# XVI Encuentro Departamental RREDSI

**Semilleros de Investigacion - Risaralda**

## Proyecto Finalizado - PF-038

### Informacion General

**Código:**
PF-038

**Fecha de Registro:**
2026-03-20 15:38:16

**Semillero de Investigación:**
Huellas sobre Sistemas Digitales y Mantenibilidad

### Ponente 1

**Rol Ponente 1:**
Estudiante / Aprendiz

**Nombre Ponente 1:**
Luis Miguel Molina Osorio

**Institucion Ponente 1:**
Universidad Tecnológica de Pereira

**Ciudad Ponente 1:**
Pereira

### Lider / Tutor 1

**Nombre Lider/Tutor 1:**
Ana Julieth Marín Hurtado

**Institucion Lider/Tutor 1:**
Universidad Tecnológica de Pereira

**Ciudad Lider/Tutor 1:**
Pereira

### Clasificacion

**Area del Conocimiento:**
Ingeniería y Tecnología

**ODS:**
4. Educación de Calidad

### Contenido del Proyecto

**Titulo del Proyecto:**
Desarrollo de un entorno interactivo para la simulación de sistemas LADDER

**Palabras Clave:**
Automatización industrial, Ladder, Software.

**Resumen:**
Los sistemas de automatización industrial requieren de entornos de diseño que permitan una validación lógica antes de su implementación en controladores físicos; sin embargo, existe una brecha formativa y operativa entre la teoría de los sistemas de control y la programación de Controladores Lógicos Programables (PLC) bajo estándares industriales. Este documento describe el desarrollo de un software interactivo basado en Lógica de Escalera (Ladder Logic), diseñado para optimizar el diseño rápido de circuitos y la enseñanza de la automatización.
El software, desarrollado en un entorno controlado, implementa un motor de simulación con un algoritmo de barrido (scan cycle) en tiempo real, permitiendo la evaluación dinámica de variables mediante una interfaz gráfica orientada a objetos. Entre las características técnicas principales se incluyen el soporte para contactos normalmente abiertos (NA), cerrados (NC) y temporizadores con retardo a la conexión y desconexión (TON/TOF).
Los resultados experimentales demuestran que el simulador proporciona una retroalimentación visual inmediata con una latencia mínima, lo que mejora la curva de aprendizaje en comparación con entornos de desarrollo industriales más rígidos. El sistema se presenta como una alternativa eficiente para el prototipado rápido, permitiendo verificar la integridad de la lógica de control y el comportamiento de los contactores en un entorno libre de riesgos eléctricos.

## Planteamiento y Justificacion

**Planteamiento del Problema:**
A pesar de la estandarización de la programación de controladores lógicos programables (PLC) bajo la norma IEC 61131-3, cuya cuarta edición fue publicada en 2025 por la Comisión Electrotécnica Internacional (IEC, 2025), persisten dificultades en la validación y comprensión del comportamiento lógico de los sistemas automatizados durante las etapas de diseño, prototipado y formación académica. Sin embargo, sigue siendo la metodología predilecta en ingenierías, debido a su capacidad para representar de forma intuitiva, el flujo de señales y la lógica booleana mediante analogías de contactos y bobinas. No obstante, la transición entre el diseño conceptual de un sistema de contactores y su implementación final en un PLC representan desafíos que las herramientas convencionales no siempre logran mitigar de manera ágil.
Las plataformas industriales de fabricantes líderes operan como ecosistemas cerrados, como Siemens TIA Portal o Rockwell RSLogix, requieren licencias costosas y hardware específico, lo cual restringe su uso en etapas preliminares de diseño y en procesos de enseñanza. Aunque existen alternativas de código abierto como OpenPLC, que implementa el estándar IEC 61131-3 sin restricciones comerciales (Sama y Sangaji, 2022), esta presenta limitaciones funcionales que dificultan la simulación completa del comportamiento lógico de un Diagrama de Escalera.
La gestión de eventos dependientes del tiempo; en particular los temporizadores, representa una de las mayores fuentes de error en la lógica de control (PLC Academy, 2024b); sin una visualización dinámica del tiempo acumulado y del estado de los bits de control, se recurre a procesos de prueba y error. Por otra parte, un problema persistente en la enseñanza y el prototipado, es la incomprensión de cómo el procesador evalúa los escalones de forma secuencial. Formby y Beyah (2020) demuestran que el ciclo de barrido es una característica determinista y única del PLC, cuya emulación es indispensable para identificar conflictos lógicos, como el doble direccionamiento de bobinas o las condiciones de carrera en el automantenimiento de señales.
Además, se identifica una limitación en las herramientas académicas que permitan validar de forma accesible y segura el comportamiento lógico y temporal de sistemas de control industrial sin depender de infraestructura especializada ni de plataformas propietarias.
Se requiere una herramienta con interfaz gráfica intuitiva, de fácil uso y comprensión, capaz de procesar variables de entrada y salida mediante un motor de ejecución con tiempos monotónicos que garanticen la precisión de los temporizadores; como señala Liu (2024) en su repositorio denominado "Python Virtual PLC & RTU Simulator". El objetivo es proporcionar una herramienta que, mediante la manipulación interactiva de componentes en una cuadrícula lógica y la retroalimentación visual del flujo lógico, permita validar arquitecturas de control sin los riesgos eléctricos y mecánicos inherentes a la experimentación en hardware real.

**Justificacion:**
La enseñanza de la automatización industrial en el nivel universitario enfrenta una tensión estructural creciente. Por un lado, los planes de estudio exigen que el estudiante comprenda el comportamiento dinámico de los sistemas de control lógico. Por otra parte, el acceso a equipos físicos como PLC y módulos de entrenamiento sigue siendo limitado debido a restricciones presupuestales y altos costos de licenciamiento de plataformas propietarias (Martínez Balbín y Garcia Curo, 2025).
En este sentido, el Semillero de Investigación Huellas sobre Sistemas Digitales y Mantenibilidad de la Universidad

Tecnológica de Pereira está desarrollando sistemas para la Automatización y Control Industrial por etapas: con el propósito de disponer de una herramienta que permita al estudiante pasar del diseño conceptual de diagramas de escalera a la verificación lógica del circuito, sin que dicha transición dependa de la disponibilidad del hardware ni de la instalación de software con restricciones de licencia.
Además, en ingeniería, la simulación previa al uso de hardware es una práctica esencial en automatización (Pesce, 2025); según trabajos recientes, la ejecución de secuencias en entornos virtuales reduce los errores de programación y garantiza una integración fluida con la maquinaria (Nirtec Studio S.L., 2025). Sin embargo, la mayoría de herramientas comerciales y de código abierto presentan limitaciones para su uso académico; algunas dependen de hardware físico, otras carecen de visualización dinámica del flujo lógico, esencial para comprender la relación entre entradas y salidas; y las más completas, imponen restricciones de licencia o de sistema operativo que las hacen inviables para su distribución en laboratorios universitarios (Echever Baez et al., 2024).
El desarrollo del simulador en Python responde a una justificación de índole práctica e institucional: Python es el lenguaje de mayor penetración en los cursos de ingeniería y ciencia de datos; su ecosistema de herramientas para interfaces gráficas, visualización y prototipado rápido es maduro y bien documentado; y su naturaleza de código abierto elimina cualquier barrera de acceso.
En consecuencia, la formación en automatización industrial continúa limitada por barreras tecnológicas y económicas, dado que persisten dificultades para validar de forma práctica los sistemas de control lógico. Por lo tanto, se fundamenta la necesidad de disponer de una plataforma de apoyo formativo, que reduzca la brecha entre teoría y aplicación.

## Objetivos

**Objetivo General:**

Desarrollar un software interactivo de simulación basado en Lógica de Escalera (Ladder Logic) que permita el diseño, validación lógica y visualización dinámica del comportamiento lógico de circuitos de control.

**Objetivos Especificos:**

Analizar herramientas de simulación de diagramas de escalera para identificar requerimientos funcionales, estándares y criterios de diseño del simulador.

Diseñar una arquitectura funcional del simulador que permita la representación y ejecución lógica de circuitos de control para facilitar su comprensión y validación.

Desarrollar el simulador con capacidades de interacción gráfica y retroalimentación visual en tiempo real para la validación didáctica de sistemas de automatización.

## Marco Teorico y Metodologia

**Referente Teorico:**

Diagrama de Escalera como Lenguaje de Control Industrial
El diagrama de escalera, es uno de los lenguajes de programación gráfica definidos por la norma IEC 61131-3; constituye, a su vez, el lenguaje de mayor adopción histórica en la programación de PLCs a nivel mundial, estimándose que se utiliza en aproximadamente el 80 % de los sistemas de automatización industrial en operación (Wevolver, 2024). Su popularidad no es fortuita; la representación visual del diagrama, dos rieles verticales que simbolizan la alimentación de potencia, conectados entre sí por escalones horizontales (rungs) que contienen los elementos de lógica, reproduce con fidelidad los esquemas de control de relés electromecánicos que precedieron a los controladores programables; de este modo, los técnicos y profesionales con formación en circuitos de control encuentran en el Diagrama de Escalera una transición natural hacia el entorno digital (PLC Academy, 2024a). La IEC 61131-3, define con precisión la sintaxis y la semántica de este lenguaje, estableciendo los elementos mínimos que deben estar presentes en cualquier implementación conforme al estándar: contactos NA, contactos NC, bobinas de salida, bloques de función y los mecanismos de temporización (IEC, 2025).
Elementos Fundamentales: Contactos, Bobinas y Flujo de Potencia
En el modelo de ejecución del Diagrama de Escalera, cada escalón es evaluado como una expresión booleana: el flujo de potencia hipotético (power flow) parte del riel izquierdo, siempre energizado, y avanza de izquierda a derecha a través de los contactos; si la condición lógica del escalón resulta verdadera, dicho flujo alcanza el elemento de salida situado en el extremo derecho y lo activa (Solis PLC, 2024). Los contactos NA cierran el paso de corriente cuando el bit de la variable asociada tiene estado lógico 1 (Alto); los contactos NC, en cambio, cierran el paso cuando el bit es 0 (Bajo), comportándose como la negación lógica de la variable; y las bobinas de salida, al recibir potencia, escriben un valor 1 en la variable de memoria correspondiente, lo que a su vez puede afectar contactos en otros escalones del mismo programa

(PDF Supply, 2023). Esta interdependencia entre escalones es precisamente hace que la comprensión del modelo de ejecución secuencial sea tan importante: un error en el orden de los escalones, o un doble direccionamiento de una bobina en dos escalones distintos, puede producir comportamientos inesperados que sólo se manifiestan bajo condiciones de operación específicas; condiciones que en hardware real pueden ser difíciles y costosas de reproducir de forma controlada (Solis PLC, 2024).

Ciclo de Barrido y su Relevancia para la Simulación

El ciclo de barrido es el mecanismo central de ejecución de un PLC; consiste en una secuencia continua y determinista de tres fases: la lectura del estado de las entradas físicas hacia la tabla de imagen de entradas, la evaluación secuencial de la lógica del programa escalón por escalón, de arriba hacia abajo, de izquierda a derecha, y la escritura de los resultados en la tabla de imagen de salidas (Industrial Monitor Direct, 2025). La naturaleza determinista de este proceso garantiza que toda la lógica evalúe una instantánea coherente del estado de las entradas, capturada al inicio del ciclo; esto evita que un cambio en una variable de entrada durante la ejecución del programa altere el resultado de escalones ya evaluados dentro del mismo barrido (Industrial Monitor Direct, 2025). Emular correctamente este comportamiento en un entorno de simulación por software es un requisito técnico no trivial; requiere que el motor de ejecución mantenga una copia del estado de las variables entre ciclos, actualice dicho estado de manera sincronizada y garantice que el diferencial de tiempo entre ciclos consecutivos se mida con un reloj que no se vea afectado por ajustes del sistema operativo, de ahí el uso de funciones de tiempo monotónico en implementaciones sobre Python (Liu, 2024).

Temporizadores TON y TOF en la Lógica de Control

Los bloques de temporización definen algunos de los comportamientos más representativos de los sistemas de control real; el temporizador con retardo a la conexión (TON) activa su salida una vez que ha transcurrido el tiempo preestablecido (preset) desde el momento en que la entrada se puso en estado activo; el temporizador con retardo a la desconexión (TOF), por su parte, mantiene la salida activa durante el tiempo preset después de que la entrada cae a cero (PLC Academy, 2024b). Ambos bloques son ampliamente utilizados en circuitos de arranque de motores eléctricos, en circuitos de frenado retardado y en la gestión de señales de seguridad con histéresis temporal (PLC Academy, 2024b). Su correcta simulación exige que el motor de cálculo que acumule el tiempo transcurrido a partir de un diferencial medido en cada ciclo de barrido; cualquier imprecisión en la medición de este diferencial introduce errores acumulativos que, en un entorno formativo, distorsionan la comprensión del estudiante sobre el comportamiento real del controlador (Formby y Beyah, 2020).

**Metodologia:**

La metodología se estructuró bajo un enfoque de ingeniería de control y desarrollo de software aplicado a la automatización industrial, dividida en tres fases secuenciales que responden a los objetivos específicos del proyecto.

Fase 1: Revisión del Estado del Arte y Requerimientos

Esta fase se centró en identificar las características funcionales, limitaciones y estándares vigentes de las herramientas de simulación de diagramas de escalera, con el fin de definir los requerimientos del simulador a desarrollar.

Actividad 1.1: Revisión Bibliográfica: Se realizó una revisión de literatura científica e industrial sobre programación de PLC, norma IEC 61131-3, en su edición vigente, y herramientas de simulación de diagramas de escalera.

Actividad 1.2: Análisis Comparativo: Se realiza un análisis comparativo de plataformas comerciales y de código abierto, identificando sus capacidades y restricciones de uso académico.

Actividad 1.3: Identificación de Elementos: Se identifican los elementos mínimos del lenguaje de escalera: contactos, bobinas, pulsadores y temporizadores, con el propósito de definir los componentes mínimos que deberá soportar el simulador.

Fase 2: Diseño de la Arquitectura del Sistema

Una vez realizada la revisión del estado del arte, se tiene como propósito definir la arquitectura modular del simulador, especificando los modelos de datos, la lógica del motor de ejecución y la estructura de la interfaz gráfica. Como resultado, se obtuvo la especificación técnica de la arquitectura del simulador, documentando modelos de datos y flujo de ejecución.

Actividad 2.1: Diseño del Modelo Mediante Clases de Datos: Se definen estructuras de datos modulares para representar componentes, escalones y temporizadores del sistema.

Actividad 2.2: Diseño del algoritmo del ciclo de barrido: Se diseña el motor de simulación replicando el ciclo de barrido del PLC: inicialización lógica, evaluación secuencial, resolución de ramas paralelas y actualización de salidas.

Actividad 2.3: Diseño la estructura de la interfaz gráfica: Se diseña la interfaz gráfica compuesta por paleta de componentes, área de edición, panel de variables y barra de control.

Fase 3: Desarrollo de la Interfaz Gráfica

Se desarrolló una interfaz gráfica que integra edición interactiva de diagramas y visualización dinámica del flujo lógico durante la simulación. Para garantizar la trazabilidad del comportamiento del sistema de manera continua durante la simulación, la interfaz incorpora retroalimentación visual continua sobre el estado de los temporizadores y la activación secuencial de los contactos y bobinas del diagrama.

Actividad 3.1: Implementación del lienzo de edición: Se implementa el lienzo de edición con cuadrícula lógica,

representación de rieles de potencia y codificación visual de estados. Incluye mecanismo de arrastre y validación de posicionamiento de componentes.
Actividad 3.2: Implementación de la Visualización Dinámica de los Elementos: Se implementa la visualización dinámica de componentes durante la simulación, actualizando su representación gráfica según el estado lógico calculado por el motor.
Actividad 3.3: Implementación de las barras de control: Se implementan panel de variables en tiempo real y barra de control con funciones de gestión de simulación y edición de escalones.

## Resultados y Conclusiones

**Resultados y Analisis:**

El desarrollo del proyecto culminó con la obtención de un prototipo funcional del simulador interactivo de lógica de escalera, implementado íntegramente en Python con interfaz gráfica construida sobre la biblioteca PyQt5. El sistema integra, en una única aplicación, los componentes fundamentales definidos por la norma IEC 61131-3: contactos NA, contactos NC, pulsadores NA y NC, bobinas de salida directa, bobinas negadas y temporizadores TON y TOF.
Las capturas de pantalla obtenidas durante las pruebas evidencian el correcto funcionamiento de la retroalimentación visual del flujo lógico. Los nodos energizados se representan en color verde, los inactivos en gris y las bobinas activas en naranja, lo que permite al usuario identificar de forma intuitiva el estado de cada elemento del circuito en tiempo real. La Figura 1 presenta la interfaz en un estado de edición previo a la simulación, ilustrando la distribución de las diferentes secciones.

La operación del simulador interactivo se divide en cuatro acciones fundamentales que el usuario ejecuta desde la interfaz gráfica principal: la colocación de componentes en el lienzo, el control de la simulación y el manejo de archivos. Cada una de estas acciones se describe a continuación.
La paleta de componentes agrupa los elementos disponibles en cuatro categorías: Contacto, Pulsadores, Bobinas y Temporizadores. Para insertar un componente, el usuario lo arrastra desde la paleta, manteniendo presionado el clic izquierdo, y lo suelta sobre la celda deseada del lienzo de escalones; si el componente arrastrado no es compatible con la celda de destino, la acción se rechaza visualmente, para los contactos acepta cualquier posición exceptuando la última, reservada para las bobinas. Una vez colocado el componente, se abre un cuadro de diálogo de configuración, donde se asigna la etiqueta que identifica la variable asociada. Para eliminar un componente, basta con hacer clic derecho sobre él. Por último, es posible crear ramas en paralelo entre escalones adyacentes mediante la combinación Ctrl + clic sobre un nodo.
La barra de control inferior concentra los botones de gestión del ciclo de simulación, los cuales son: Iniciar, Detener, Un Ciclo, Reset, + Escalón, - Escalón. Y por último se tienen unos botones para la gestión de archivos que son los siguientes: Nuevo, Guardar y Cargar, al guardar el archivo este se guarda con una extensión .json.
Con el propósito de evaluar la pertinencia pedagógica del simulador, se realizó una prueba de usabilidad con estudiantes de Ingeniería Eléctrica de la Universidad Tecnológica de Pereira. 15 estudiantes interactuaron con el simulador durante una sesión de hora y media, en la que diseñaron y ejecutaron circuitos de control básico sin instrucción previa sobre el software. Los resultados de la encuesta aplicada al finalizar la sesión indicaron que el 100% de los participantes consideró que la retroalimentación visual del flujo lógico facilitó la comprensión del comportamiento del ciclo de barrido.
Los diseños realizados fueron los siguientes: Verificación de flancos de subida, bajada y la función Toggle, se pueden observar en las Figuras 2, 3 y 4, respectivamente.

**Conclusiones:**

El desarrollo del simulador interactivo de lógica de escalera demostró que es posible implementar, sobre tecnologías de código abierto, un entorno de simulación de circuitos de control que emula con fidelidad el comportamiento determinista del ciclo de barrido de un controlador lógico programable.
El análisis comparativo de las plataformas de simulación existentes, confirmó la existencia de una brecha funcional y económica que justifica el desarrollo de una herramienta alternativa. Este análisis permitió definir con precisión los requerimientos funcionales del simulador y orientar las decisiones de diseño hacia las carencias más relevantes para el entorno formativo.
La arquitectura modular orientada a objetos adoptada demostró ser una decisión de diseño acertada: garantiza la mantenibilidad del sistema, facilita la incorporación futura de nuevos elementos del lenguaje de escalera y establece una base tecnológica extensible para las etapas posteriores del proyecto de automatización industrial del Semillero Huellas sobre Sistemas Digitales y Mantenibilidad. La validación de los circuitos de referencia, detección de flancos de subida y bajada mediante pulsadores, y la función Toggle, confirmó la coherencia del modelo de ejecución implementado con la semántica formal del estándar IEC 61131-3.
La prueba de usabilidad realizada con 15 estudiantes de Ingeniería Eléctrica en la materia de Automatización Industrial

de la Universidad Tecnológica de Pereira, en una sesión de hora y media sin instrucción previa sobre el software, arrojó que el 100% de los participantes consideró que la retroalimentación visual del flujo lógico facilitó la comprensión del mecanismo de evaluación secuencial del ciclo de barrido del PLC. Este resultado constituye evidencia preliminar de que el simulador cumple su propósito educativo y reduce efectivamente la brecha entre el diseño conceptual de circuitos de control y comprensión práctica, sin depender de hardware físico ni de plataformas con restricciones de licencia.

**Impactos Alcanzados:**

Impacto Tecnológico: El proyecto aporta una herramienta de software de escritorio que centraliza, en una única aplicación, el diseño interactivo de diagramas de escalera, la ejecución del ciclo de barrido y la visualización dinámica del flujo lógico, eliminando la dependencia de múltiples plataformas propietarias para la validación de circuitos de control. La arquitectura modular orientada a objetos, con separación explícita entre el modelo de datos, el motor de simulación y la capa de visualización gráfica, facilita la extensibilidad del sistema hacia la incorporación futura de nuevos elementos del lenguaje de escalera, como bloques de función, contadores o instrucciones de comparación, posicionando el prototipo como base tecnológica para etapas posteriores del proyecto de automatización industrial desarrollado por el Semillero Huellas sobre Sistemas Digitales y Mantenibilidad.

Impacto Educativo: En consonancia con el Objetivo de Desarrollo Sostenible 4, Garantizar una educación inclusiva, equitativa y de calidad, el simulador se consolida como una herramienta virtual accesible que permite a los estudiantes e investigadores experimentar con la lógica de control de sistemas automatizados en un entorno libre de riesgos eléctricos y mecánicos. La retroalimentación visual inmediata del flujo lógico, la posibilidad de ejecutar la simulación ciclo a ciclo y el panel de monitoreo de variables en tiempo real reducen la curva de aprendizaje asociada a la programación de PLC, favoreciendo la comprensión del modelo de ejecución secuencial del ciclo de barrido sin requerir acceso a hardware físico.

Impacto Económico: La adopción exclusiva de tecnologías de código abierto, Python, PyQt5 y el estándar IEC 61131-3 como referencia normativa pública, elimina por completo las barreras económicas asociadas al licenciamiento de plataformas industriales de propietario, cuyo costo de adquisición y mantenimiento resulta inviable para laboratorios universitarios con presupuestos limitados. Al no requerir hardware especializado ni módulos de expansión, el simulador puede desplegarse en cualquier equipo de cómputo convencional disponible en los laboratorios de la institución, reduciendo significativamente la inversión necesaria para ofrecer prácticas de automatización industrial en el nivel universitario.

## Bibliografia

**Bibliografia:**


Echever Baez, J. C., Astudillo Guayllasaca, H. O., y Morales Montero, J. F. (2024). Caracterización de la perspectiva sociológica, tecnológica y metodológica del proceso de automatización industrial. Ciencia Latina Revista Científica Multidisciplinar, 8(5). https://doi.org/10.37811/cl_rcm.v8i5.13638

Formby, D., y Beyah, R. (2020). Temporal execution behavior for host anomaly detection in programmable logic controllers. IEEE Transactions on Information Forensics and Security, 15, 1455-1469. https://doi.org/10.1109/TIFS.2019.2940890

Industrial Monitor Direct. (2025). PLC scan cycle explained: Scan, logic execution, I/O update methods. https://industrialmonitordirect.com/blogs/knowledgebase/plc-scan-cycle-characteristics-scan-logic-continuous-update-io-copy

International Electrotechnical Commission. (2025). IEC 61131-3:2013/AMD1:2025 - Programmable controllers - Part 3: Programming languages (4.ª ed.). IEC. https://www.iec.ch

Liu, Y. (2024). PLC and RTU simulator: A cross-platform Python library for OT device emulation (v0.1.3) [Software]. GitHub. https://github.com/LiuYuancheng/PLC_and_RTU_Simulator

Martínez Balbín, L., y Garcia Curo, G. (2025). Implementación didáctica de autómatas programables para la optimización de procesos industriales. Journal of Scientific and Technological Research Industrial, 6(1), 25-34. https://doi.org/10.47422/jstri.v6i1.58

Nirtec Studio S.L. (2025). Simulation training: what it is, how it works and its benefits. https://www.nirtec.com/blog/simulation-training/

PDF Supply. (2023). Ladder logic basics: What you need to know to get started. https://www.pdfsupply.com/blog/index.php/2023/10/09/ladder-logic-basics-what-you-need-to-know-to-get-started/

Pesce, J. A. (2025). Simulación en Automatización Industrial. Electromecánica SEI. https://servicioselectromecanicossei.com/simulacion-en-automatizacion-industrial/

PLC Academy. (2024a). Ladder logic tutorial. https://www.plcacademy.com/ladder-logic-tutorial/

PLC Academy. (2024b). Timers in PLC programming: TON, TOF and TP explained. https://www.plcacademy.com/plc-timers/

Sama, A., y Sangaji, C. S. (2022). Towards Industry 4.0 - PLC programming with open-source IEC 61131-3 tools. Medium

/ Industrial Automation Series.
https://andisama.medium.com/towards-industry-4-0-2-plc-programming-hello-world-ce8e35a49944
Solis PLC. (2024). Ladder logic decoded: A step-by-step tutorial for PLC programmers.
https://www.solisplc.com/tutorials/ladder-logic-tutorial-for-plc-programmers
Wevolver. (2024). Ladder logic programming: A comprehensive guide.
https://www.wevolver.com/article/ladder-logic-programming

## Autores

**Autor 1 - Nombre:**
Molina Osorio, Luis Miguel

**Autor 1 - Institucion:**
Universidad Tecnológica de Pereira

**Autor 2 - Nombre:**
Cifuentes Molano, Michael Felipe

**Autor 2 - Institucion:**
Universidad Tecnológica de Pereira

**Autor 3 - Nombre:**
Holguin Londoño, German Andres

**Autor 3 - Institucion:**
Universidad Tecnológica de Pereira

**Autor 4 - Nombre:**
Holguin Londoño, Mauricio

**Autor 4 - Institucion:**
Universidad Tecnológica de Pereira

**Autor 5 - Nombre:**
Ortega Quiñones, Kevin David

**Autor 5 - Institucion:**
Universidad Tecnológica de Pereira

## Documentos de Soporte

**URL Carta de Aval:**
https://drive.google.com/file/d/1QUYxYYjwpX1ySJTQ9tFCNc0CqV9XM0je/view?usp=drive_link