PROGRAMA INGENIERÍA DE SISTEMAS
PROGRAMACIÓN ORIENTADA A OBJETOS
POO – MVC
Ejercicio Calculadora

Trabajo Grupal

<Apellidos y Nombres> <ID>
<Apellidos y Nombres> <ID>
<Apellidos y Nombres> <ID>

ESCENARIO

Programación Orientada a Objetos (POO)
La Programación Orientada a Objetos es un paradigma que permite estructurar el software en torno a entidades llamadas objetos, los cuales combinan datos (atributos) y comportamientos (métodos). Este enfoque facilita la reutilización de código, la modularidad y la escalabilidad de las aplicaciones. En el desarrollo académico del programa de Ingeniería de Sistemas, se ha promovido el uso de POO como base para la construcción de soluciones robustas, utilizando lenguajes como Java y Python, y aplicando principios como encapsulamiento, herencia, polimorfismo y abstracción.

- Secuencia refina escenarios concretos y valida la colaboración entre clases.
- Objetos muestran ejemplos reales de instancias y datos en un momento dado.

Modelo Vista Controlador (MVC)
El Modelo Vista Controlador (MVC) es una arquitectura de software que separa la lógica de negocio, la interfaz de usuario y el control de flujo en tres componentes independientes: Modelo, Vista y Controlador. Esta separación permite desarrollar aplicaciones más organizadas, mantenibles y escalables. En el contexto del programa de Ingeniería de Sistemas, se ha trabajado con el patrón MVC en el desarrollo de aplicaciones web y de escritorio, destacando su utilidad para estructurar proyectos en frameworks como Django, Laravel y Spring, y fomentando buenas prácticas en el diseño de software.

En el desarrollo de aplicaciones con Python, el uso de diagramas UML (Lenguaje Unificado de Modelado) permite representar de manera estructurada los componentes del sistema, facilitando la planificación y el diseño orientado a objetos. A través de diagramas de clases, de secuencia y de casos de uso, se puede visualizar la interacción entre objetos, sus atributos y métodos, lo cual es esencial para implementar soluciones bajo el paradigma de Programación Orientada a Objetos. Esta metodología se complementa eficazmente con el patrón arquitectónico Modelo Vista Controlador (MVC), que organiza el código en tres capas independientes: el modelo gestiona la

lógica y los datos, la vista presenta la información al usuario, y el controlador coordina la interacción entre ambos. En conjunto, UML, MVC y POO permiten desarrollar aplicaciones modulares, escalables y mantenibles, promoviendo buenas prácticas en el diseño de software dentro del programa de Ingeniería de Sistemas.

DISEÑAR

Se presenta una construcción de una calculadora que no se ha terminado aplicando MVC y POO con lenguaje de Programación Python, es necesario implementar los siguientes cambios siguiendo el código entregado:

1. Realizar reingeniería y crear el diseño en UML de la aplicación que se entrega (Calculadora) con las funciones o métodos requeridos.
2. Programar las siguientes funciones por las series, recordar lo visto en clase sobre las funciones a utilizar.

Programar la función e^x, converge para toda x	$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$
La función Coseno(x)	$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} \quad \text{para toda } x$
La función Arcotangente(x)	$\operatorname{arctanh} x = \sum_{n=0}^{\infty} \frac{x^{2n+1}}{2n+1} \quad \text{para } \lvert x\rvert < 1$
La función Seno(x)	$\operatorname{sen} x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} \quad \text{para toda } x$
la fórmula de Leibniz sirve para el cálculo de π	$\sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1}$

Fuente: https://es.wikipedia.org/wiki/Serie_de_Taylor

GITHUB

Hacer uso de la cuenta de GitHub Académico, para almacenar sus entregables como también utilizar las diferentes herramientas de IA que ofrece al estudiante.

CUESTIONES EN LA ENTREGA

1. Portada

2. Introducción
3. Índice
4. Desarrollo
   4.1.Diagrama de clases
   4.2.Explicación de las funciones realizadas.
5. Referencias bibliográficas. Utilizar libros o las bases de datos de la biblioteca de UNIMINUTO (genera puntos según su utilización).

CALIFICACIÓN

Se aplica el siguiente instrumento de calificación únicamente a los trabajos entregados en aula virtual en el apartado y en los tiempos estipulados.

La calificación resulta de dos momentos:

1. El entregable en grupo que se califica de 1.0 a 5.0.
2. La defensa la podrá realizar aquellos que hayan realizado la entrega, y se calificará de 0,0 a 5,0.
3. Ejemplo: Entrega perfecta calificación 5,0.
   Defensa perfecta calificación 1,0
   Calificación final del trabajo = E*D => 5,0 * 1,0 = 5,0

RUBRICAS DE ENTREGA					
Entrega	Incumplimiento en lo solicitado en un 90% falta de correspondencia	Cumple con 30% de lo solicitado en la entrega, de forma como de diseño.	Cumple con un 50% de lo solicitado en la entrega, de forma y diseño	Cumple con 70% de Diseño y la forma en la entrega tiene o no falencias.	Cumple con todo lo solicitado, puede tener faltas de forma.
Calificación	0,0 a 1,0	1,1 a 2,0	2,1 a 3,0	3,1 a 4,0	4,1 a 5,0

RUBRICAS DE DEFENSA				
Entrega	Se realizan hasta tres preguntas y las respuestas no tienen contexto.	Responde dejando dudas de fondo se realiza hasta tres preguntas.	La respuesta es objetiva, pero deja espacio para realizar otra.	La respuesta es contundente y no deja dudas, asiste a tutorías.
Calificación	0,0 a 0,2	0,3 a 0,5	0,6 a 0,8	0,9 a 1,0

Ing. Alonso Guevara Pérez, Profesor