Actividad: Ejercicios sobre Máquinas de Turing en Python

CADI: Lenguajes y Autómatas (8º semestre)

Programa: Ingeniería de Sistemas y Computación

Institución: Universidad de Cundinamarca – Seccional Ubaté

Propósito de la actividad

- Comprender el modelo de Máquina de Turing como fundamento de la computación.
- Implementar en Python una clase TuringMachine dinámica, que pueda adaptarse a distintos problemas.
- Resolver mediante simulación los 10 ejercicios planteados en este taller.
- Practicar el trabajo colaborativo en GitHub con ramas y Pull Requests.

Dinámica de clase

1. Fase 1 – Construcción de la clase base
   - Cada grupo debe diseñar en Python una clase TuringMachine que sea flexible:
     • Debe permitir definir estados, alfabeto, funciones de transición, cinta, posición del cabezal y estados finales.
     • Debe incluir un método para ejecutar paso a paso la máquina y otro para correrla completa.

2. Fase 2 – Aplicación a los ejercicios
   - Una vez construida la clase, usarla para simular las 10 máquinas de Turing propuestas.
   - Cada ejercicio requiere definir la función de transición y la lógica particular.

3. Fase 3 – GitHub colaborativo
   - Cada estudiante debe trabajar en su rama personal.
   - Implementar al menos un ejercicio por integrante.
   - Subir cambios a la rama y luego fusionar con main mediante un Pull Request (PR).

Ejercicios a implementar

1. Complemento de un número binario
Diseñar una Máquina de Turing que calcule el complemento a 1 de un número binario.
Entrada: 10101 → Salida: 01010

2. Sucesor de un número en codificación unaria
Diseñar una Máquina de Turing que obtenga el sucesor de un número en codificación unaria.
Entrada: 111 → Salida: 1111

3. Predecesor de un número en codificación unaria
Diseñar una Máquina de Turing que obtenga el predecesor de un número en codificación unaria.
Entrada: 1111 → Salida: 111

4. Paridad de un número binario
Diseñar una Máquina de Turing que calcule la paridad de un número binario.
Si #1's es par → añade 0; si es impar → añade 1.
Ejemplo: 1011 → 10111

5. Contador unario de caracteres {a, b, c}
Diseñar una Máquina de Turing que devuelva tantos 1's como caracteres tenga la palabra.
Entrada: abca → Salida: 1111

6. Copia de una cadena de símbolos {A,B,C}
Diseñar una Máquina de Turing que copie la cadena en la misma cinta.
Entrada: AABC → Salida: AABCAABC

7. Reemplazo de M primeras A's por B's
Diseñar una Máquina de Turing que cambie las M primeras A's por B's.
Entrada: 11AAAAB (M=2) → Salida: 11BBAAAB

8. Comparación de dos palabras separadas por #
Diseñar una Máquina de Turing que verifique si dos palabras separadas por # son iguales.
Entrada: 101#101 → Acepta
Entrada: 110#101 → Rechaza

9. Sucesor de un número binario
Diseñar una Máquina de Turing que obtenga el sucesor de un número binario.
Entrada: 1011 → Salida: 1100

10. Antecesor de un número binario
Diseñar una Máquina de Turing que obtenga el antecesor de un número binario.
Entrada: 1100 → Salida: 1011

Entregables

1. Clase TuringMachine en Python, construida desde cero por cada grupo.
2. Implementación de los 10 ejercicios usando la clase.
3. Repositorio GitHub con ramas, PRs y fusión en main.