UNIVERSIDAD TECNOLÓGICA NACIONAL

Documentación adicional
TP Integrador II - Matematica

Link a Youtube: https://youtu.be/HT44leDxbro

Link al repositorio:
https://github.com/tomasAnch/UTN-Integrador-Matematica-N-2

Integrantes

- Anchorena Tomás
- Angelelli Rodrigo
- Schneider Astrid

Tareas Realizadas

- **Schneider Astrid:** Se encargó de las operaciones entre los conjuntos A y B, generados a partir de los dígitos únicos de los DNIs. Implementó funciones para calcular la unión, intersección, diferencia y diferencia simétrica, utilizando los métodos propios de los conjuntos en Python. También fue responsable de la visualización de resultados y su validación mediante diagramas de Venn.

- **Angelelli Rodrigo:** Trabajó sobre las operaciones entre los conjuntos B y C. Además, implementó las expresiones lógicas redactadas en el informe:
  - Verificación de que los conjuntos A, B y C tengan al menos cinco elementos, lo que implica una alta diversidad numérica.
  - Búsqueda de dígitos comunes presentes en los tres conjuntos.

  Estas condiciones fueron traducidas al código usando funciones como len() e intersection(), permitiendo verificar de manera lógica y precisa las propiedades entre los conjuntos.

- **Anchorena Tomás**: Desarrolló la segunda parte del programa, correspondiente a los años de nacimiento. En esta sección:
  - Se contó cuántos años ingresados eran pares e impares.
  - Se verificó si alguno de los años era bisiesto.
  - Se generó un producto cartesiano entre los años de nacimiento y las edades actuales de los integrantes.

  Estas operaciones se realizaron mediante ciclos, listas, condicionales y funciones auxiliares, aplicando conceptos fundamentales de programación como la divisibilidad, iteraciones y relaciones entre conjuntos

Relación entre las expresiones lógicas y el código:

En este trabajo, se propusieron expresiones lógicas que fueron luego implementadas en código Python para verificar ciertas condiciones sobre los datos ingresados.

La primera expresión lógica fue: **“Si todos los conjuntos tienen al menos 5 elementos, entonces se considera que hay una alta diversidad numérica.”**
Esta condición busca evaluar la variedad de dígitos en los DNIs ingresados. En Python, esto se implementó mediante la función `len()`, que permite conocer la cantidad de elementos únicos en cada conjunto.
El código evalúa si los tres conjuntos generados a partir de los DNIs (`A`, `B` y `C`) tienen como mínimo 5 elementos cada uno:

```
if len(A) >= 5 and len(B) >= 5 and len(C) >= 5:

    print("Hay alta diversidad numérica.")
```

Si se cumple, el programa considera que hay "alta diversidad numérica", es decir, que hay una buena cantidad de dígitos distintos en cada DNI.

La segunda expresión fue: **“Si algún dígito aparece en todos los conjuntos, se marca como dígito común.”**
Esta condición se enfoca en encontrar elementos compartidos por los tres conjuntos, es decir, dígitos que aparecen en los tres DNIs. Para resolverlo, se usó la función `intersection()`, que permite obtener los elementos comunes entre conjuntos:

```
digitos_comunes = A.intersection(B).intersection(C)

if digitos_comunes:

        print(f"Dígitos comunes en los tres conjuntos: {digitos_comunes}")
```

Si la intersección no está vacía, significa que hay al menos un dígito presente en los tres conjuntos, y se lo muestra como “dígito común”.

Estas expresiones lógicas permitieron transformar condiciones planteadas en lenguaje natural en validaciones precisas dentro del programa, usando herramientas propias del lenguaje Python como `len()` e `intersection()` para analizar conjuntos de datos numéricos de manera eficiente.