Introducción a la Programación/ CONTROL No.3/ Valor: 2.5%/ 2025-10

Instructivos de Entrega:

- Individual
- Entrega en Canvas, cargar archivo .py
- Cumplir periodos de entrega

Hacer un programa en Python con las siguientes características:

Un vendedor desea calcular su comisión sobre la venta de un artículo. Al vendedor le corresponde el 2% de comisión por los artículos cuyo precio sea menor de $30000, el 4% de comisión por los artículos cuyo precio sea mayor o igual a $30000 y menor a $50000, un 5% de comisión por los artículos cuyo precio sea mayor o igual a %50000.

• Debe crear una clase de nombre “Ventas”, con los atributos: nombre_vendedor, precio_articulo, monto_comision.

• Debe crear un objeto de nombre “comision” con su correspondiente método constructor (__init__) que inicialice los valores de sus atributos: nombre_vendedor, precio_articulo, monto_comision=0. El atributo monto_comision debe ser asignado luego del calculo del mismo.

• Debe visualizar el estado del objeto creado con el método (__str__).

El programa debe tener:

Un menú con las siguientes opciones:

1.- Ingreso de datos y creación del objeto

2.- Calcular comisión

3.- Mostrar datos (estado) del objeto

4.- Salir

Acciones para realizar por cada opción:

• Al seleccionar la opción 1, se deben ingresar los datos: nombre_vendedor, precio_articulo. Además, se debe crear el objeto.

• Al seleccionar la opción 2, se debe calcular la comisión y asignar el valor al atributo monto_comision.

• Al seleccionar la opción 3, se debe mostrar el estado del objeto, es decir, el valor de cada uno de sus atributos.

• Al seleccionar la opción 4, el programa debe finalizar su ejecución.

Rúbrica de evaluación

Criterios de la entrega de la solución (.py)	Excelente (10 puntos)	Bueno (7 puntos)	Regular (4 puntos)	Deficiente (1 punto)
Puntualidad	Entrega a tiempo	Entrega con 1-2 días de atraso	-	-
Diseño del Algoritmo	El algoritmo está claramente definido, es eficiente y sigue una lógica óptima para resolver el problema. Se consideran todos los casos posibles y se anticipan errores.	El algoritmo es funcional y resuelve el problema, pero podría ser más eficiente o claro en su lógica. Se consideran la mayoría de los casos posibles.	El algoritmo tiene deficiencias en su lógica o no cubre todos los casos posibles. La eficiencia es baja y la claridad es limitada.	El algoritmo es confuso, incompleto o no resuelve el problema planteado. No se consideran los casos posibles.
Diseño de la solución de los requisitos	La solución implementa todos los requisitos del problema de forma	La solución implementa la mayoría de los	La solución implementa solo algunos de los	La solución no aborda el problema planteado o lo hace de forma

	creativa y eficiente. Se demuestra un profundo entendimiento del contexto y la problemática.	requisitos del problema de forma funcional. Se demuestra un buen entendimiento del contexto.	requisitos del problema o lo hace de forma incompleta. El entendimiento del contexto es limitado.	incorrecta. No se demuestra entendimiento del contexto.
Correcto Uso de Estructuras de Datos en Python	El código utiliza de manera eficiente y apropiada las estructuras de datos de Python (listas, strings, sentencias de control, etc.). Se justifica la elección de cada estructura y se optimiza su uso.	El código utiliza las estructuras de datos de Python de manera funcional, pero podría ser más eficiente o justificar mejor su elección.	El código utiliza las estructuras de datos de Python de manera incorrecta o ineficiente. No se justifica su elección o se utilizan estructuras inadecuadas para el problema.	El código no utiliza o utiliza de manera muy deficiente las estructuras de datos de Python.
Correcto uso de la sintaxis	El código utiliza correctamente la sintaxis de Python, incluyendo estructuras de control, funciones, listas y manejo de strings. No hay errores de sintaxis.	El código utiliza la sintaxis de Python de forma mayormente correcta, pero puede haber errores menores o inconsistencias.	El código presenta errores de sintaxis frecuentes que dificultan su ejecución. El uso de las estructuras de Python es limitado o incorrecto.	El código tiene numerosos errores de sintaxis que impiden su ejecución. El uso de Python es muy deficiente.
Correcta Ejecución	El programa se ejecuta sin errores y produce los resultados esperados en todos los casos de prueba. La salida es clara y fácil de entender.	El programa se ejecuta con algunos errores menores o produce resultados incorrectos en casos específicos. La salida e	El programa presenta errores que impiden su correcta ejecución o produce resultados incorrectos en la mayoría de los casos. La salida es confusa o incompleta.	El programa no se ejecuta o produce resultados completamente erróneos. La salida es inexistente o incomprensible.
Documentación de la solución	Correcta aplicación de los comentarios para documentación del código presentado	Aplicación de los comentarios en mediana escala para documentación del código presentado	Aplicación de los comentarios regularmente para documentación del código presentado	Sin documentación el código presentado