­**SUSTENTACION DEL TRABAJO**

**2.1**

I

Los condicionales se utilizan en los programas para tomar decisiones basadas en ciertas condiciones

-El if o ‘si’ en español, evalúa una condición y si esta es verdadera ejecuta el código dentro del bloque

-El elif o ‘sino si’ en español, se usa después de un if para evaluar otra condición si la primera fue falsa

-El else o ‘sino’ en español, se ejecuta cuando ninguna de las condiciones anteriores fue verdadera.

**¿Por qué los usamos?**

Permiten que el programa tome decisiones de manera dinámica el cual hace que el código sea más flexible y adaptable a diferentes situaciones

Al Usar elif dentro del código ayuda a optimizar el código disminuyendo algunas líneas de código innecesarias, ahorrando tiempo y facilitando la lectura del código

**2.1**

II

El ciclo while se usa para repetir un bloque de código mientras se cumple una condición.

Este sirve mucho cuando no se sabe cuántas veces se necesita que se ejecute el código, pero sí sabemos que debe repetirse hasta que una condición deje de ser verdadera, lo que evita copiar y pegar el mismo código n cantidad de veces

**¿Por qué los usamos?**

En nuestro caso usamos la estructura While ya que este nos permite el ingreso, consulta, modificación, etc. de constante información hasta que el usuario decida cerrar el programa, o mejor dicho cerrar el bucle. Brindándole una experiencia agradable al usuario

**2.2**

I

Un diagrama de flujo es una representación gráfica de un proceso o algoritmo, el cual utiliza símbolos y flechas para mostrar el flujo de ejecución de un programa

**¿Por qué los usamos?**

Por lo practico que puede llegar a ser este diagrama, fue de gran ayuda durante el desarrollo del programa y más aún cuando surgían algunos inconvenientes al ejecutar el código en Python, lo que nos permitió encontrar los errores de una forma más rápida

**2.2**

II

Estos diagramas usan símbolos, los cuales cada uno tienen un significado, el significado puede variar según el creador del diagrama pero normalmente son: el óvalo para el inicio y fin del proceso ; el paralelogramo para la entrada y salida de datos ; el rectángulo para representar una acción o proceso a ejecutar ; el rombo para decisiones (if, else) ; y las flechas indican la dirección del flujo

**¿Cómo lo usamos?**

En nuestro caso usamos solamente 3 simbolos con la diferencia que cada uno de estos cuenta con un color diferente:

el rectángulo de color amarillo representa la declaración de las variables individuales ;

el rectángulo de color azul representa el Imprimir, equivalente de print() en Python ;

el rectángulo de color morado representa el ingreso de un valor y asignárselo a un variable ;

el rectángulo de color azul turquesa representa la actualización del valor de una variable ;

el rectángulo de color verde representa una variable a la que previamente se le hace un calculo;

el rombo de color gris representa el if;

el rombo de color naranja representa los elif;

el rombo de color rojo representa los if anidados;

y por ultimo, las flechas indican el flujo del programa

**2.3**

I

El sistema propuesto permite gestionar el presupuesto asignado a diferentes regiones mediante una interfaz de menú interactivo; su diseño se basa en una estructura de opciones numeradas, lo que facilita la navegación y ejecución de acciones específicas de manera intuitiva.

**¿Cómo lo logramos?**

El menú principal presenta siete opciones que cubren todas las funciones clave del sistema:

1. Ingresar presupuesto inicial → Permite registrar los montos asignados a cada región.
2. Consultar el estado del presupuesto → Muestra los valores actuales asignados a cada región.
3. Modificar presupuesto de una región específica → Permite actualizar el presupuesto de una región seleccionada.
4. Redistribuir presupuesto equitativamente → Ajusta el presupuesto para que todas las regiones tengan la misma cantidad.
5. Ajustar presupuesto de una región → Permite hacer cambios específicos en una región sin afectar a las demás.
6. Clasificar presupuesto total → Muestra el presupuesto total y verifica si está equilibrado.
7. Salir → Finaliza la ejecución del sistema.

**2.3**

II

Mediante al uso de este sistema se logra garantizar posibles actualizaciones en un futuro, si se le desea agregar más secciones al programa, este adaptándose a diferentes requisitos sin necesidad de rediseñarlo por completo

**¿Cómo lo haríamos?**

Al menú se le podrían agregar las siguientes opciones:

Establecer Límites de Gasto por Región, definir un tope máximo y mínimo de presupuesto para cada región

Ajustar Presupuesto Basado en Desempeño, modifica automáticamente el presupuesto según las ventas de la región

**3.1**

I

La prueba de escritorio es una técnica utilizada para evaluar la lógica del sistema antes de su implementación; esta consiste en simular la ejecución del programa paso a paso, registrando el estado de las variables en cada iteración para verificar que los resultados sean los esperados.

**3.1**

II

Mediante la prueba de escritorio se concluyó que:

las variables mantienen valores coherentes después de cada acción;

cada opción del menú realiza la operación esperada sin interrupciones.

las actualizaciones y ajustes reflejan correctamente los cambios en las variables.

al finalizar el proceso, la información se guarda correctamente, asegurando la integridad del presupuesto.

**¿Cómo lo logramos?**

1. Definición del Escenario de Prueba
2. Simulación de Ejecución Paso a Paso
3. Registro de Resultados en la Tabla
4. Análisis de Resultados y Conclusión

**3.2**

I

El código representa un sistema interactivo de gestión de presupuesto

Se basa en un menú de opciones que permite al usuario administrar presupuestos para diferentes regiones

**3.2**

II

Mediante del uso de diferentes estructuras como lo son: el input, el print, el if-elif-else, etc. El usuario puede interactuar con el siguiente menú y poder realizar su respectiva accion

**¿Qué logramos?**

(1) Ingresar presupuesto inicial: Permite establecer valores de presupuesto para cada región.

(2) Mostrar presupuesto actual: Imprime los valores de las regiones almacenados en las variables.

(3) Modificar presupuesto de una región: Permite cambiar el presupuesto de una región específica.

(4) Redistribuir presupuesto equitativamente: Suma los presupuestos y los reparte en partes iguales.

(5) Simular campaña de marketing: Agrega un incremento fijo a una región elegida.

(6) Clasificar presupuesto: Evalúa el presupuesto total y lo clasifica en Bajo, Equilibrado o Alto.

(7) Salir: Termina la ejecución del programa.