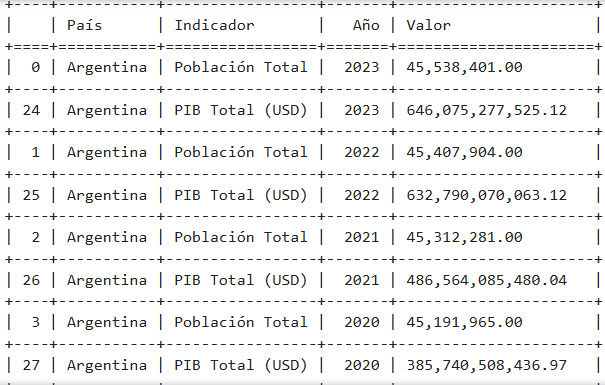
**PASO A PASO EJERCICIO 1 (PARTE A)**

Imagen 1

\*En la imagen se reflejan solo las primeras filas.



**1- IMPORTAMOS LAS BIBLIOTECAS NECESARIAS PARA EL TRABAJO:**

request: Permite hacer solicitudes HTTP a servicios web, en este caso, al Banco Mundial.  
pandas: Facilita la manipulación y análisis de datos.  
tabulate: Da formato a los datos para imprimirlos de manera más ordenada en la consola.

**2-DEFINIMOS LOS INDICADORES Y PAÍSES**

countries: contiene los códigos de los países que se van a consultar. En este caso “ARG” para Argentina y “BRA” para Brasil.

Indicators: es un diccionario donde reflejamos los indicadores con los que vamos a trabajar con los códigos de indicadores del Banco Mundial. ("SP.POP.TOTL": "Población Total", "NY.GDP.MKTP.CD": "PIB Total (USD)")

**3- CREAMOS UNA LISTA VACÍA PARA ALMACENAR LOS DATOS OBTENIDOS**

En este paso creamos una lista vacía “data" donde se guardarán los datos extraídos de la API.

**4- RECORRER CADA PAÍS E INDICADOR**

Usamos un doble bucle for: El primer for recorre los países en countries (ARG y BRA) y en el segundo for recorre los indicadores y sus nombres en indicators.

## **5-CONSTRUIR LA URL Y HACER LA SOLICITUD HTTP**

Primero se construye la URL con la estructura de la API del Banco Mundial. Por otro lado, la URL contiene el código del indicador, código del país, el código format=json para recibir la respuesta en formato JSON y los el códigos per\_page=1000 para obtener hasta **1000 registros por página**,  
page={page} para manejar la paginación y requests.get(url) que realiza la solicitud HTTP para obtener los datos.

**6- VERIFICAR SI LA SOLICITUD FUE EXITOSA**

En este paso se verifica con el código response.status\_code == 200 si la solicitud fue exitosa.

**7- VERIFICAR SI HAY DATOS DISPONIBLES**

La respuesta de la API es una lista con dos elementos:

\*json\_data[0]: Información general de la consulta.

\*json\_data[1]: Lista de registros con los datos.

\*Se verifica que json\_data[1] exista y tenga datos.

**8- EXTRAEMOS LOS DATOS CON LOS QUE VAMOS A TRABAJAR**

En este paso se recorre la lista de datos (json\_data[1]) y se filtran los registros:

\*Se excluyen valores None (record["value"] is not None).

\*Solo se toman datos desde el año 2000 en adelante.

\*Se agregan los datos a la lista data.

**9-MANEJO DE ERRORES**

En este paso proponemos que;

\*Si no hay datos (json\_data[1] is None), se imprime un mensaje y se sale del bucle.

\*Si la solicitud a la API falla (status\_code != 200), se imprime un error y se interrumpe el proceso

**10- SE CONVIERTEN Y ORDENAN LOS DATOS EN UN DATAFRAME**

Una vez obtenidos los datos:

\*Se convierten en un DataFrame de pandas.

\*Se ordenan por país y año.

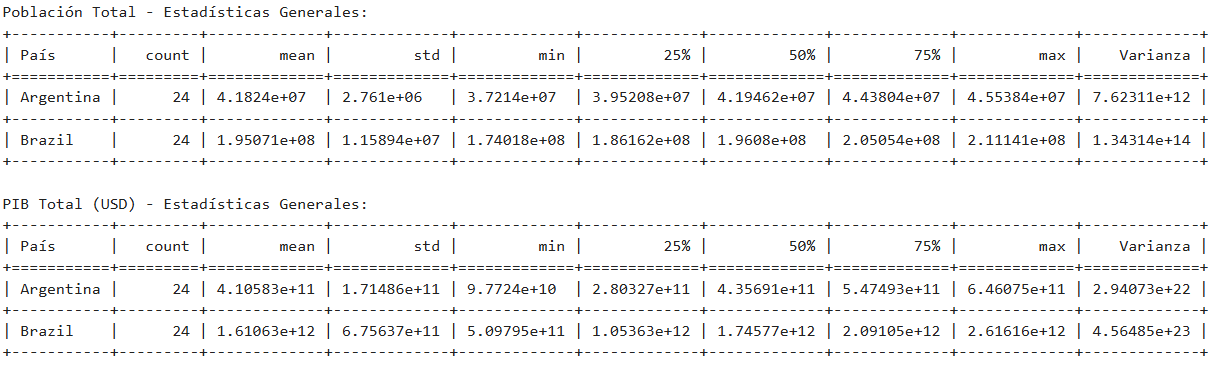
\*Se imprimen los resultados si hay datos disponibles.

\*Si no hay datos, se muestra un mensaje indicando esto mismo.

* Si el código se ejecuta correctamente, la salida muestra una tabla con la estructura de la “imagen 1”. Donde se muestra información para Argentina y Brasil con los datos de la población total y PBI total en usd de forma ordenada.

**PASO A PASO EJERCICIO 2 (PARTE A)**

\*Imagen 2



**1- IMPRIMIR ENCABEZADO DEL ANÁLISIS**

A raíz de lo realizado en el ejercicio anterior, imprimimos el título para indicar que comienza el análisis descriptivo (print("\nAnálisis Estadístico Descriptivo:"))

**2-CREAR UNA LISTA VACÍA PARA ALMACENAR COMPARACIONES**

Se inicializa una lista vacía comparativo, donde se almacenarán las métricas estadísticas para su comparación entre países.

**3- RECORREREMOS CADA INDICADOR DE MANERA ÚNICA**

En este paso utilizamos el código df["Indicador"].unique() que devuelve una lista con los nombres únicos de los indicadores en el DataFrame.

Esto se itera sobre cada indicador para calcular estadísticas de manera separada.

**4- Filtrar el DataFrame para cada indicador**

Para filtrar el DataFrame para cada indicador utilizamos el código subset = df[df["Indicador"] == indicator\_name] que crea un subconjunto subset que contiene sólo los datos del indicador actual.

Esto permite realizar cálculos estadísticos específicos para cada indicador.

**5- Calcular estadísticas descriptivas**

Utilizamos stats = subset.groupby("País")["Valor"].describe() para agrupa los datos por país (groupby("País")) y calcular estadísticas descriptivas sobre la columna Valor.

describe() devuelve:

\*count: Número de datos

\*mean: Media

\*std: Desviación estándar

\*min: Valor mínimo

\*25%, 50% (mediana), 75%: Cuartiles

\*max: Valor máximo

Para calcular la varianza utilizamos stats["Varianza"] = subset.groupby("País")["Valor"].var() donde se calcula la varianza de los valores por país y se agrega como una nueva columna en stats.

**6- Mostrar los resultados en una tabla**

En este paso imprimimos la tabla stats utilizando tabulate() con formato “grid” para una presentación prolija.

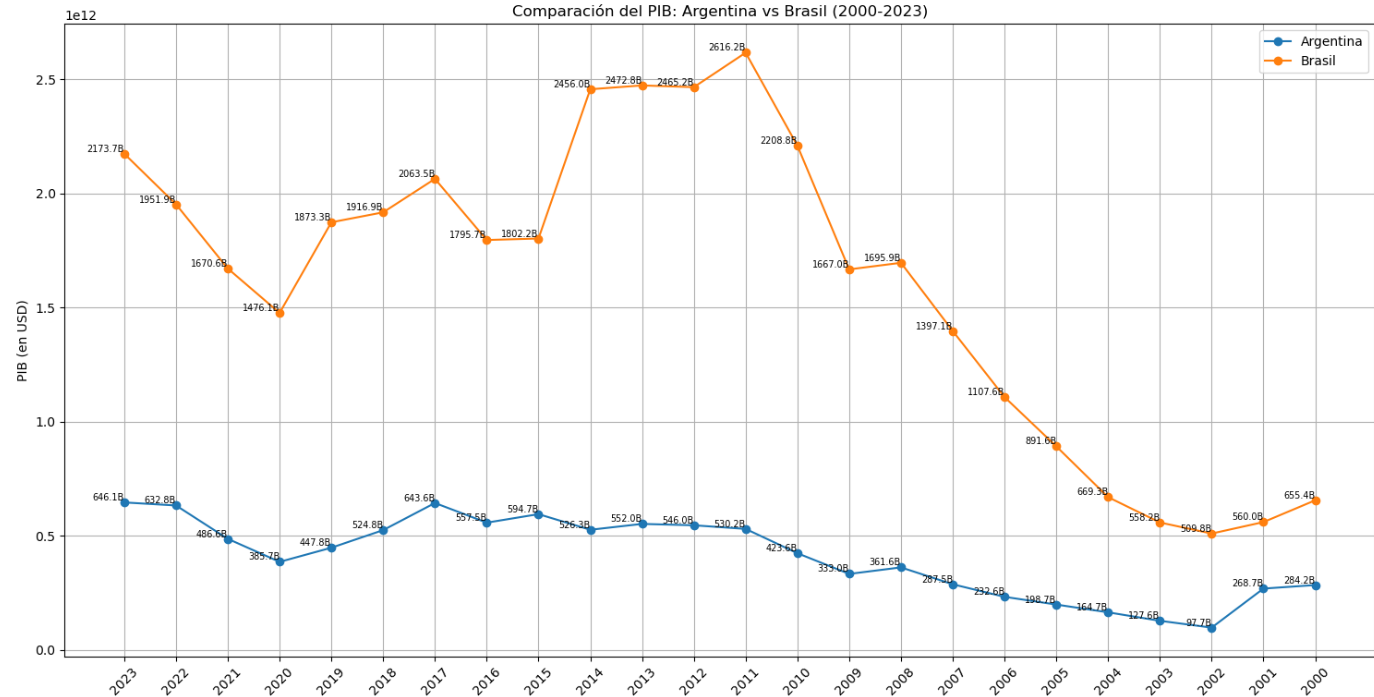
**7- Convertir la lista comparativa en un DataFrame**

Con comparativo\_df = pd.DataFrame(comparativo, columns=["País", "Indicador", "Media", "Desviación Estándar", "Varianza"]) convertimos la lista comparativo en un dataframe con columnas

**8- Imprimir la tabla comparativa**

Imprimimos el título "Comparación de Indicadores Entre Países" y se muestra la tabla comparativo\_df formateada con tabulate().

**PASO A PASO EJERCICIO 3a (PARTE A)**

****

**\***Imagen 3 (en el gráfico los valores del PBI se muestran en Billiones en escala Americana)

**1- IMPORTAMOS LAS BIBLIOTECAS NECESARIAS PARA EL TRABAJO:**

request: Permite hacer solicitudes HTTP a servicios web, en este caso, al Banco Mundial.  
pandas: Facilita la manipulación y análisis de datos

matplotlib.pyplot: Para graficar

**2- FUNCIÓN OBTENER PIB(PAÍS, INICIO, FIN):**

Llamamos a la API del Banco Mundial y extraemos PIB por año.

**3- FUNCIÓN ABREVIAR\_NUMERO(VALOR):**

En este paso se convierten números grandes a formato corto (B = Billón, M = Millón, K = Mil).

**4- DEFINIMOS PARÁMETROS**

Se crea un diccionario de países con sus códigos según el Banco Mundial y se define el rango de años para obtener los datos (2000-2023).

\*Países: Argentina (ARG), Brasil (BRA)  
\*Rango de años: 2000-2023

**5- OBTENEMOS LOS DATOS Y LOS GUARDAMOS EN UN DATAFRAME**:

Se crea un DataFrame vacío que itera sobre cada país en el diccionario (obtiene los datos del PIB con obtener\_pib() y agrega los valores al DataFrame bajo el nombre del país.) Asigna los años como índice.

**6- SE GENERA LA GRÁFICA CON MATPLOTLIB:**

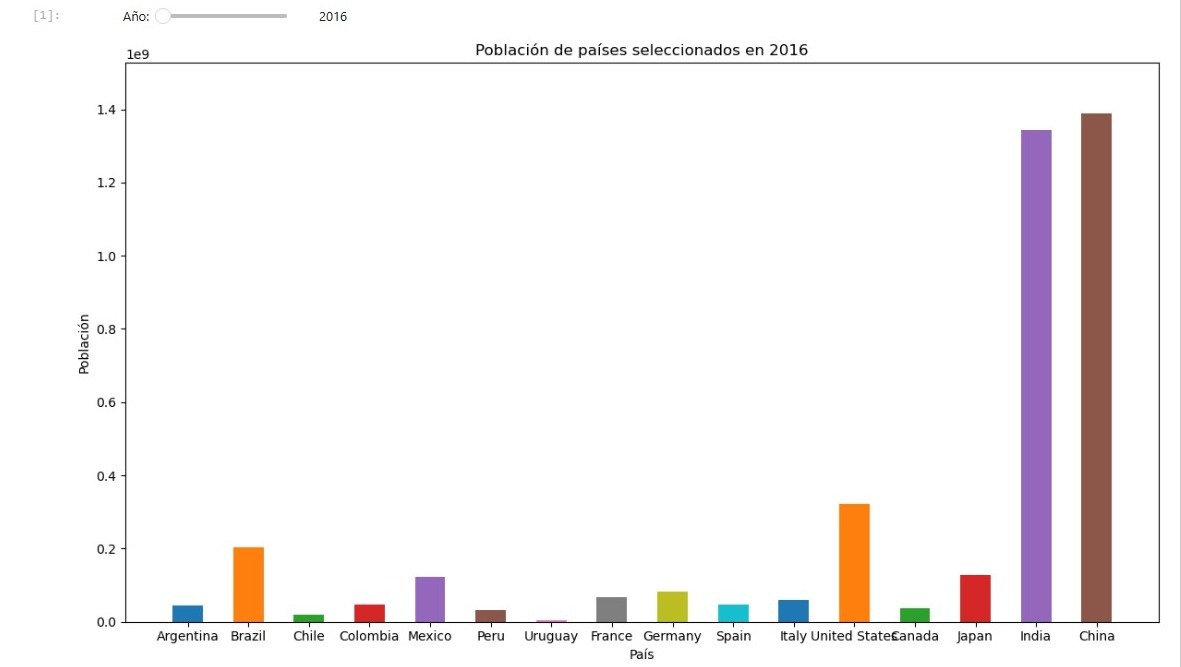
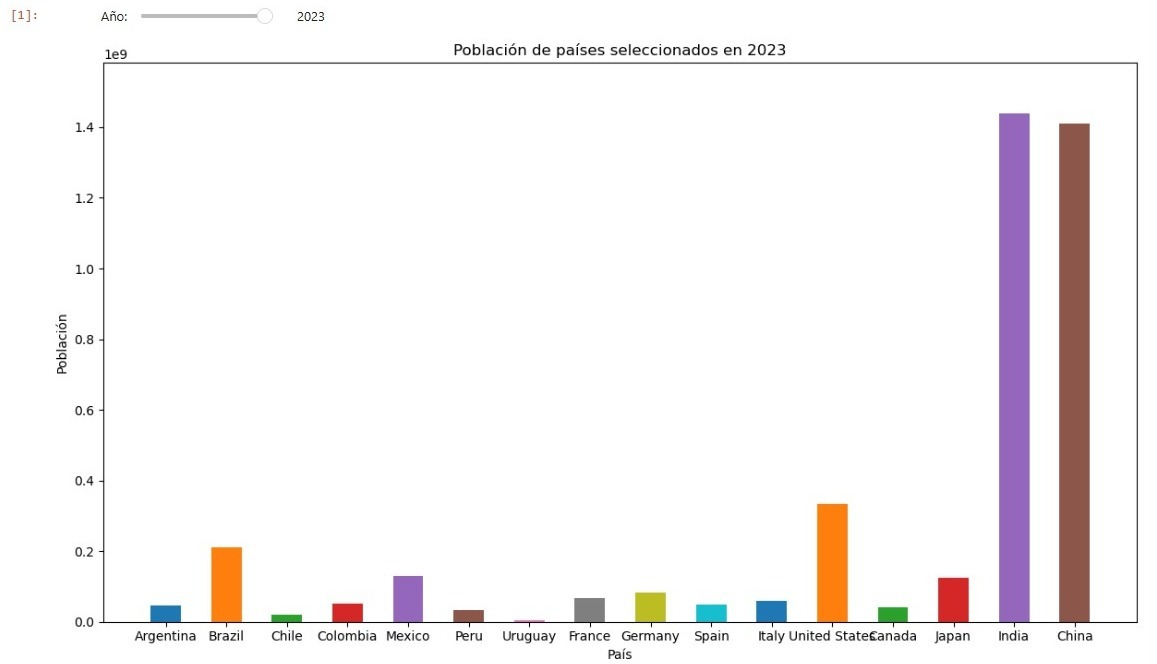
Se gráfica líneas con marcadores para cada país y añade etiquetas con valores del PIB en formato abreviado.

**7- SE PERSONALIZA Y MUESTRA LA GRÁFICA:**

Se añade Títulos, etiquetas, leyenda y ajuste de formato.

* Si el código se ejecuta correctamente podremos ver la comparación visual clara del PIB de Argentina vs Brasil (2000-2023), como en la \*imagen 3 anteriormente adjunta.

**PASO A PASO EJERCICIO 3b (PARTE A)**

****

\*imagen4 \*imagen5

**1- IMPORTAMOS LAS BIBLIOTECAS NECESARIAS PARA EL TRABAJO:**

request: Permite hacer solicitudes HTTP a servicios web, en este caso, al Banco Mundial.  
pandas: Facilita la manipulación y análisis de datos

matplotlib.pyplot: Para graficar

ipywidgets: Para agregar interactividad (selector de año).

2- **CLASS POPULATIONDATA**

Utilizamos el constructor \_\_init\_\_ para recibir lista de países y años. Y por otro lado el método get\_population\_data() que obtiene los datos de la API y crea un DataFrame y filtra valores nulos.

**3- CLASS POPULATIONPLOTTER**

Utilizamos el constructor \_\_init\_\_ para recibir el DataFrame de población. Y por otro lado el Método plot\_population(year)que filtra los datos por año y crea un gráfico de barras con matplotlib.

**4- EJECUTAMOS CÓDIGO**

En este paso ejecutamos código que nos permite crear instancias de las clases (PopulationData y PopulationPlotter) y generar un slider interactivo para cambiar el año y actualizar la gráfica.

* Si el código se ejecuta correctamente podremos ver un gráfico dinámico, como las imagen4 e imagen5 adjuntas anteriormente, que muestra la población de varios países por año.

Este gráfico dinámico nos permite cambiar el año con un slider, facilitando el análisis sin generar múltiples gráficos y permitiendo una visualización clara que nos facilita el análisis de tendencias del indicador con el que estamos trabajando.