Teoría de la Computación
Licenciatura en Informática con Orientación en Desarrollo de Software
Universidad Nacional de Quilmes

Práctica 1
Repaso

Ejercicio 1. Demostrar por inducción en n:

1. $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
2. $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$
3. $n < 2^n$

Ejercicio 2.

1. Sean $w_1, w_2 \in \Sigma^*$ palabras. Por inducción en $|w_1|$ demostrar que $(w_1 \cdot w_2)^r = w_2^r \cdot w_1^r$.
2. Sea $w \in \Sigma^*$ una palabra. Por inducción en $|w|$ demostrar que $(w^r)^r = w$.

Ejercicio 3. En un grafo no dirigido G sea n el número de vértices y m el número de aristas. Demostrar que $m \leq \frac{n(n-1)}{2}$.

Ejercicio 4. Sea Σ un alfabeto y sean $L_1, L_2, L_3 \subseteq \Sigma^*$ lenguajes. Demostrar:

1. $L_1 \cdot (L_2 \cup L_3) = (L_1 \cdot L_2) \cup (L_1 \cdot L_3)$
2. $L_1^* \cdot L_1^* = L_1^*$
3. $(L_1^*)^* = L_1^*$

Ejercicio 5. Sean $f : A \to B$ y $g : A \to B$ funciones y sea $g \circ f : A \to C$ la composición.

1. Demostrar que si f y g son inyectivas, entonces $g \circ f$ es inyectiva.
2. Demostrar que si f y g son suryectivas, entonces $g \circ f$ es suryectiva.
3. Demostrar que si f y g son biyectivas, entonces $g \circ f$ es biyectiva.

Ejercicio 6. Escribimos $A \approx B$ si existe una función biyectiva $f : A \to B$. Demostrar:

1. $A \approx A$
2. $A \approx B \implies B \approx A$
3. $A \approx B \wedge B \approx C \implies A \approx C$
4. $\mathbb{N} \approx \mathbb{N} \setminus \{10\}$
5. $\mathbb{N} \approx \mathbb{N} \times \mathbb{N}$
6. $\mathbb{N} \approx \Sigma^*$ para cualquier alfabeto Σ finito y no vacío
7. $\mathbb{N} \not\approx \mathbb{N}^{\mathbb{N}}$ donde $\mathbb{N}^{\mathbb{N}}$ denota el conjunto de funciones $\mathbb{N} \to \mathbb{N}$

Máquinas de Turing

Ejercicio 7. Recordemos que una máquina de Turing *decide* un lenguaje $L \subseteq \Sigma^*$ si acepta a las palabras que están en L y rechaza a las palabras que no están en L.

1. Definir una máquina de Turing sobre $\Sigma = \{a, b\}$ que decida el lenguaje $\{w \in \Sigma^* \mid w \text{ tiene un número par de } a\text{s}\}$.
2. Definir una máquina de Turing sobre $\Sigma = \{a, b, \#\}$ que decida el lenguaje $\{w\#w \in \Sigma^* \mid w \text{ no tiene ocurrencias de } \#\}$.
3. Definir una máquina de Turing sobre $\Sigma = \{a, b, \#\}$ que decida el lenguaje $\{w\#w \in \Sigma^* \mid w \text{ no tiene ocurrencias de } \#\}$.