Unidad 1: Buenas Prácticas de Programación

Ejercicios en Clase

Ejercicio 1: Mejora de Nombres de Variables y Funciones

Instrucciones:

1. Se te proporcionará una lista de variables y funciones mal nombradas.
2. Mejora los nombres para que sean más descriptivos y sigan las buenas prácticas.
3. Asegúrate de no usar nombres genéricos o los nombres ya utilizados en el material de clase.

Lista de variables:

- n1, r, calcTotal(), tempData, x, y, procesar(), res()

numeroProductos, totalImpuestos, calcularTotalVentaSinImpuestos(), precioProducto,

Ejercicio 2: Nomenclatura y Formatos

Instrucciones:

1. A continuación, se te da una lista de nombres de variables y funciones.
2. Debes convertir cada uno de ellos a snake_case, camelCase, y kebab-case.

Lista inicial:

- **total ventas**: total_ventas; totalVentas; total-ventas

- **calcular precio descuento**: calcular_precio_descuento; calcularPrecioDescuento; calcular-precio-descuento

- **costoEnvio:** costo_envio; costoEnvio; costo-envio

Ejercicio 3: Investigación de Roles de Programación

Instrucciones:

1. Investiga tres roles diferentes en el desarrollo de software, como:

- Backend Developer: El **Backend Developer** es responsable de **desarrollar y mantener la lógica del servidor, las bases de datos y los servicios** que permiten el funcionamiento de una aplicación. Entre sus principales responsabilidades se encuentran el diseño de la arquitectura backend, la implementación de la lógica de negocio, la creación de APIs y la gestión segura y eficiente de los datos.

Su contribución al ciclo de desarrollo de software es clave en todas las etapas: participa en la planificación y diseño del sistema, desarrolla las funcionalidades durante la implementación, garantiza la calidad mediante pruebas y corrección de errores, y colabora en el despliegue y mantenimiento del software. Gracias a su trabajo, el sistema es estable, escalable y seguro.

- Data Scientist: El **Data Scientist** es responsable de **analizar grandes volúmenes de datos** para extraer información útil que apoye la toma de decisiones. Sus principales responsabilidades

incluyen la recolección, limpieza y análisis de datos, el desarrollo de modelos estadísticos y de aprendizaje automático, y la interpretación de resultados mediante visualizaciones y reportes.

Su contribución al ciclo de desarrollo de software se centra en las fases de análisis y diseño, al definir métricas y modelos, y en la implementación, al integrar modelos predictivos en los sistemas. Además, participa en la evaluación y mejora continua del producto, ayudando a optimizar funcionalidades y a orientar el desarrollo basándose en datos.

- QA Engineer: **El QA Engineer** es responsable de **asegurar la calidad del software** mediante la planificación, ejecución y automatización de pruebas. Sus principales responsabilidades incluyen la definición de casos de prueba, la detección y reporte de errores, la validación de requisitos y la verificación del correcto funcionamiento del sistema.

Su contribución al ciclo de desarrollo de software es fundamental en las fases de pruebas y validación, aunque también participa desde el diseño, ayudando a prevenir errores tempranos. Gracias a su trabajo, se garantiza que el producto final sea estable, confiable y cumpla con los estándares de calidad antes y después del despliegue.

2.

3. Comparte tus respuestas con tu grupo.

Ejercicio 4: Pseudocódigo para Calcular el Área de una Figura

Instrucciones:

1. Escribe un pseudocódigo para calcular el área de una figura geométrica que elijas

(círculo, triángulo, rectángulo, etc.).

2. Utiliza funciones modulares con responsabilidades claras y nombres descriptivos.
3. Añade comentarios donde expliques por qué tomas ciertas decisiones.

Ejemplo:

funcion calcular_area_circulo(radio):

area = 3.1416 * radio * radio

retornar area

mostrar("El área del círculo es: " + calcular_area_circulo(5))

función calcularAreaCuadrado(lado)

área = lado * lado

retornar área

mostrar(“El área del cuadrado es: “ + calcular_area_cuadrado(2)

Ejercicio 5: Mejorar Fragmento de Código

Instrucciones:

1. Se te proporciona el siguiente pseudocódigo con problemas de legibilidad y sin modularidad.

2. Tu tarea es mejorarlo aplicando modularización y usando nombres claros y descriptivos.

Pseudocódigo inicial:

```
total = 0
para cada i en lista:
total += i
si total > 100
mostrar("Total alto")
si_no
mostrar("Total bajo")

//Inicializamos el total
Total = 0
Función calcularTotal(num)
para cada num en lista:
//le añadimos el numero (num) al total
total +=numero
//si el total en mayor de 100 diremos que es alto y en el caso contrario es bajo
si total > 100
mostrar("Total alto")
si_no
mostrar("Total bajo")
```

Ejercicios para Tarea en Casa

Tarea 1: Renombrar Variables usando Diferentes Nomenclaturas

Instrucciones:

1. A continuación se te dan los nombres de variables en kebab-case.

2. Convierte cada uno de ellos a snake_case y camelCase.

Lista de nombres:

- **precio-total:** precio_total; precioTotal

- **costo-envio-extra:** costo_envio_extra; costoEnvioExtra

- **impuesto-añadido:** impuesto_añadido; impuestoAñadido

Tarea 2: Investigación Profunda de Roles en Programación

Instrucciones:

1. Elige un rol en el desarrollo de software (e.g., Backend Developer, QA Engineer).

2. Investiga más a fondo ese rol y prepara un resumen de una página con los siguientes apartados:

- Responsabilidades principales

- Habilidades técnicas necesarias

- Cómo contribuye este rol al proceso de desarrollo

Backend Developer

El Backend Developer es un rol fundamental dentro del desarrollo de software moderno, encargado de diseñar, implementar y mantener la lógica del lado del servidor que sostiene el funcionamiento de aplicaciones web, móviles y sistemas distribuidos. Su trabajo se centra en la gestión eficiente de datos, la comunicación entre servicios y la garantía de que las operaciones internas del sistema sean seguras, escalables y de alto rendimiento. Aunque su labor no es visible directamente para el usuario final, constituye la base sobre la cual se construye la experiencia del producto digital.

Responsabilidades principales

Las responsabilidades de un Backend Developer abarcan el análisis de requisitos técnicos y funcionales para definir la arquitectura del sistema. Esto incluye el diseño de estructuras de datos, la definición de modelos de dominio y la implementación de lógica de negocio compleja. Asimismo, desarrolla y mantiene APIs RESTful o GraphQL, asegurando una comunicación eficiente, consistente y segura entre el frontend, servicios externos y otros microservicios.

Otra responsabilidad clave es la gestión de bases de datos, tanto relacionales como no relacionales. El Backend Developer diseña esquemas, optimiza consultas, implementa transacciones y garantiza la integridad y consistencia de los datos. Además, se encarga de la implementación de mecanismos de autenticación y autorización, como OAuth 2.0, JWT o sistemas de control de acceso basados en roles (RBAC).

El mantenimiento del sistema también forma parte de sus funciones, incluyendo la detección y corrección de errores, la refactorización de código, la mejora del rendimiento y la aplicación de buenas prácticas de desarrollo como pruebas unitarias, pruebas de integración y revisión de código. En entornos modernos, el Backend Developer participa en procesos de CI/CD, automatizando despliegues y asegurando la estabilidad del software en producción.

Habilidades técnicas necesarias

Desde el punto de vista técnico, un Backend Developer debe dominar uno o varios lenguajes de programación del lado del servidor, como Java (Spring Boot), Python (Django, Flask, FastAPI), Node.js (Express, NestJS), C# (.NET) o Go. Es esencial comprender los principios de programación orientada a objetos, programación funcional, y patrones de diseño como MVC, Repository o Singleton.

Debe poseer conocimientos avanzados en bases de datos relacionales (PostgreSQL, MySQL, SQL Server) y no relacionales (MongoDB, Redis, Cassandra), así como en técnicas de optimización, indexación y modelado de datos. También es fundamental entender arquitecturas modernas como microservicios, arquitecturas orientadas a eventos y sistemas distribuidos.

El manejo de herramientas de control de versiones como Git, la experiencia con contenedores (Docker) y plataformas de orquestación como Kubernetes, junto con conocimientos en servicios en la nube (AWS, Azure o Google Cloud), son habilidades altamente valoradas. Asimismo, se requiere un sólido entendimiento de seguridad informática, incluyendo cifrado, protección contra ataques comunes (SQL Injection, XSS, CSRF) y gestión de secretos.

Contribución al proceso de desarrollo

El Backend Developer desempeña un papel crítico en todas las fases del ciclo de vida del software. Durante la etapa de diseño, contribuye a la definición de la arquitectura técnica y a la toma de decisiones que afectan la escalabilidad y mantenibilidad del sistema. En la fase de desarrollo, implementa la lógica de negocio que permite transformar los datos en funcionalidades útiles para el usuario final.

Su colaboración constante con desarrolladores frontend, DevOps, analistas y arquitectos de software asegura que los distintos componentes del sistema se integren de forma coherente y eficiente. Además, al garantizar la estabilidad, seguridad y rendimiento del backend, el Backend Developer contribuye directamente a la calidad del producto, la experiencia del usuario y la capacidad del sistema para crecer y adaptarse a nuevas necesidades del negocio.

Tarea 3: Algoritmo para Gestionar una Lista de Tareas

Instrucciones:

1. Escribe un pseudocódigo para gestionar una lista de tareas pendientes.

2. Debes incluir las siguientes funcionalidades:

- Añadir una tarea

- Marcar una tarea como completada

- Eliminar una tarea

3. Usa nombres descriptivos y modulariza las funcionalidades en funciones.

tarea = “”

estadoTarea = false

función EstadoTareas(tarea, estadoTarea)

si estadoTarea = false

mostrar(“tarea pendiente”)

si_no

mostrar(“tarea finalizada”

Tarea 4: Documentación sobre Buenas Prácticas

Instrucciones:

1. Escribe un documento de máximo dos páginas sobre buenas prácticas en

programación.

2. Incluye:

- La importancia de usar una nomenclatura descriptiva.

- Modularidad.

- Uso adecuado de comentarios.

Tarea 5: Identificación de Problemas en Pseudocódigo

Instrucciones:

1. A continuación se te proporciona un pseudocódigo lleno de malas prácticas.

2. Identifica al menos cinco problemas en el pseudocódigo y describe cómo los mejorarías.

Pseudocódigo:

a = 10

```
b = a * 2

print(a)
```

1. Cambiaría el nombre de la variable a por algo más descriptivo
2. Cambiaría el nombre de la variable b por algo más descriptivo
3. Añadiría una función dónde se calcularía b (con el nombre cambiado)
4. Si queremos que varíe el valor a multiplicar, crearía una variable para ese valor
5. Mostraría b en lugar de a, porque es lo que queremos saber