**Trabajo Integrador 2 – Documento Adicional**

**Materia:** Matemática  
**Profesor/a:** Vanina Durrutty  
**Integrantes del grupo:**

* **Michael Chiappone**
* **Jonatan Campana**

**1. Distribución de tareas**

| **Integrante** | **Tareas realizadas** |
| --- | --- |
| **Michael Chiappone** | Desarrollo de las funciones de lógica matemática: operaciones con conjuntos, frecuencias y suma de dígitos. Análisis y escritura de las expresiones lógicas en lenguaje natural y su traducción a notación matemática. |
| **Jonatan Campana** | Implementación de la lógica en Python (parte A y B), carga de datos, control de flujo y validaciones condicionales. Desarrollo de la lógica para contar pares/impares, detectar años bisiestos y calcular edades. |

Ambos integrantes participaron activamente en la integración del trabajo, revisión final del código y redacción del informe.

**2. Relación entre expresiones lógicas y el código implementado**

**Expresiones Lógicas Parte A – Conjuntos de dígitos del DNI**

* **Expresión 1: "Los dígitos que están en A y no están en B"**
  + **Traducción matemática:** A – B
  + **Código implementado:**

python

CopiarEditar

diferencia\_AB = conjunto\_A.difference(conjunto\_B)

print(f"Dígitos en A y no en B: {diferencia\_AB}")

* + **Correspondencia directa:** El código usa la operación .difference() para representar la diferencia de conjuntos.
* **Expresión 2: "Los dígitos que están en A o en B, pero no en ambos"**
  + **Traducción matemática:** A Δ B (Diferencia simétrica)
  + **Código implementado:**

python

CopiarEditar

diferencia\_simetrica = conjunto\_A.symmetric\_difference(conjunto\_B)

print(f"Dígitos en A o en B, pero no en ambos: {diferencia\_simetrica}")

* + **Correspondencia directa:** La función .symmetric\_difference() refleja la lógica de la expresión.
* **Condición adicional: “Alta diversidad numérica”**
  + **Código:**

python

CopiarEditar

if len(conjunto\_A) > 6:

print("Conjunto A tiene alta diversidad numérica.")

* + **Lógica usada:** Verificación del tamaño del conjunto > 6, como umbral de diversidad.

**Expresiones Lógicas Parte B – Años de nacimiento**

* **Condición 1: "Si todos nacieron después del 2000, mostrar 'Grupo Z'"**
  + **Código:**

python

CopiarEditar

if all(anio > 2000 for anio in anios\_nacimiento):

print("Grupo Z")

* + **Traducción lógica:** ∀x ∈ años\_nacimiento, x > 2000 → "Grupo Z"
* **Condición 2: "Si alguno nació en un año bisiesto, mostrar 'Tenemos un año especial'"**
  + **Código:**

python

CopiarEditar

if any(es\_bisiesto(anio) for anio in anios\_nacimiento):

print("Tenemos un año especial")

* + **Traducción lógica:** ∃x ∈ años\_nacimiento tal que es\_bisiesto(x)
* **Condición 3: "Cantidad de integrantes nacidos en año par o impar"**
  + **Código:**

python

CopiarEditar

if anio % 2 == 0:

pares += 1

else:

impares += 1

* + **Lógica aplicada:** Condición booleana para clasificar según paridad.

**3. Conclusión del análisis**

Las expresiones lógicas desarrolladas en el informe (en lenguaje natural y en notación matemática) fueron implementadas correctamente en Python, respetando la lógica de conjuntos y condicionales. El código muestra cómo los conceptos teóricos de Matemática se pueden trasladar de forma clara y directa a un programa funcional, logrando así el objetivo integrador del trabajo.