos de múltiples tablas
- Conocer las cláusulas (FROM, ORDER BY, etc.)

Contenidos

- **Artículo** Amazon: ¿Qué es SQL? (https://aws.amazon.com/es/what-is/sql/?nc1=h_ls)

Contenidos Alura:

- **Artículo** MySQL: desde la descarga e instalación hasta su primera tabla (<https://www.aluracursos.com/blog/mysql-desde-la-descarga-e-instalacion-hasta-su-primera-tabla>)
- **Artículo** Bases de datos relacionales (<https://www.aluracursos.com/blog/base-de-datos-relacional>)
- **Artículo** ¿Qué es SQL? (<https://www.aluracursos.com/blog/que-es-sql>)
- **Artículo** Normalización en base de datos - Estructura (<https://www.aluracursos.com/blog/normalizacion-en-base-de-datos>)
- **Artículo** En SQL, null es null, vacío está vacío (<https://www.aluracursos.com/blog/en-sql-null-es-null-vacio-es-vacio>)
- **Artículo** SELECT, INSERT, UPDATE y DELETE en SQL: aprende a utilizar cada uno (<https://www.aluracursos.com/blog/select-insert-update-delete-sql>)
- **Artículo** Funciones de agregación con GROUP BY en SQL, ¿cómo utilizarlas? (<https://www.aluracursos.com/blog/funciones-de-agregacion-con-group-by-en-sql-como-utilizarlas>)
- **Artículo** SQL JOIN: Aprenda INNER, LEFT, RIGHT, FULL e CROSS (<https://www.aluracursos.com/blog/sql-join-aprenda-inner-left-right-full-e-cross>)
- **Artículo** select count(*), count(1) y count(nombre): batalla de los counts de SQL (<https://www.aluracursos.com/blog/select-count-count1-e-countnome-la-batalla-de-los-counts-de-sql>)
- **YouTube** Descomplicando Base de Datos | #Aluramás (https://www.youtube.com/watch?v=G1cDRqKuxpg&t=6s&ab_channel=AluraLatam)
- **YouTube** ¿Qué es SQL y NoSQL? (<https://www.youtube.com/watch?v=cLLKVd5CNlc>)
- **YouTube** Banco de Datos MySQL (<https://www.youtube.com/watch?v=8J0AoPZMVxA>)
- **Curso** SQL con MySQL (<https://app.aluracursos.com/formacion-sql-con-mysql>)

- **Curso** Curso Introducción a SQL con MySQL: Manipule y consulte datos (<https://app.aluracursos.com/course/introduccion-sql-mysql-manipule-consulte-datos>)
- **Curso** SQL Server: consultas avanzadas con Microsoft SQL Server 2019 (<https://app.aluracursos.com/course/sql-server-consultas-microsoft-sql-server-2019>)
- **Curso** Formación Modelado de datos (<https://www.aluracursos.com/formacion-modelado-de-datos>)
- **Curso** Formación SQL con Microsoft SQL Server (<https://www.aluracursos.com/formacion-SQL-con-Microsoft-SQL-Server-2019>)

Habilidad Auxiliar: Buenas prácticas y herramientas

SOLID:

- Solid tiene cinco principios considerados como buenas prácticas en el desarrollo de software que ayudan a los programadores a escribir los códigos más limpios, dividiendo las responsabilidades, disminuyendo los acoplamientos, facilitando la refactorización y estimulando el reaprovechamiento del código. Propuesto por Robert C. Martin, SOLID propicia el desarrollo de un código limpio, legible y comprobable.

Contenidos

- **Web** Los principios SOLID de programación orientada a objetos (<https://www.freecodecamp.org/espanol/news/los-principios-solid-explicados-en-espanol/>)
- **Web** Principios SOLID: Qué son, cuáles, y qué beneficios aporta usarlos (<https://devexperto.com/principios-solid/>)
- **Podcast** SOLID - los androides (<https://open.spotify.com/episode/47eElzhTPR1RluVcxeVal4?si=4df70e899c034283>)
- **YouTube** Los principios SOLID, ¡explicados! (<https://www.youtube.com/watch?v=2X50sKeBAcQ>)

Clean Architecture:

- Clean Architecture (Arquitectura Limpia) es una forma de desarrollar software, de tal forma que solo mirando el código fuente de un programa, debes ser capaz de decir lo que el programa hace.

Contenidos

- **Web** ¿Qué es Clean Architecture? (<https://clean-architecture-python.readthedocs.io/en/latest/introduccion/index.html>)
- **Artículo** ¿Por qué utilizo Clean Architecture? (<https://xurxodev.com/por-que-utilizo-clean-architecture-en-mis-proyectos/>)
- **YouTube** Revisando Clean code, vale la pena leerlo? | review clean code (<https://www.youtube.com/watch?v=uQfm6YaJTJI>)
- **YouTube** Hexagonal architecture, qué es y qué diferencias tiene contra Clean Architecture? - PT 1 (<https://www.youtube.com/watch?v=NOWU4K6piwo>)
- **YouTube** Desarrollo ágil con Arquitectura limpia Hexa3 (<https://medium.com/@dariopalminio/desarrollo-%C3%A1gil-de-ecosistemas-de-aplicaciones-hexagonales-3-capas-hexa3l-d6370bf11db0>)
- **Podcast** DevTalles - 125: Arquitectura Limpia (<https://open.spotify.com/episode/3ftlJfucj7Jx4ZO2cDSghx?si=e6b2c5607ead43b5>)

Contenidos Alura:

- **Artículo** Design Patterns: Breve introducción a los patrones de diseño (<https://www.aluracursos.com/blog/design-patterns-introduccion-a-los-patrones-de-diseno>)

Design Patterns:

- En ingeniería de software, un "patrón de diseño" (Design Pattern en inglés) es una solución general y reutilizable para un problema que ocurre normalmente dentro de un determinado contexto de diseño de software.
- Conocer y aplicar los principales patrones de diseño.

Contenidos

- **Web** Patrones de Diseño (<https://refactoring.guru/es/design-patterns>)
- **Web** Designer Patterns (<https://github.com/FernandoCalmet/design-patterns>)
- **Web** ¿Qué son los patrones de diseño de software? (<https://profile.es/blog/patrones-de-diseno-de-software/>)
- **Web** Design Patterns: conoce los diferentes tipos que existen y sus beneficios (<https://www.hostgator.mx/blog/design-patterns-que-debes-saber/>)
- **Podcast** Patrones Diseño, ventajas y desventajas (<https://open.spotify.com/episode/3VjQHnPVusU6zz5PyIMVFu?>)

[si=a613cb4c157e4055\)](#)

- **Podcast** Patrones Diseño
(<https://open.spotify.com/episode/6QO1HYdAgzrMGLVpz2kn0C?si=c95296f387c4490a>)
- **YouTube** ♦ Patrones de diseño software: Repaso completo en 10 minutos
(<https://www.youtube.com/watch?v=6BHOeDL8vls>)

Contenidos Alura:

- **Artículo** Design Patterns: Breve introducción a los patrones de diseño
(<https://www.aluracursos.com/blog/design-patterns-introduccion-a-los-patrones-de-diseno>)

Jupyter y Colab:

- Jupyter Notebook y Google Colaboratory son portátiles que permiten la creación de bloques de texto y bloques de código
- Los Notebooks facilitan la elaboración de proyectos de Data Science por ser posible visualizar el resultado de la ejecución luego del trecho de código
- Google Colaboratory le permite escribir y ejecutar códigos Python directamente en el navegador, sin ninguna o pocas configuraciones necesarias
- Facilitan el intercambio de proyectos entre el equipo

Contenidos

- **Web** Project Jupyter (<https://jupyter.org/>)
- **Web** Anaconda (<https://www.anaconda.com/>)
- **Web** Te damos la bienvenida a Colaboratory (<https://colab.research.google.com/>)
- **YouTube** Google Colab TUTORIAL para Principiantes
(https://www.youtube.com/watch?v=U_g5KRiJ5DA)
- **YouTube** Cómo usar JUPYTER NOTEBOOK 📝 qué es JUPYTERLAB [Curso Python Data Science Español] (<https://www.youtube.com/watch?v=CwbMaSkKDZg>)
- **YouTube** Cómo HACER un Análisis de Datos en Python con Jupyter Notebook 🐍
(<https://www.youtube.com/watch?v=Vku-9Us6Rpw>)

Contenidos Alura:

- **Artículo** Google Colab: ¿qué es y cómo usarlo?
(<https://www.aluracursos.com/blog/google-colab-que-es-y-como-usarlo>)

Extracción y tratamiento de datos:

- Obtener los datos que se analizarán
- Tratar los datos obtenidos, transformándolos, alterando su estructura y valores a fin de dejar la base de datos más coherente y garantizar que los datos que serán trabajados estén en las mejores condiciones para ser analizados

Contenidos

- **Web** ¿Qué es extracción de datos y cómo funciona?
(<https://www.klippa.com/es/blog/informativo/extraccion-datos/>)
- **Artículo** Extracción de datos con Destructuring en Javascript
(<https://medium.com/@artjulus/extracción-de-datos-con-destructuring-en-javascript-8dc46d05af4f>)
- **Artículo** Automatizar mis descargas de datos makes me happy 😊
(<https://medium.com/tacosdedatos/automatizar-mis-descargas-de-datos-makes-me-happy-62df02b6bcc0>)
- **Artículo** Herramientas para extracción de datos estructurados
(<https://medium.com/@marianmoldovan/herramientas-para-extracción-de-datos-estructurados-715168f060a4>)

Contenidos Alura:

- **Curso** Python Pandas: Tratamiento y análisis de datos
(<https://app.aluracursos.com/course/python-pandas-tratamiento-analisis-datos>)
- **Curso** Pandas: Formatos diferentes de entrada y salida (IO)
(<https://app.aluracursos.com/course/pandas-formatos-entrada-salida-io>)
- **Curso** Manipulando datos gigantes con Pandas
(<https://www.aluracursos.com/blog/manipulando-datos-gigantes-con-pandas>)