

INTRODUCCIÓN A LA PROGRAMACIÓN



**Control de facturación del consumo de energía para la empresa Sencom**

**Integrantes:**

Sara Valeria Ruiz Castillo.

Marco Antonio Salas Fonseca.

Osman Jatniel Cerpas Alvarado.

Víctor Mateo Alcocer López.

**Docente:**

MSc. Silvia Gigdalia Ticay López.

**Fecha:**

Managua, viernes 4 de Julio de 2025

**Índice**

[**I.** **Introducción** 5](#_Toc202474958)

[**II.** **Planteamiento del problema** 6](#_Toc202474959)

[***Proceso a automatizar*** 6](#_Toc202474960)

[**III.** **Objetivos.** 7](#_Toc202474961)

[Objetivo General 7](#_Toc202474962)

[Objetivos específicos 7](#_Toc202474963)

[**IV.** **Justificación** 8](#_Toc202474964)

[**V.** **Análisis del problema** 9](#_Toc202474965)

[¿Qué entradas se requieren? 9](#_Toc202474966)

[¿Cuál es la salida deseada? 9](#_Toc202474967)

[¿Qué método produce la salida deseada? 9](#_Toc202474968)

[Requisitos adicionales y restricciones 9](#_Toc202474969)

[Diagrama de estructura. 10](#_Toc202474970)

[Requerimientos funcionales 11](#_Toc202474971)

[Entradas requeridas 11](#_Toc202474972)

[Salidas deseadas 11](#_Toc202474973)

[*Método que produce la salida deseada* 12](#_Toc202474974)

[Validación estricta de entradas: 12](#_Toc202474975)

[Cálculos de producción energética: 12](#_Toc202474976)

[Cálculo de facturación: 12](#_Toc202474977)

[Emisión del reporte: 12](#_Toc202474978)

[Requisitos adicionales y restricciones 12](#_Toc202474979)

[Requerimientos no funcionales. 13](#_Toc202474980)

[**VI.** **Fase Algoritmo** 15](#_Toc202474981)

[Definiciones 15](#_Toc202474982)

[Validación de nombre. 15](#_Toc202474983)

[Validación de dirección. 16](#_Toc202474984)

[Capacidad planta. 16](#_Toc202474985)

[Tiempo de funcionalidad de la planta. 17](#_Toc202474986)

[Cálculos. 18](#_Toc202474987)

[**VII.** **Ejecución del programa.** 22](#_Toc202474988)

[Documentación General del Sistema de Monitoreo Inteligente de Energía Renovable 22](#_Toc202474989)

[1. Módulo interfaz.py 22](#_Toc202474990)

[2. Módulo reporte.py 22](#_Toc202474991)

[3. Módulo usuarios.py 23](#_Toc202474992)

[4. Módulo principal main.py 23](#_Toc202474993)

[Aspectos Técnicos Destacados 23](#_Toc202474994)

[**VIII.** **Conclusión** 25](#_Toc202474995)

[**IX.** **Recomendaciones** 26](#_Toc202474996)

[**X.** **Anexos.** 27](#_Toc202474997)

[**XI.** **Referencias bibliográficas.** 32](#_Toc202474998)

# **Introducción**

En la era digital actual, el desarrollo de software se ha consolidado como una herramienta esencial para optimizar procesos, mejorar la eficiencia y responder a las demandas cambiantes del entorno. Su aplicación se extiende a múltiples áreas, permitiendo resolver problemas concretos mediante soluciones tecnológicas personalizadas. A través de la automatización, se reducen errores, se ahorra tiempo y se facilita la toma de decisiones informadas.

En este contexto, la asignatura “Introducción a la programación” propone como actividad final el desarrollo de un caso de estudio que implique la creación de una aplicación de consola para automatizar un proceso real. El equipo ha identificado una situación que puede beneficiarse de la implementación de una solución informática, con el fin de agilizar tareas actualmente realizadas de forma manual.

La propuesta consiste en diseñar un programa en lenguaje Python que permita aplicar los conocimientos adquiridos durante el curso. Esta solución recogerá datos relevantes, los procesa adecuadamente y generará un resultado útil, demostrando la importancia de implementar el desarrollo de software en escenarios cotidianos.

Más allá de su valor académico, este proyecto representa una oportunidad para fortalecer habilidades técnicas y comprender el impacto práctico que tiene la programación en el mundo real y ofrecer una implementación de herramientas más modernas para el sistema de la empresa Sencom.

# **Planteamiento del problema**

En la empresa Sencom, el proceso de facturación del consumo energético se realiza de forma manual, lo que ha generado diversas dificultades. La encargada de esta tarea suele invertir una gran parte de su jornada en realizar cálculos extensos, lo que no solo representa una pérdida de tiempo, sino que también incrementa el riesgo de cometer errores. En ocasiones anteriores, se han producido facturaciones incorrectas, lo que ha generado confusión y desconfianza entre los involucrados.

# ***Proceso a automatizar***

Para resolver esta problemática, se propone automatizar el cálculo del consumo mensual de energía y su equivalente en monto monetario. El sistema desarrollado debe solicitar al usuario datos clave como nombre, dirección, cantidad de meses en funcionamiento y capacidad máxima de generación energética. Con esta información, el programa procesa los datos, validará su formato y generará un reporte final con el monto total a facturar de forma precisa y eficiente.

# 

# **Objetivos.**

## Objetivo General

* Diseñar un programa de consola en lenguaje de programación Python para automatizar el proceso de facturación del consumo de energía renovable de la empresa Sencom.

## Objetivos específicos

* Desarrollar un algoritmo que permita mostrar la lógica del problema, para posteriormente, traducirlo a lenguaje de programación python.
* Aplicar conceptos y técnicas de programación para asegurar el funcionamiento eficiente del programa.
* Documentar el desarrollo del proyecto, incluyendo la implementación del código y las decisiones técnicas tomadas con descripciones con el fin de facilitar su comprensión y evaluación.

# **Justificación**

La automatización de procesos se ha convertido en una necesidad clave para mejorar la eficiencia y precisión en las tareas administrativas dentro de cualquier organización. En este sentido, el presente proyecto busca desarrollar una solución informática que permita optimizar el proceso de facturación del consumo energético en la empresa Sencom, el cual actualmente se realiza de manera manual, consumiendo tiempo excesivo y siendo propenso a errores.

Implementar un sistema automatizado no solo facilitará el trabajo del personal encargado, sino que también garantizará cálculos más precisos, reduciendo el margen de error y mejorando la confiabilidad del servicio. Además, este proyecto representa una oportunidad de aplicar conocimientos adquiridos en la asignatura “Introducción a la programación”, mediante la creación de una herramienta útil en un contexto real.

La propuesta aporta valor tanto a nivel académico como profesional, ya que promueve el uso de la tecnología para resolver problemas cotidianos, fomenta el pensamiento lógico y refuerza competencias en el desarrollo de software. Su ejecución tiene el potencial de impactar positivamente en la productividad de la empresa y en la calidad del trabajo del equipo administrativo.

# **Análisis del problema**

## ¿Qué entradas se requieren?

* ***Usuario y contraseña:*** (texto) Credenciales de acceso previamente establecidas.
* ***Datos del cliente:***
  + Nombre (texto)
  + Dirección (texto)
  + Capacidad de la planta (número decimal o entero)
  + Número total de meses de operación (entero positivo)

## ¿Cuál es la salida deseada?

* Código único asignado al cliente.
* Reporte de producción (en pantalla o archivo).
* Cálculo de recuperación de inversión (número o mensaje).
* Gráfico de producción mensual (visualización).
* Guía para instalar la librería matplotlib.

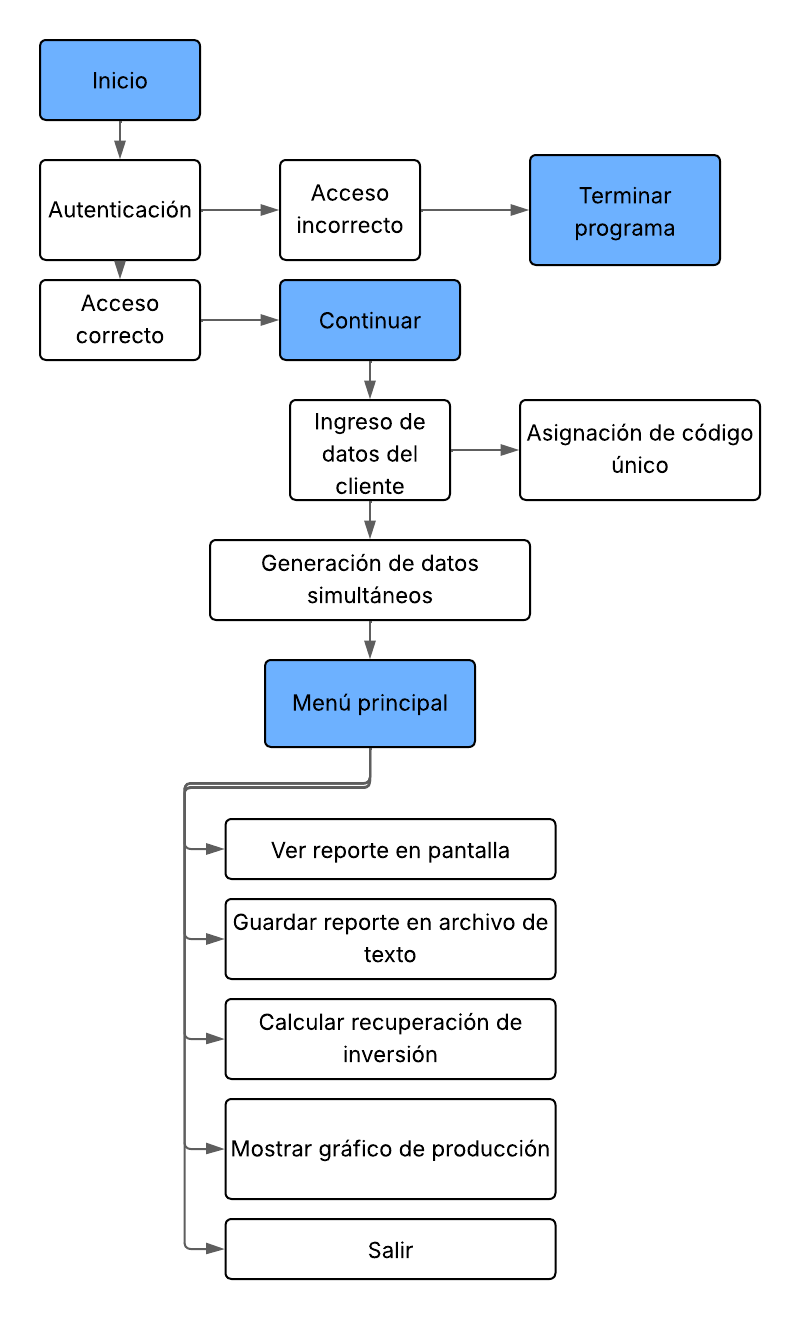
## ¿Qué método produce la salida deseada?

* ***Simulación de datos:*** como no hay base de datos, los datos de producción mensual se generan de forma simulada.
* ***Condicionales y menús:*** para seleccionar las opciones deseadas.
* ***Gráficos*:** uso de la librería matplotlib para generar visualizaciones.
* ***Archivo de texto:*** guardar resultados mediante escritura en archivos.

## Requisitos adicionales y restricciones

* El acceso está restringido mediante autenticación (usuario y contraseña).
* El menú debe ofrecer seis opciones claras.
* El gráfico solo puede visualizarse si la librería matplotlib está instalada.
* El programa debe generar datos de manera ficticia para compensar la ausencia de base de datos.
* Los reportes deben ser claros y estar correctamente formateados.

## Diagrama de estructura.



## Requerimientos funcionales

### Entradas requeridas

|  |  |  |
| --- | --- | --- |
| Entrada | Tipo de dato | Descripción |
| Nombre del cliente | Texto (String) | Validado para que solo contenga letras y espacios |
| Dirección del cliente | Texto (String) | Debe contener solo letras, números y espacios |
| Capacidad de la planta | Número→ Real | Valor numérico positivo con o sin decimales (se convierte a número) |
| Meses de funcionamiento | Número → Entero | Cantidad de meses que la planta ha estado operando |
| Consulta de acumulado (hasta cierto mes) | Número → Entero | Muestra la consulta de facturación de un mes en específico. |

### Salidas deseadas

El algoritmo genera varios reportes mediante instrucciones, con los siguientes tipos de resultados:

|  |  |  |
| --- | --- | --- |
| **Salida** | **Tipo de dato** | **Cantidad** |
| Reporte resumido con información del proveedor y cliente | Texto | 1 |
| Producción energética mensual estimada por mes | Real (kWh) | n (número de meses) |
| Factura mensual actual | Real (USD) | 1 |
| Facturación acumulada (general o hasta cierto mes) | Real (USD) | 1 |
| Producción acumulada total | Real (kWh) | 1 |
| Corte de producción del mes anterior | Real (kWh) | 1 |

## *Método que produce la salida deseada*

El programa sigue estos pasos principales:

### Validación estricta de entradas:

Asegura que los datos ingresados sean adecuados en formato y contenido.

### Cálculos de producción energética:

* + Calcula la eficiencia de la planta con un 20% de su capacidad.
  + Usa promedio de 4.5 horas solares diarias y 30 días al mes para estimar producción mensual base.
  + Aplica una variación aleatoria del ±5% para simular fluctuaciones reales usando Aleatorio().

### Cálculo de facturación:

* + Multiplica la producción mensual por el precio unitario del kilovatio ($0.13).
  + Suma progresivamente para generar el total acumulado.

### Emisión del reporte:

Muestra los datos en consola con formato estructurado.

### Requisitos adicionales y restricciones

* **Validaciones obligatorias** en todas las entradas (no vacías, sin símbolos incorrectos, formato numérico adecuado).
* **Restricciones de tipo**:
  + La capacidad de planta no puede tener más de un punto decimal ni comenzar/terminar con uno.
  + Los meses deben ser enteros positivos.
* **Interfaz de consola**: No incluye interfaz gráfica; toda la interacción es por texto.
* **Persistencia no requerida**: No guarda los datos en archivos (a menos que se agregue en otra fase).
* **Escalabilidad limitada**: Está diseñado como prototipo básico, no multiusuario ni con almacenamiento masivo.

## Requerimientos no funcionales.

El sistema muestra una pantalla de bienvenida estilizada con pyfiglet y una barra de carga animada antes de iniciar.

permite registrar nuevos usuarios, almacenando su nombre de usuario y contraseña en el archivo usuarios.txt.

Tras el inicio de sesión exitoso, el sistema solicita datos del cliente: nombre, dirección, capacidad de la planta (kW) y número de meses de operación.

valida que los nombres solo contengan letras y espacios, y que los valores numéricos (enteros y decimales) sean mayores que cero.

calcula la producción energética mensual simulada usando una fórmula basada en eficiencia y horas solares, y acumula los valores mensuales.

permite al usuario ver un reporte resumido en pantalla, mostrando información clave como producción mensual, acumulada y facturación.

permite guardar el reporte en un archivo de texto (.txt), personalizado con el nombre del cliente.

Calcula y muestra el número de meses necesarios para recuperar una inversión, según la facturación mensual promedio.

Ofrece un menú para visualizar un gráfico de barras con la producción mensual usando matplotlib.

El menú principal permite al usuario salir del sistema en cualquier momento, además de repetir acciones sin reiniciar el programa.

El sistema ofrece una interfaz clara en español, con íconos y mensajes amigables para el usuario.

El código utiliza os.name para hacer limpieza de pantalla compatible con Windows y Unix (cls / clear).

Las contraseñas se ingresan de forma oculta utilizando la biblioteca pwinput, evitando que se muestren en pantalla.

La información se guarda en archivos de texto (usuarios.txt y reporte\_<cliente>.txt) con codificación utf-8.

El sistema maneja errores de entrada con validaciones, mostrando mensajes claros sin detener la ejecución.

El código está modularizado en archivos separados (interfaz.py, usuarios.py, reporte.py, main.py), lo que mejora la mantenibilidad.

# **Fase Algoritmo**

## Definiciones

Son las variables a las cuales se le estarán asignando valores a lo largo del proceso del sistema. Se definen respecto al valor que se van a trabajar en ellas como en “nombre\_cliente”, ”dirección” son definidos caracteres porque son datos que se necesitan guardar en formato de carácter porque son nombres.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

## Validación de nombre.

Se estarán creando bucles para la correcta digitación de las entradas.

Para “nombre\_cliente” primero se da inicio al bucle por medio de una “llave” que es un valor lógico en una variable.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

recorre la entrada por la cantidad de caracteres verificando uno por uno que sean todos letras, también tiene su añadido que le permite ingresar letras acentuadas y la “ñ”, para que no den error en el español.

En caso de no ingresar datos válidos, el bucle se repite hasta que el valor de la llave vuelva a ser verdadero para dar fin al bucle y seguir el programa.

## Validación de dirección.

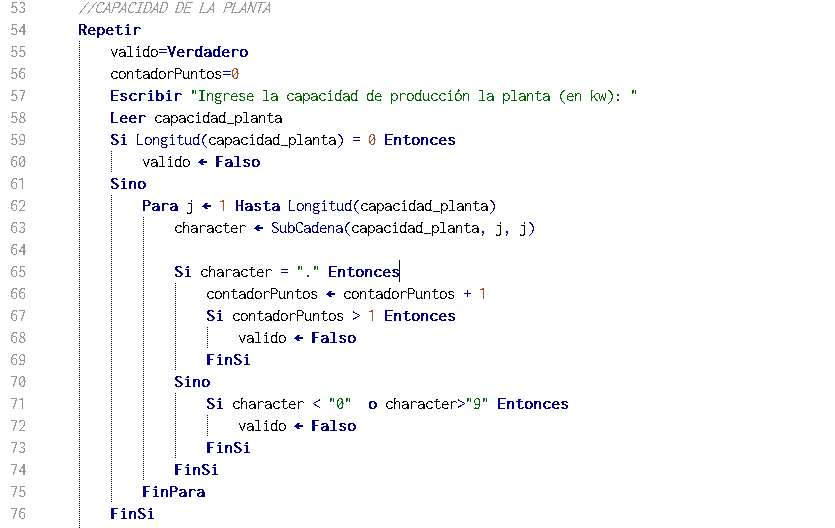
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

El mismo procedimiento sucede con el bucle para digitar correctamente dirección, con el agregado de que uno puede digitar espacios y números.

## Capacidad planta.

Para “capacidad\_planta”, el mismo concepto de la autenticación del dato de entrada es parecida con el añadido de la variable contador puntos para verificar la cantidad en decimal de los KW, evitando errores en la entrada del valor (Más de un punto decimal).

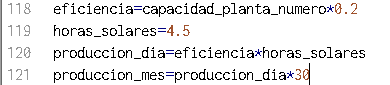


## Tiempo de funcionalidad de la planta.

**Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.**

## Cálculos.

****

Se calcula la eficiencia de la planta (restándole un 80% debido a factores como pérdidas por instalación, eficiencia de paneles e inversores, entre otros.

La eficiencia obtenida se multiplica por las horas solares promedio en que un panel produce energía (4.5 horas estimadas)

Se multiplica lo obtenido para encontrar la producción solar en un mes

Se definen los limites (superiores e inferiores) para posteriormente, utilizar la Función random.

Para definir los límites, se toma en cuenta una desviación estándar del 5%, es decir, que los demás datos de producción mensual pueden variar en un 5% aproximadamente, basados en el valor central (producción en el mes, que lo tomamos como promedio de producción mensual).

Se multiplica la producción solar en un mes por (1-0.05) para límite inferior, y por (1+0.05) para el superior

El "1" representa el 100% del valor actual, y como le queremos restar 5%, aplicamos "(1-0.05), lo mismo para el límite superior

De esta forma obtenemos valores lógicos generados por random, que mostrarán la producción estimada de cada mes, en función de la capacidad de la planta.

Creamos un arreglo para guardar la producción mensual.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Luego convertimos las entradas de números asignándoles valor como mes. Y en el segundo bloque calcula cual fue el mes anterior al solicitado.



El programa pregunta si el usuario quiere ver la facturación acumulada hasta un mes específico.

- Si responde “Sí”, se valida el número del mes ingresado (debe ser un número positivo entre 1 y el mes actual).

- Si la entrada es válida, se calcula la facturación acumulada sumando la producción de cada mes hasta el mes elegido, y se muestra el nombre de ese mes.

- También se genera un reporte con fecha actual, detalles del proveedor, datos del cliente, producción de energía, y facturación correspondiente.

Si el usuario responde “No”, se genera un reporte estándar sin calcular facturación acumulada, solo la producción y cobro del mes actual.

Si la respuesta es inválida o está vacía, se muestra un error y se repite la pregunta.

Y se muestra luego las salidas dependiendo de la decisión de quien trabaja el algoritmo.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

La diferencia en las salidas está en la facturación acumulada en cierto mes (especificado por el usuario).

# **Ejecución del programa.**

## Documentación General del Sistema de Monitoreo Inteligente de Energía Renovable

El presente sistema fue diseñado con el objetivo de brindar una solución sencilla e interactiva para el monitoreo y reporte de la producción energética en plantas solares. El sistema está implementado en Python y está organizado en varios módulos funcionales: interfaz, reporte, usuarios y un módulo principal main. A continuación, se describe cada uno de ellos, incluyendo su funcionalidad esencial y los aspectos técnicos más relevantes.

## 1. Módulo interfaz.py

Este módulo está encargado de la interacción visual inicial con el usuario. Utiliza la librería pyfiglet para mostrar una pantalla de bienvenida estilizada, centrando el texto según el tamaño de la terminal del usuario. Posteriormente, presenta una animación de carga simulada a través de una barra progresiva con caracteres, utilizando time.sleep() y escritura directa con sys.stdout.

Ambas funciones —bienvenida() y barra\_de\_carga()— ayudan a brindar una experiencia visual más atractiva y profesional al usuario desde el inicio del programa. Este módulo no interactúa directamente con los datos, sino que establece el tono visual del sistema.

## 2. Módulo reporte.py

Este es el núcleo funcional del sistema y contiene tanto las rutinas de entrada y validación de datos como la generación de reportes y análisis de inversión. Al ejecutarse, solicita al usuario el nombre, dirección, capacidad de la planta solar y el número de meses en operación. A partir de estos datos, se simula la producción mensual usando valores aleatorios en un rango del 95% al 105% de la producción teórica estimada.

Dentro del módulo se encuentran funciones que permiten:

* Validar entradas del usuario (nombres, valores numéricos).
* Calcular la producción mensual, acumulada y la facturación correspondiente (con un precio fijo de $0.15 por kWh).
* Mostrar el reporte en pantalla con información detallada del cliente, producción y facturación.
* Guardar el reporte en un archivo .txt con nombre personalizado según el cliente.
* Calcular el tiempo estimado para recuperar la inversión según el promedio de facturación mensual.
* Graficar la producción energética mensual mediante la librería matplotlib.

También incluye un menú interactivo (menu()) que guía al usuario por las distintas opciones disponibles. Si la librería matplotlib no está instalada, el sistema ofrece una guía detallada sobre cómo instalarla.

## 3. Módulo usuarios.py

Este módulo gestiona la seguridad del sistema mediante autenticación de usuarios. Permite registrar nuevos usuarios, guardar credenciales en un archivo usuarios.txt y luego validar accesos mediante una rutina de inicio de sesión. Usa la librería pwinput para ocultar las contraseñas ingresadas.

El sistema admite hasta tres intentos de autenticación por usuario. Si las credenciales son correctas, se permite el acceso al módulo de reportes. En caso de no existir ningún usuario registrado, el sistema informa y sugiere realizar un registro primero. Este módulo es fundamental para restringir el acceso a los datos e informes generados.



## 4. Módulo principal main.py

Este archivo es el punto de entrada del programa. Inicializa el entorno visual con las funciones del módulo interfaz, y luego invoca el inicio de sesión a través de usuarios.inicio(). Si la autenticación es exitosa, el sistema continúa con la recopilación de datos e interacción con las funciones del módulo reporte.

También se asegura de limpiar la consola al inicio para brindar una vista más limpia y profesional. Toda la lógica condicional para el flujo del programa se encuentra encapsulada bajo el bloque if \_\_name\_\_ == "\_\_main\_\_":, evitando así ejecuciones no deseadas si se importa desde otro script.

### Aspectos Técnicos Destacados

* **Seguridad básica**: Implementación de autenticación mediante archivo local con contraseña oculta.
* **Experiencia de usuario**: Uso de gráficos ASCII y barra de carga para mejorar la interacción visual.
* **Portabilidad**: Funciona tanto en sistemas Windows como Unix/Linux (uso de os.name).
* **Persistencia**: Guardado de reportes en formato .txt con codificación UTF-8.
* **Manejo de errores**: Validaciones robustas para entradas de texto, enteros y decimales.
* **Visualización de datos**: Generación de gráficas de barras con matplotlib.

Este sistema puede ser utilizado como base para aplicaciones más complejas en el monitoreo de energías renovables, incluyendo mejoras como conexión a bases de datos, interfaces gráficas o exportación en formatos como Excel o PDF. Su diseño modular permite que cada componente se desarrolle y mantenga de forma independiente.

# **Conclusión**

El desarrollo de esta aplicación en Python representa una solución efectiva a la problemática de facturación manual en la empresa Sencom, permitiendo automatizar cálculos, reducir errores y optimizar el tiempo de trabajo. A través de este proyecto, se logró aplicar de forma práctica los conocimientos adquiridos en la asignatura “Introducción a la programación”, demostrando cómo el desarrollo de software puede responder a necesidades reales del entorno laboral. Además de su valor técnico, esta experiencia fortaleció habilidades en la lógica, validación de datos y diseño funcional de soluciones informáticas. La propuesta final no solo mejora la eficiencia operativa, sino que también reafirma el potencial de la programación como herramienta clave en la mejora de procesos empresariales.

# **Recomendaciones**

Para aprovechar al máximo el funcionamiento del programa desarrollado, se recomienda:

1. **Implementar una interfaz gráfica web:** Se sugiere desarrollar una versión del sistema con una interfaz gráfica a través de una página web, lo cual facilita su uso por parte del personal y permitiría el acceso desde diferentes dispositivos con conexión a internet.
2. **Utilizar bases de datos para el almacenamiento:** Para mejorar el manejo y conservación de la información a largo plazo, se recomienda integrar una base de datos que permita almacenar los registros de facturación y consumo energético de manera estructurada y segura.
3. **Generar un archivo ejecutable (.exe):** Convertir el programa a un archivo ejecutable permitiría su instalación y uso en equipos sin necesidad de tener Python configurado, facilitando su distribución y ejecución por parte de usuarios no técnicos.
4. **Agregar generación automática de reportes en PDF o Excel:**

Incorporar la opción de exportar los resultados a formatos como PDF o Excel facilitaría la documentación y el envío de los datos de facturación.

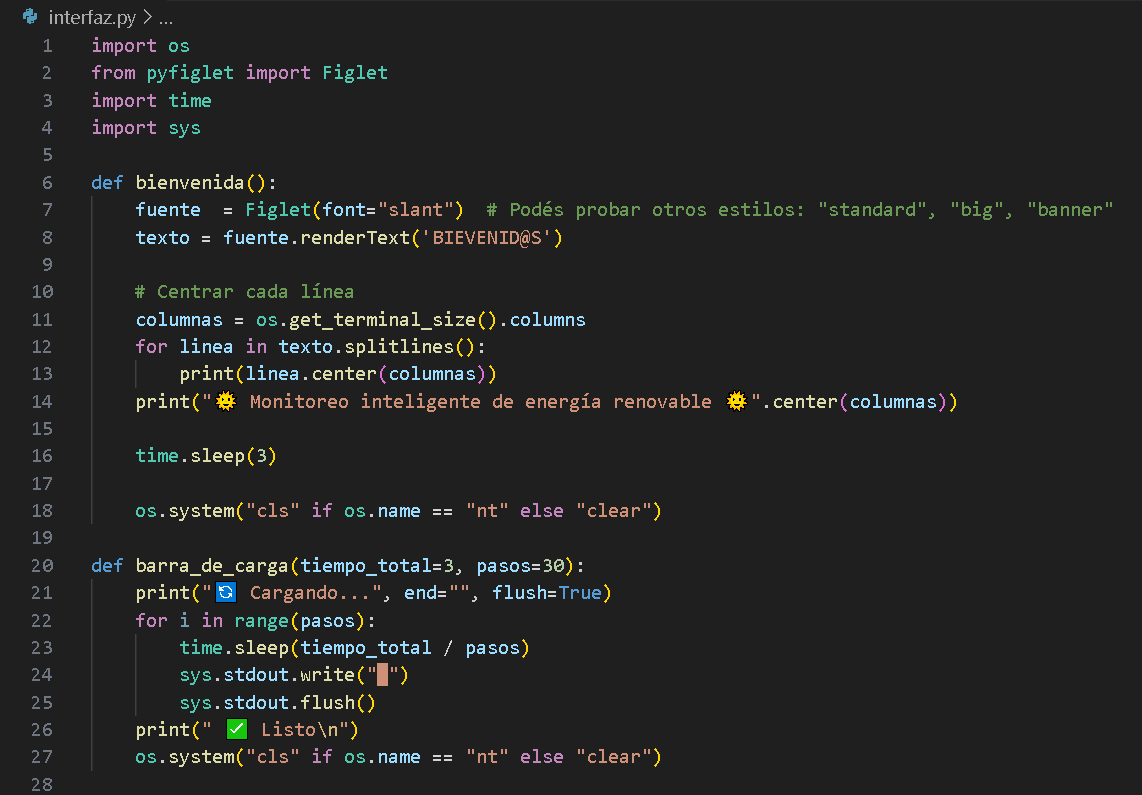
1. **Diseñar una función de respaldo automático de datos:**

Implementar copias de seguridad automáticas garantizaría la protección de los datos en caso de fallos o pérdidas accidentales.

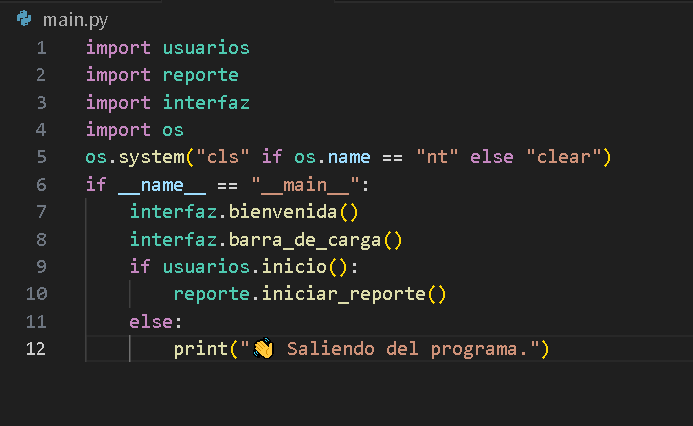
# 

# **Anexos.**

**Módulo interfaz.py.**

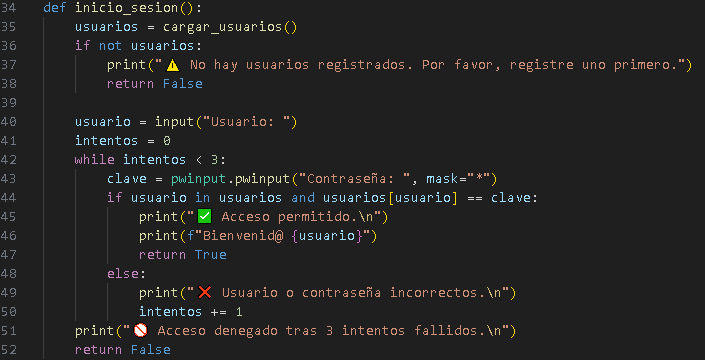
****

**Módulo main.py.**

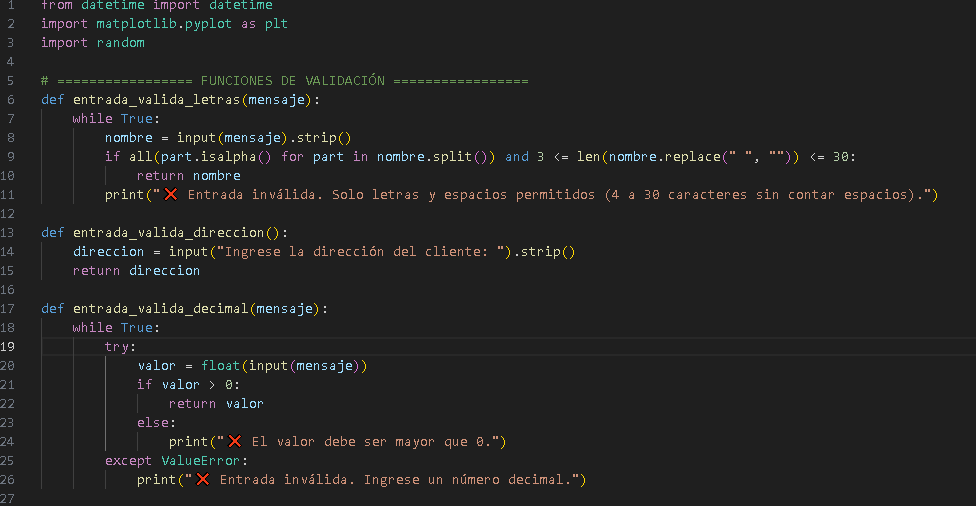


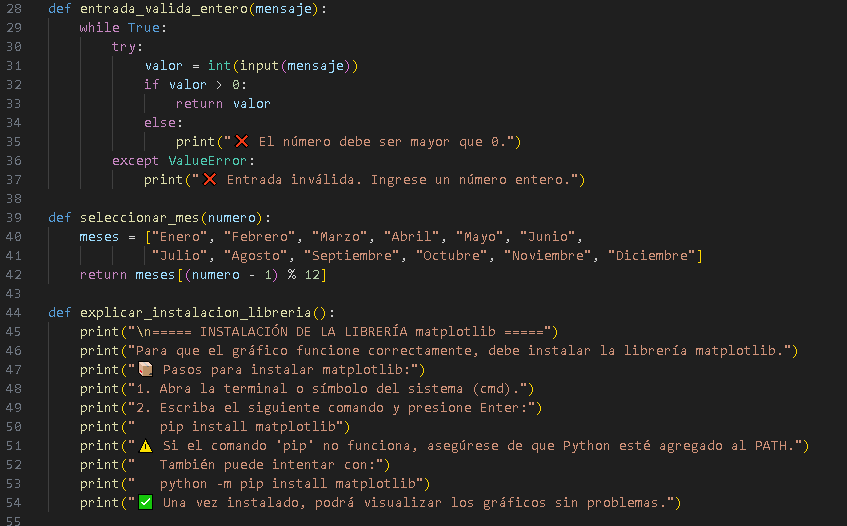
Módulo usuarios.py.

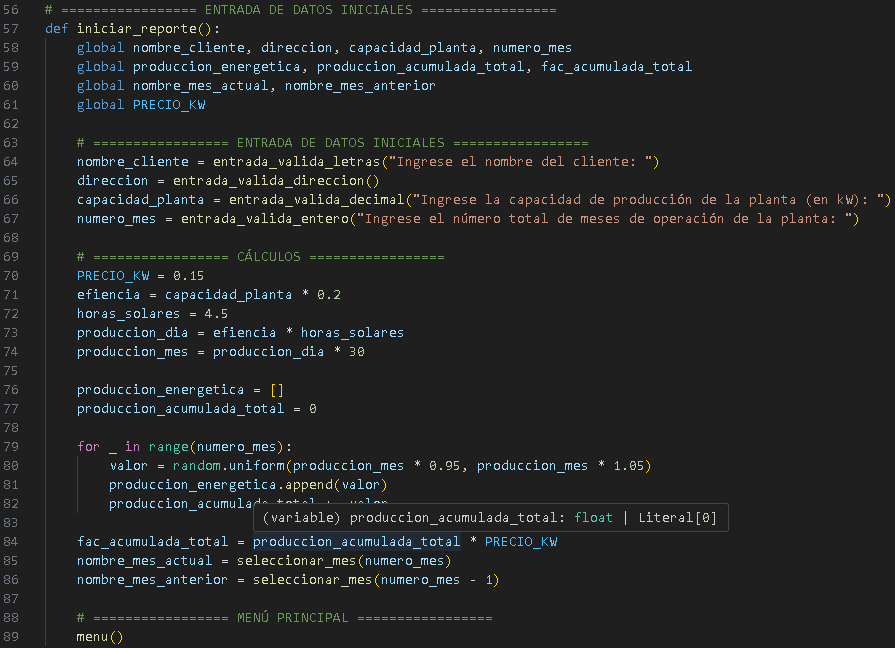


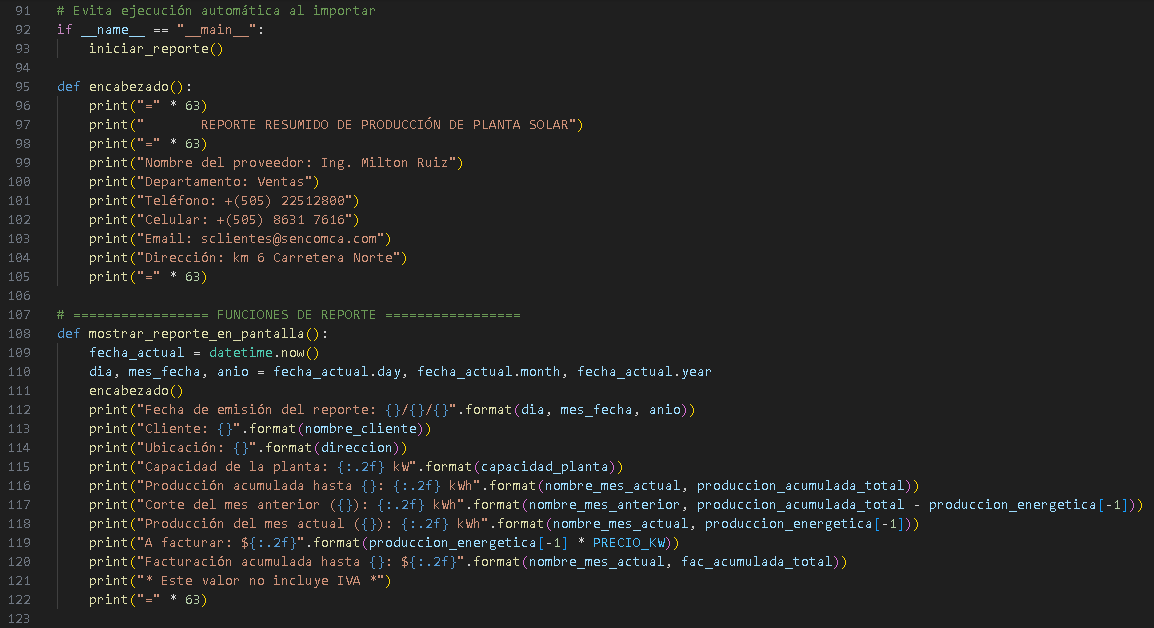


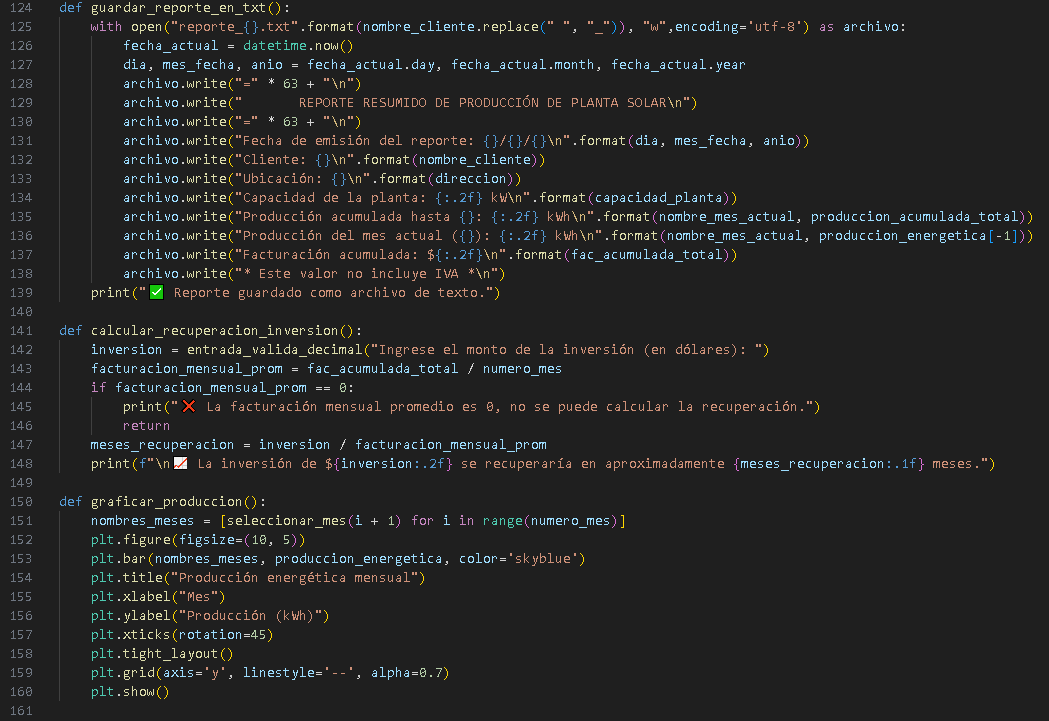
Módulo reporte.py.

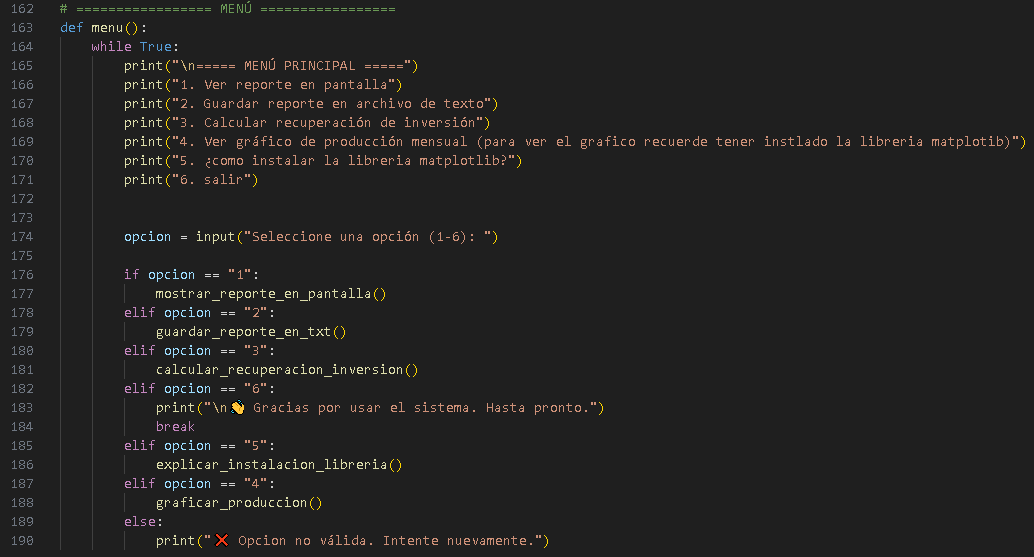












# **Referencias bibliográficas.**

https://www.netacad.com/launch?id=da0847b7-e6fc-4597-bc31-38ddd6b07a2e&tab=curriculum&view=173610eb-7ec1-5972-ac9a-4e5443ccd67f

Python Crash Course A Hands On, Project Based Introduction to Programming - Eric Matthes – 2nd Edition