Trabajo Práctico Integrador

Gestión de Datos de Países en Python: filtros, ordenamientos y estadísticas

Alumnos - Grupo n° 84

Constanza Jazmín Jimenez (Comisión 14) y Thaís Cristina de Oliveira Alvim (Comisión 14).

Tecnicatura Universitaria en Programación - Universidad Tecnológica Nacional.

Programación

Docente Titular

Ariel Enferrel

Docente Tutor

Maximiliano Sar Fernandez

GitHub

Youtube

11 de noviembre de 2025

INTRODUCCIÓN

El presente trabajo práctico integrador tiene como objetivo aplicar los conocimientos adquiridos en la materia Programación 1, desarrollando una aplicación en Python para gestionar información sobre países.

El sistema permite leer datos desde un archivo CSV, agregar, modificar, buscar, filtrar y ordenar registros, como también calcular estadísticas básicas.

A lo largo del desarrollo se han implementado estructuras de datos (listas y diccionarios), funciones modulares, condicionales, bucles, validaciones y manejo de archivos.

OBJETIVOS

Desarrollar una aplicación en Python que permita gestionar información sobre países, aplicando listas, diccionarios, funciones, estructuras condicionales y repetitivas, ordenamientos y estadísticas. El sistema debe ser capaz de leer datos desde un archivo CSV, realizar consultas y generar indicadores clave a partir del dataset. El objetivo principal es afianzar el uso de estructuras de datos, modularización con funciones y técnicas de filtrado/ordenamiento, aplicando los conceptos aprendidos en Programación 1.

LISTAS

Según Downey (2015), una lista en Python es una colección ordenada de valores, en los que cada valor puede ser de cualquier tipo. Las listas son mutables, lo cual significa que pueden modificarse luego de su creación. Se definen mediante corchetes ([]) y sus elementos pueden ser de cualquier tipo, incluso otras listas o diccionarios. Esta flexibilidad las convierte en una de las estructuras más utilizadas en Python para la manipulación y gestión de datos. Cada elemento de la lista tiene un índice, que indica su posición dentro de la secuencia. El índice comienza con 0.

Downey (2015) señala que el acceso a elementos por índice es una operación muy eficiente, ya que Python utiliza referencias directas en memoria para localizar los datos. Python proporciona varios métodos y funciones nativas para trabajar con listas: append(x) para agregar un elemento al final de la lista, insert(i, x) para insertar un elemento en una posición específica, remove(x), para eliminar el primer elemento que coincida con el valor dado, pop(i), para eliminar y devolver un elemento en la posición i, len(lista), para devolver cantidad de elementos, sort() / sorted(), para ordenar los elementos e in, para verificar si un elemento pertenece a la lista.

En este proyecto, las listas se utilizan principalmente para guardar el conjunto de países leídos desde el archivo CSV, donde cada elemento es un diccionario con los datos de un país. Las listas permiten recorrer los datos, filtrarlos, ordenarlos y realizar operaciones estadísticas.

DICCIONARIOS

El diccionario de datos es un conjunto de metadatos; se define como una documentación escrita de forma estructurada y almacena una gama de informaciones esenciales como el nombre y la descripción de los elementos de datos, su tipo y tamaño y los valores permitidos. (Cruz *et al.*, 2024)

Como parte obligatoria en el desarrollo y mantenimiento de un sistema informático, los diccionarios se utilizan en lo más diversos tipos de aplicaciones. (Blaschek, 1987)

Los diccionarios de datos son esenciales porque almacenan información que puede ser consultada cuando sea necesario por el usuario o el administrador. (Santos *et al.*, 2013)

Downey (2015) resalta que los diccionarios son útiles cuando se necesita establecer una relación entre un identificador y un conjunto de datos.

En este proyecto, cada país se representa mediante un diccionario que contiene su nombre, población, superficie y continente, lo que permite un acceso rápido y organizado a cada campo.

FUNCIONES

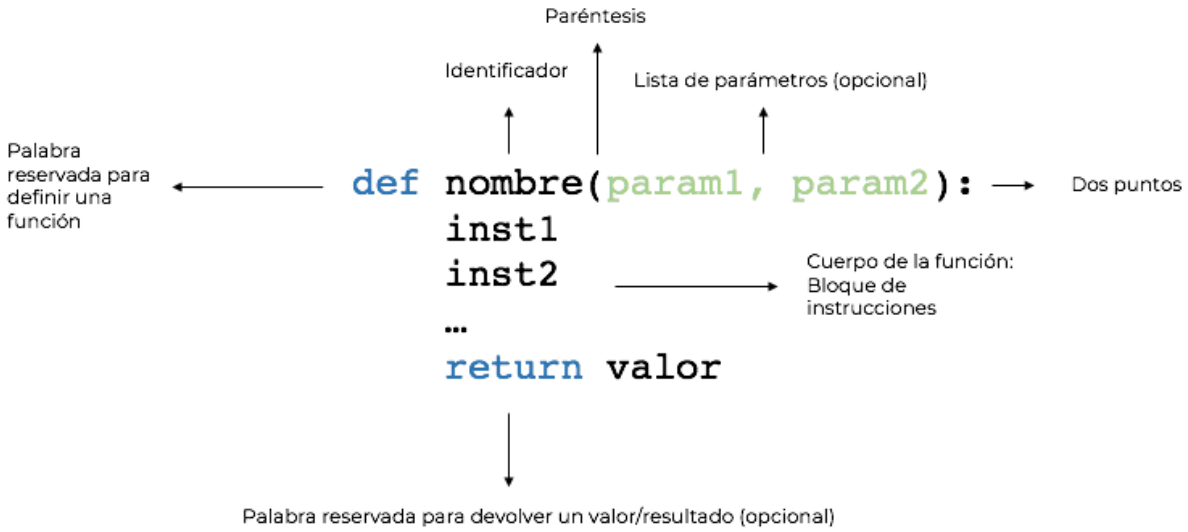
A medida que el código se vuelve más grande y complejo, con el aumento de líneas y partes repetidas que cumplen el mismo objetivo, se hace necesario el uso de funciones. (Gotardo, 2015)

Las funciones son bloques de código que pueden contener variables y constantes, se relacionan por medio de operadores y producen un valor una vez evaluadas. Desempeñan un papel fundamental en el área de la programación. Entre las ventajas del uso de funciones se pueden

citar la mejor organización del código, facilidad de mantenimiento y legibilidad, así como un código más eficiente y reutilizable. (Lopes y Garcia, 2022)

En el presente trabajo, se implementaron funciones para las principales operaciones, como ser leer el archivo CSV, agregar países, realizar búsquedas, aplicar filtros, ordenar datos y calcular estadísticas.

En Python tenemos a continuación la siguiente imagen con la ejemplificación de una función y sus partes.



Disponible en: https://aulavirtual.espol.edu.ec/courses/4558/pages/funciones-en-python

En Python, para crear una función se utiliza la palabra reservada *def*. El identificador es el nombre de la función, que debe ser único y descriptivo de su propósito, seguido de la entrada de valores (parámetros), que son opcionales. Luego tenemos el bloque de instrucciones o código, y al final la sentencia *return*, que devolverá un valor al concluir la ejecución.

ESTRUCTURAS CONDICIONALES

Los condicionales nos brindan la posibilidad de poder verificar si una condición propuesta resulta ser verdadera o falsa y en consecuencia de ello, lograr que un programa se ejecute. Son estructuras que permiten que un programa tome decisiones. Para ello, por ejemplo, se suele expresar de la siguiente forma: if x > 0: print('x es positivo'). A la expresión que viene luego del if, se la llama condición. Si dicha condición es verdadera, se ejecuta lo que viene luego del print, en caso contrario, no se ejecuta, ya que las condiciones se basan en expresiones booleanas, dando como resultado True o False (Downey, 2016).

Asimismo, dichos condicionales pueden ser de varios tipos, tales como, condicionales simples, dobles, múltiple, anidados y combinados. Los condicionales simples consisten en lo explicado en el párrafo anterior y sólo ejecutan un bloque de código sí la condición resulta True. Los condicionales dobles ejecutan un bloque, si la condición es verdadera y otro si la condición es falsa, agregando el 'else'. Luego, los condicionales múltiples consisten en evaluar varias condiciones distintas, una tras otra, utilizando luego del if, el 'elif'. Por último, los condicionales anidados permiten agregar condicionales dentro de otros condicionales, utilizándose cuando hay que evaluar decisiones más complejas (Downey, 2016).

En este proyecto se aplican para validar datos ingresados por el usuario, evitar errores de ejecución y definir el comportamiento del programa frente a distintas opciones del menú.

ORDENAMIENTOS

Los ordenamientos en Python implican utilizar operadores de comparación a los fines de establecer relaciones lógicas entre valores, como mayor que (>), menor que (<) o igual a (==) (Downey, 2015).

Downey (2015) indica que Python ofrece funciones nativas como sorted() y el método .sort() que facilitan el ordenamiento de listas utilizando funciones personalizadas mediante el parámetro key.

En este trabajo práctico integrador, el ordenamiento cumple una función esencial dentro de las operaciones del sistema. Permite organizar los países de acuerdo con distintos criterios solicitados por el usuario (nombre, población o superficie) y presentar los resultados de forma clara, ordenada y significativa.

Además, el hecho de poder cambiar el tipo de orden (ascendente o descendente) demuestra el dominio del uso de parámetros y funciones como argumentos, uno de los aprendizajes clave de la materia.

Downey (2015) explica que el ordenamiento no sólo es una técnica de organización, sino también una aplicación práctica de conceptos fundamentales de la programación: iteración, comparación y manipulación de estructura de datos. El autor señala que comprender cómo se ordenan los datos permite entender también la eficiencia algorítmica y la importancia de elegir el método más adecuado según el tipo y tamaño de la colección.

El ordenamiento de datos en Python representa un recurso esencial para la organización, comparación y visualización de información. En el marco de este proyecto, los ordenamientos

no sólo cumplen una función práctica, sino que también ilustran el poder expresivo y la simplicidad sintáctica que caracterizan al lenguaje de Python.

ESTADÍSTICAS BÁSICAS

Las estadísticas básicas permiten obtener información descriptiva a partir de un conjunto de datos. En este trabajo se calcularon indicadores como el país con mayor y menor población, los promedios de población y superficie y la cantidad de países por continente.

Según Downey (2015), el análisis de datos en Python se puede abordar mediante operaciones de agregación, que consisten en aplicar funciones como sum(), len(), max() o min() sobre colecciones de datos, como listas o diccionarios. Estas funciones devuelven valores que representan medidas globales del conjunto, lo cual es esencial para generar indicadores numéricos.

En resumen, las estadísticas básicas permiten extraer información cuantitativa y comparativa de los datos, transformando un conjunto de registros en indicadores interpretables.

Esto favorece la toma de decisiones y el análisis exploratorio, principios que, según Downey (2015), son fundamentales en la programación orientada al procesamiento de datos y la resolución de problemas reales.

ARCHIVO CSV

La nomenclatura CSV (*Comma Separeted Values*) de una forma bien sencilla podemos definir el archivo CSV como valores separados por coma, en la cual almacena los datos en forma de tabla, donde cada columna es un registro y las líneas son los valores. (Okada, 2023)

La base de datos utilizada para la generación del archivo CSV corresponde al conjunto de datos proporcionado por *https://www.geonames.org*, el cual fue empleado para realizar los análisis y generar los resultados presentados en este trabajo.

Fueron extraídos los datos de 12 países al entorno de desarrollo Visual Studio Code, y posteriormente se procesaron para obtener únicamente el nombre, el área, la población y el continente.

Figura 2. E*jemplo de los datos del archivo CSV utilizado en el sistema*

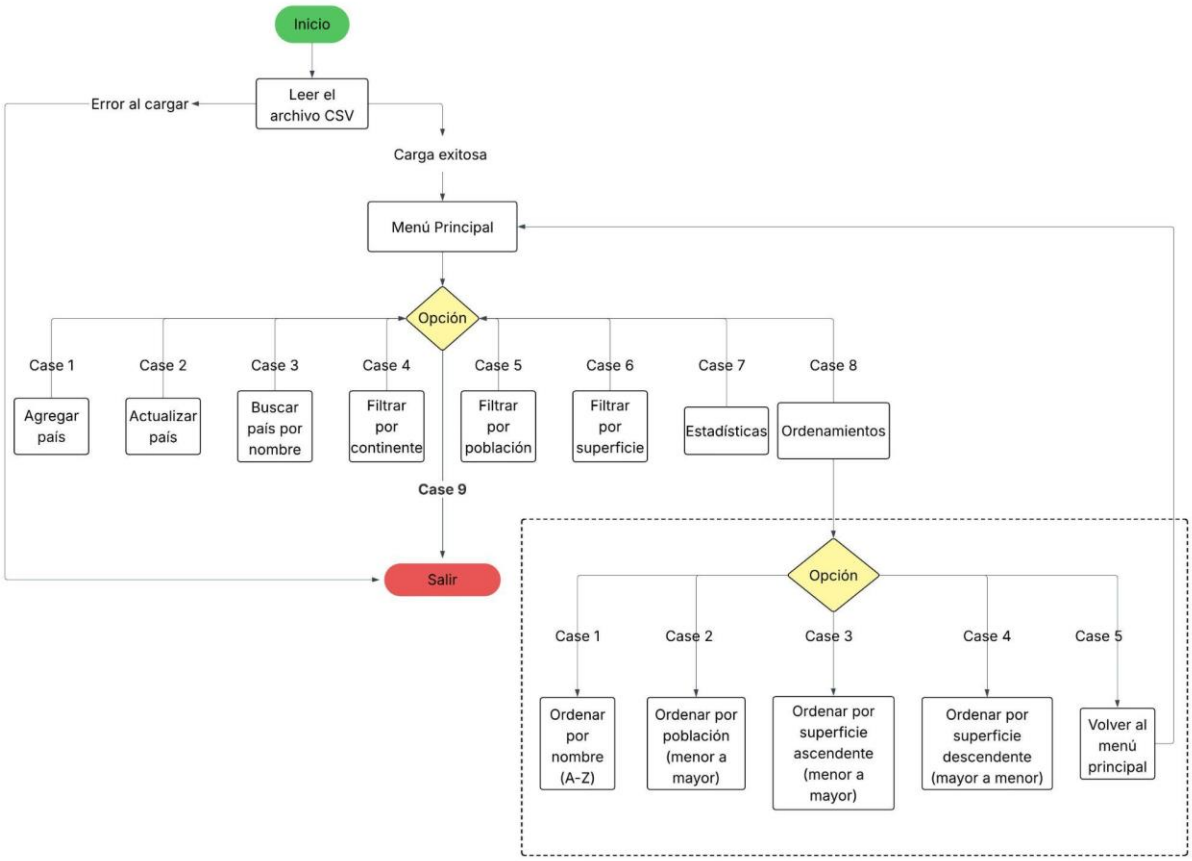
```
AD"></a>AD</td><td>AND</td><td>020</td><td>AN</td><td><a
href="https://www.geonames.org/countries/AD/andorra.html">Andorra</a></td><td>A
ndorra la Vella</td><td class="rightalign">468.0</td><td
class="rightalign">77,006</td><td>EU</td></tr>
```

Fuente: Elaboración propia (2025).

FLUJO DE OPERACIONES PRINCIPALES

El diagrama abajo representa la lógica principal del nuestro sistema desarrollado en Python. Al iniciar, son cargados los datos del archivo CSV que contiene la información de los países. Si la carga es exitosa, se presenta el menú principal con las distintas opciones disponibles, que permiten agregar, actualizar, buscar, filtrar, obtener estadísticas y ordenar la información. Cada opción ejecuta una función específica definida en los módulos del sistema y luego retorna al menú principal, hasta que el usuario selecciona la opción de salida.

Figura 3. *Diagrama de flujo general del sistema*



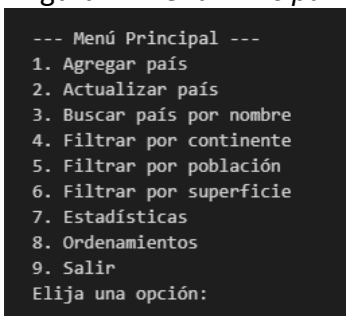
Fuente: Elaboración propia (2025).

CAPTURAS DE PANTALLAS

En las figuras 4, 5, 6 y 7 se pueden observar capturas de pantalla que muestran el funcionamiento principal del programa.

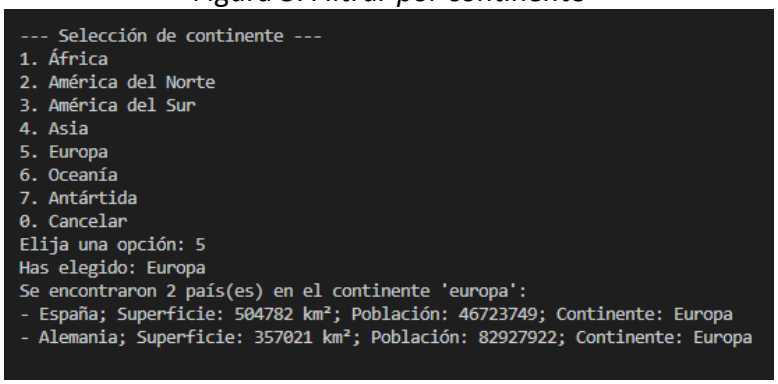
A continuación, se presenta el menú principal en ejecución, con las distintas opciones disponibles.

Figura 4. *Menú Principal*



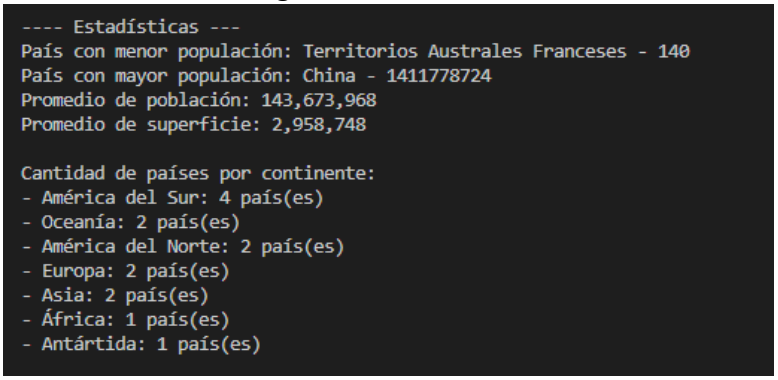
Resultado de filtrar los países, para evitar errores de escritura del continente, el usuario puede elegir el continente que desea filtrar.

Figura 5. *Filtrar por continente*



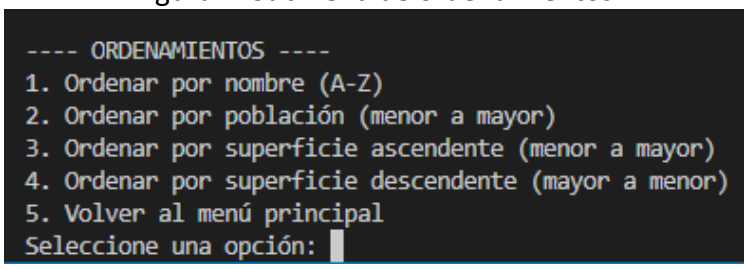
Para los datos de estadísticas tenemos los países con menor y mayor población, el promedio de población y superficie, y la cantidad de países por continente.

Figura 6. *Estadísticas*



Y, por último, el menú correspondiente a la opción de ordenamientos, que permite ordenar por nombre (A-Z), por población, por superficie ascendente (de menor a mayor) y por superficie descendente (de mayor a menor).

Figura 7. *Submenú de ordenamientos*



CONCLUSIÓN

El desarrollo del trabajo permitió consolidar los conocimientos sobre estructuras de datos, modularización y control de flujo en Python, aplicando los principios de programación estructurada descritos por Downey (2015).

El uso de listas y diccionarios resultó clave para organizar los datos, mientras que las funciones garantizaron una estructura de código clara y reutilizable.

Asimismo, el manejo de archivos CSV permitió trabajar con información externa y las estadísticas básicas facilitaron la obtención de indicadores relevantes.

El proyecto demuestra la versatilidad de Python para resolver problemas reales de gestión de datos.

El proyecto cumplió con el objetivo principal propuesto: el desarrollo de un programa para gestionar los países a partir de un archivo CSV. Durante el desarrollo surgieron desafíos relacionados con la lectura del CSV, el manejo de los acentos en los nombres de los continentes y la validación de los datos. Sin embargo, se realizaron pruebas para corregir los errores detectados y se implementaron funciones adicionales con el fin de obtener un programa completamente funcional.

Referencias Bibliográficas

Bazzotti, C., & Garcia, E. (2006). A importância do Sistema de Informação Gerencial na gestão empresarial para tomada de decisão. *Ciências Sociais Aplicadas em Revista, 6*(11). https://saber.unioeste.br/index.php/csaemrevista/article/view/368/279

Blaschek, J. R. S. (1987). Dicionário de dados para análise. En *1º Simpósio Brasileiro de Engenharia de Software*.

Cruz, A., et al. (2024). Dicionário de dados: Um olhar de qualidade sobre os dados de recursos humanos da Universidade Federal da Integração Latino-Americana. *AtoZ: Novas Práticas em Informação e Conhecimento, 13*(Número especial).

Downey, A. B. (2016). *Pensar en Python: Aprende a programar como lo hacen los científicos informáticos* (2.ª ed., J. Espinoza, Trad.). Green Tea Press. https://github.com/espinoza/ThinkPython2-spanish/blob/master/book/thinkpython2-spanish.pdf

Gotardo, Reginaldo Aparecido. Linguagens De Programação 1. ed. Rio de Janeiro: Seses, 2015. JAVA

Lopes, A., Garcia, G., Introdução à Programação: 500 Algoritmos Resolvidos, Rio de Janeiro: Editora Campus/Elsevier, 2002

Okada, R. S., & Bezerra, W. dos R. (2023, agosto 29–31). *Manipulador de arquivos CSV na linguagem Java, MongoDB e Docker*. Anais da XXIV FETEC – Feira do Conhecimento Tecnológico e Científico, Instituto Federal Catarinense, Campus Rio do Sul. Instituto Federal Catarinense.

Python Software Foundation. (2024). Python 3.12 Documentation: Sorting HOWTO. https://docs.python.org/3/howto/sorting.html

Santos, P. L. V. A. da C., & Santana, R. C. G. (2013). Dado e granularidade na perspectiva da informação e tecnologia: Uma interpretação pela Ciência da Informação. *Ciência da Informação, 42*(2), 199–209.