[[1]](#footnote-1)

PROCESAMIENTO DE DATOS A GRAN ESCALA

Autores: González Gonzáles Dinalut y Ruiz Rincon Jessica Jineth estudiantes de la fundación universitaria de San gil Unisangil (SEDE-CHIQUINQUIRA)

***Resumen*—****En el presente documento hemos diseñado un algoritmo en Python de registro de actividad del usuario donde el usuario ingresa su nombre, hora de entrada y hora de salida. Principalmente aplicamos una metodología donde establecimos el procesamiento de información requerido para el desarrollo del algoritmo. Además, utilizamos PSeInt para crear un pseudocódigo y diagrama de programación modular que luego implementamos en Python. Al ejecutar el programa logramos identificar que el algoritmo registra las veces que el usuario ingresa.**

**Palabras clave-hora salida, hora entrada, registro de usuario, Python. Programación**

INTRODUCCIÓN

 En este documento se busca diseñar un algoritmo que permita calcular cuántos paneles solares necesita una casa para cubrir su consumo energético anual. Para ello es necesario aplicar una metodología que identifique y calcule los datos necesarios para la implementación de un lenguaje de programación como lo es Python.

1. *ANÁLISIS*
2. *Contexto*

En proyectos de ingeniería de sistemas, es común procesar grandes cantidades de datos secuenciales, como logs de acceso o registros de usuarios. Esta actividad emula el análisis de un archivo de logs

*B. Población*

Familia que quiere instalar un sistema de energía solar para cubrir su consumo de electricidad anual y ser más sostenible.

*C. Limitaciones y alcance*

Las limitaciones y alcances que se pudieron identificar son las siguientes:

**Limitaciones**

* Datos ingresados por el usuario: nombre, hora entrada y hora salida.
* Exactitud en los datos ingresados
* Menu de opciones: ingresar datos, mostrar datos, salir

**Alcances**

* El usuario podrá ingresar los datos requeridos.
* Almacenar los resultados donde se muestra nombre y número de entradas

III. OBJETIVOS

1. *Objetivo general*

Diseñar un algoritmo donde el usuario pueda ingresar los datos como nombre, hora de entrada y hora de salida. Almacenar la información y determinar su cantidad de ingresos.

1. *Objetivos específicos*

* Solicitar al usuario los datos: nombre, hora de entrada y salida.
* Determinar el lenguaje de programación que se va a usar para crear el algoritmo.
* Generar la salida correspondiente de los datos.
* Presentar al usuario un programa que pueda manejar de forma comprensible.

IV. ESPECIFICACIÓN DE REQUERIMIENTOS.

1. *CON RESPECTO AL DISPOSITIVO*

A continuación, se indican los requerimientos mínimos:

* Mínimo 4 GB de RAM.
* Sistema operativo: Windows 8 en adelante.
* CPU: Intel Core i3.
* Disco: HDD o SSD.

1. *CON RESPECTO AL USUARIO.*

*Requisitos mínimos*

* El usuario debe ingresar datos requerido sin margen de error.

V. DISEÑO DEL ALGORITMO

Utilizamos un programa generador de pseudocódigo denominado PSeInt, el cual utilizamos como guía, para luego usar el lenguaje de programación Python, siendo este el estructurador del software requerido.

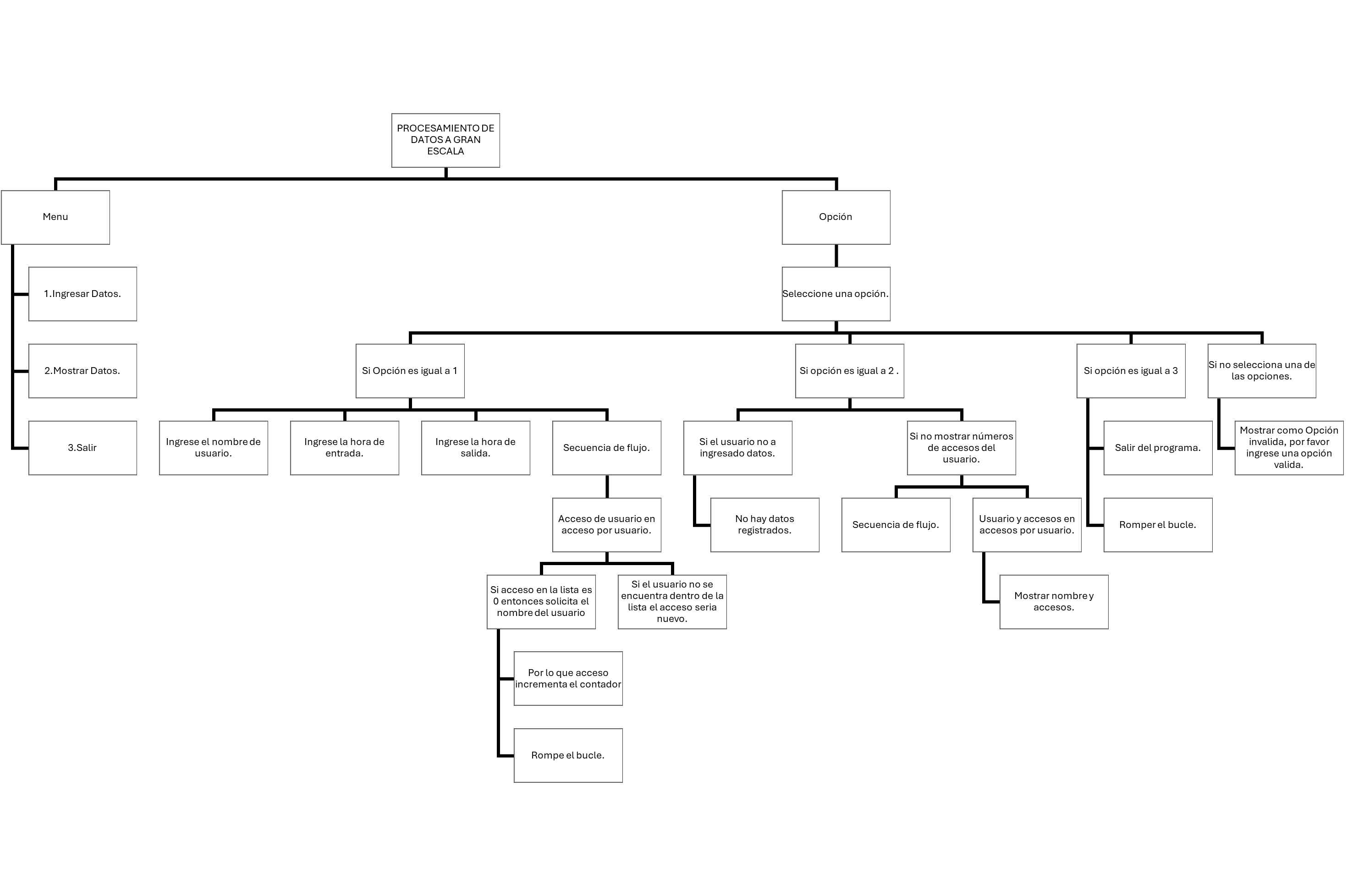


Ilustración .Progamación modular: análisis del problema y solución.

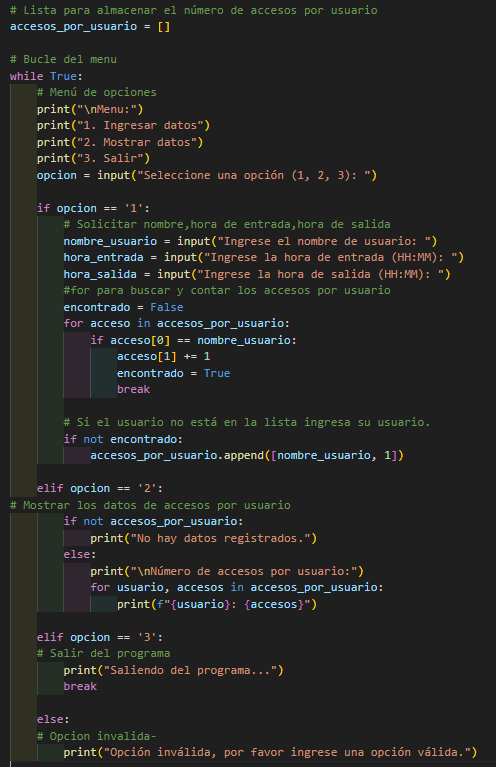
VI.CODIFICACIÓN.

 Para la codificación del algoritmo utilizamos los siguientes programas:

* PSeInt.
* Visual Studio Code.
* Python
* Git hub.

*Instrucciones del código fuente*

1. . Se crea una lista vacía
2. Luego un bucle donde se muestra el menu el cual está conformado por tres opciones.
3. Se crea una variable denominada opción solicitando al usuario que ingrese alguna opción.
4. La opción uno solicita al usuario que ingresé los datos y contabiliza el nivel de acceso del usuario.
5. La opción dos se imprimen el nombre del usuario y el número de acceso.
6. La opción tres sirve para salir del programa



*B. Código fuente*

Ilustración .Algoritmo en Python donde se ejecuta el programa.

*VII. EJECUCIÓN DEL PROGRAMA.*

En este caso comenzamos con la apertura de la terminal de Git Bash. Si ya se creó el repositorio remoto y se sincronizan las cuentas de los desarrolladores con el repositorio local, se envía el archivo al Staging área con el comando git add.  Luego ejecutamos el git commit -m “Comentario” para confirmar los cambios, donde, posteriormente usamos git push origin o pb máster o main dependiendo la rama que se esté usando con el fin de almacenar la información en el repositorio remoto de GitHub.

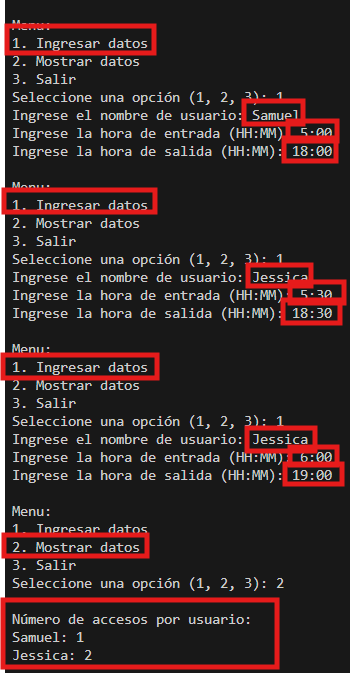


Ilustración .Codigo ejecutado en la ilustración podemos visualizar que él usuario selecciona la opción 1 y diligencia los datos, luego otro usuario ingresa y también diligencia los datos y nuevamente vuelva a ingresar el usuario por lo que se contabilizan.

Nota: en esta imagen se muestra que efectivamente el algoritmo funciona cuando el usuario digita los datos requeridos

**VERIFICACIONES Y DEPURACIONES.**

1. Se verifico la sintaxis dentro del algoritmo donde no se encontraron errores.
2. Verifica si el usuario anteriormente había ingresado su nombre para contabilizar la cantidad de ingresos por parte de este
3. Contabiliza el ingreso del usuario en base a las veces que este ingresa su nombre y hora.
4. Si el usuario no esta en a lista agrega el usuario como nuevo.
5. Si no hay usuarios registrados al mostrar los datos muestra no hay números de acceso por usuario.
6. Si hay usuarios registrados al mostrar los datos los muestra.
7. Si el usuario selecciona la opción tres de salir el terminal le muestra un mensaje donde le indica que esta saliendo por lo que rompe el bucle.
8. Si el usuario pone una opción invalida dentro del menu la muestra al usuario como opción invalida y retorna nuevamente el menu
9. Al ejecutar el código y hacer uso del el funciona correctamente

**Manual de usuario.**

**Al ejecutar el código se le muestra el siguiente menu al usuario:**

Menu:

1. Ingresar datos

2. Mostrar datos

3. Salir

Seleccione una opción (1, 2, 3): 1

En este caso el usuario selecciono la opción 1 por lo que imprime lo siguiente:

**Ingrese el nombre de usuario:**

Por ejemplo:

Ingrese el nombre de usuario: **Emanuel.**

**Lugo el usuario debe presionar enter le solicita lo siguiente:**

**Ingrese la hora de entrada (HH:MM):**

Por ejemplo:

Ingrese la hora de entrada (HH:MM): **5:00**

**Nuevamente presiona enter y le solicita un nuevo dato:**

**Ingrese la hora de salida (HH:MM):**

Por ejemplo:

Ingrese la hora de salida (HH:MM): **18:00**

**Al usuario terminar de ingresar los datos se muestra nuevamente el menu de opciones:**

Menu:

1. Ingresar datos

2. Mostrar datos

3. Salir

Seleccione una opción (1, 2, 3):

Donde el usuario puede agregar nuevamente datos con la opción uno, puede solicitar los datos anexados con la opción 2 o simplemente salir del menu. Hipotéticamente digamos que el usuario desea verificar los dato por lo que selecciona la opción 2 por consiguiente el usuario le imprime lo siguiente y retorna nuevamente el menu para que el usuario pueda seguir ingresando usuarios y visualizar les veces que ha ingresado cada usuario:

**Número de accesos por usuario:**

**Emanuel: 1**

Menu:

1. Ingresar datos

2. Mostrar datos

3. Salir

Seleccione una opción (1, 2, 3):

VIII CONCLUSIONES.

* En el presente trabajo logramos describir el diseño y codificación de un algoritmo el cual permite al usuario calcular el consumo anual generado por sus paneles solares.
* Implementamos un análisis situacional específico para captar con mayor detalle los requisitos del cliente, los cuales gracias a dicho análisis pudimos utilizar para escribir el código de manera correcta
* Incorporamos la evaluación y pruebas para el programa, con ellas constatamos y verificamos que el código realizado por nosotros los desarrolladores esté a fin con los requerimientos del cliente.
* Aprendimos que es necesario aplicar una metodología que hace más comprensible y efectivo el proceso a la hora de desarrollar un algoritmo.
* Empleamos un generador de pseudocódigo denominado PSeInt como guía para estructurar de una forma clara el algoritmo en Python.
* Para realizar la codificación del código utilizamos varios programas como Visual Studio Code y GitHub los cuales facilitan el trabajo en grupo y la gestión de información.

1. Documento elaborado el 15 de septiembre 2024 [↑](#footnote-ref-1)