Portafolio digital

Alumno: Jhonal Roca
Profesor: Fran Gomez
1º DAW - 25/26

Este portafolio de programación tiene como objetivo recopilar y mostrar lo aprendido a lo largo del curso. Al finalizar, servirá como un resumen práctico-teórico de los temas más importantes que vimos en clase, de manera que sea fácil de entender y consultar.

La idea es que no solo sea útil para mí como repaso, sino que también cualquier persona que nunca haya tenido contacto con la programación pueda leerlo, comprender de qué trata la asignatura y animarse a intentarlo. La programación puede parecer complicada al inicio, pero aquí se muestra de forma sencilla y accesible, para demostrar que cualquiera con curiosidad y ganas puede aprenderla.

Act 1. Comparativa de Lenguajes de Programación

Hoy en día la informática es uno de los recursos que más se está utilizando debido a el avanzado desarrollo del internet en todas las cosas, a día de hoy el internet está tan incorporado en nuestras vidas que dependemos del día a día, por eso hoy veremos los distintos tipos de lenguajes de programación teniendo en cuenta los siguientes puntos.

- Qué tan fáciles son de aprender
- Qué tantas oportunidades de trabajo tienen
- En qué áreas se pueden usar
- Qué tan rápidos y eficientes son
- Qué tan grande es la comunidad que los respalda

Lenguaje	Facilidad para aprender	Demanda de trabajo	Donde se usa	Velocidad	Comunidad y recursos
Python	Muy fácil y parece inglés	Alta, (datos, IA, web, automatizaciones)	Super versátil	Medio	Enorme, miles de cursos y foros
JavaScript	Fácil -Medio	Altísima (Web sobre todo)	Web, apps móviles, servidores	Buena	gigante, casi imprescindible
Java	UN poco más complicado de aprender	Muy alta(Empresas grandes, android)	Empresas, apps, sistemas	Muy buena	Muy estable y con soporte
C++	Difícil (Muchos detalles técnicos)	Alta (Videojuegos y sistemas)	Juegos, software de alto rendimiento	Rápida	Comunidad tecnica grande
C#	Medio	Alta (Apps y videojuegos con	Videojuegos, apps de	Muy Buena	Fuerte y en crecimiento

		unity)	escritorio y backend		
PHP	Fácil pero algo anticuado	Aun alta (Muchas webs usan wordPress)	Principalmente webs	Regular	Grande, aunque menos moderna
Rust	Difícil	Va en crecimiento	Seguridad, sistemas críticos	Muy buena	Comunidad más técnica
Go (Golang)	Bastante fácil	Cada vez más demandado	Servidores, microservicios, nube	Muy rápido	Comunidad más pequeña pero activa

Ahora toca la pregunta de muchas personas que quieren empezar en el mundo de la programación y es ¿Con qué lenguaje empiezo a programar?.

- Yo personalmente empresaria a programar con Python ya que es uno de los más fáciles y a día de hoy muchas empresas están empezando a trabajar con él, actualmente es un lenguaje sencillo de aprender y está abriendo muchas puertas en el mercado.

- Línea Cronológica de los lenguajes de programación

Act 2. programación, algoritmos y Códigos.

1. ¿En qué consiste la programación?

La programación es el proceso de darle instrucciones a una computadora para que realice tareas específicas. Estas instrucciones se escriben en un lenguaje de programación (como Python, Java o JavaScript) que la máquina pueda entender y ejecutar.

En pocas palabras, programar es decirle paso a paso a una computadora qué hacer, desde algo tan simple como sumar dos números hasta algo tan complejo como manejar una aplicación de inteligencia artificial.

Diferencia entre algoritmos y código

- **Algoritmos**
    - Son como las recetas o los pasos lógicos que se deben seguir para resolver un problema.

- Están escritos de manera general, sin importar el lenguaje de programación.
- Ejemplo: “1. Encender la cafetera, 2. Colocar el café, 3. Agregar agua, 4. Servir”.

- **Código**
  - Es la traducción de este algoritmo a un lenguaje que la computadora pueda entender.
  - Se escribe con la sintaxis y reglas propias de cada lenguaje de programación.
  - Ejemplo en Python para sumar dos números:

```
# Definir los dos números
numero1 = 5
número 2 = 3
# Sumar los dos números
suma = numero1 + numero2
# Imprimir el resultado
print("La suma es:", suma)
```

Act 3. Clasificación de los lenguajes

Lenguaje	Propósito	Tipo	Uso principal	Plataforma	Ventajas	Desventajas
Python	General	Interpretado	Ciencia de datos, ML, desarrollo web	Web, ciencia de datos, ML	Fácil de aprender, gran comunidad	Velocidad de ejecución
Java	General	Compilado	Aplicaciones empresariales, Android	Android, Desktop, Server	Portabilidad (JVM), fuerte tipado	Verbosidad, mayor consumo de memoria
JavaScript	Web	Interpretado	Desarrollo web (front-end, back-end)	Navegadores, Node.js	Versatilidad (front-end y back-end), popularidad	Problemas con rendimiento en grandes apps
C#	General	Compilado	Desarrollo de aplicaciones de escritorio, videojuegos	Windows, .NET Framework	Integración con Windows, robusto para apps grandes	Requiere entorno .NET, menos portable
C	Sistemas,	Compilado	Desarrollo de	Multiplataforma	Eficiencia, control	Difícil de

	general		sistemas embebidos, sistemas operativos		total sobre hardware	aprender, manejo manual de memoria
C++	Sistemas, general	Compilado	Videojuegos, aplicaciones de alto rendimiento	Multiplataforma	Alta performance, control sobre recursos	Complejidad, manejo manual de memoria
PHP	Web	Interpretado	Desarrollo web dinámico	Servidores web (LAMP stack)	Rápido para desarrollo web, gran ecosistema	Malas prácticas comunes, no adecuado para apps grandes
Go	General	Compilado	Backend, microservicios, sistemas concurrentes	Multiplataforma	Alto rendimiento, concurrencia fácil de usar	Bibliotecas aún en desarrollo, menor ecosistema
Rust	Sistemas, general	Compilado	Sistemas, aplicaciones de alto rendimiento	Multiplataforma	Seguridad en memoria, alto rendimiento	Curva de aprendizaje, menos maduro que C/C++
Ruby	General	Interpretado	Desarrollo web (Rails), scripts	Web, servidores	Sintaxis sencilla, productividad alta	Menor rendimiento, escaso en áreas fuera de la web
Swift	General	Compilado	Desarrollo de apps iOS/macOS	iOS, macOS	Alto rendimiento, fácil de aprender, seguro	Solo para productos Apple, comunidad pequeña
Bash/Batch/PowerShell	Scripting	Interpretado	Automatización, administración de sistemas	Linux/Unix, Windows	Rápido para scripting, automatización	Limitado a administración de sistemas, no es adecuado para apps complejas

Paradigmas de programación

- **Imperativa** → Describe cómo se debe realizar cada paso de un proceso. El código es una secuencia de instrucciones detalladas.
- **Interpretado** → El código se ejecuta directamente por un intérprete, línea por línea, sin necesidad de compilación previa.

- **Declarativa** → Enfocado en qué resultado se quiere obtener, dejando que el sistema decida cómo lograrlo.

- **Estructurada** → Organiza el flujo de control en tres estructuras básicas: secuencia, selección e iteración. Facilita la legibilidad y el control del programa:
    - Secuencial - paso a paso.
    - Selección - Conlleva a opciones de pasos para finalizar.
    - Iteración - Repite los pasos hasta obtener lo deseado.

- **Modular** → Divide el código en módulos o bloques independientes que realizan tareas específicas, promoviendo la reutilización y la mantenibilidad.

Act. 4 Algoritmos y diagramas de flujo

1. Escribe un algoritmo y dibuja el diagrama para calcular el área del círculo.
2. Debes solicitar al usuario datos de entrada y mostrar el resultado cuando finalice.

Algoritmo para calcular el área de un círculo

- Iniciar

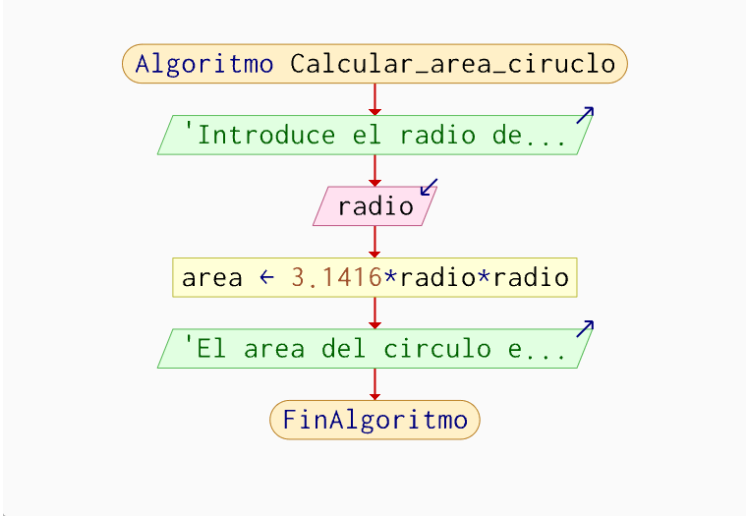
- Mostrar en pantalla: "Introduce el radio del círculo:"

- Leer el número ingresado por el usuario y guardarlo en la variable radio

- Calcular el área usando la fórmula:
- area = 3.1416 * radio * radio

- Mostrar en pantalla: "El área del círculo es: " seguido del valor calculado en area

- Terminar



Act 5. Algoritmos y diagramas de flujo

1. Diagrama y algoritmo que calcule la nota media de tres actividades.
2. Usa los paréntesis para modificar la precedencia y así realizar correctamente el cálculo.

Algoritmo para calcular la nota media

- Inicio
- Mostrar por pantalla: Ingrese notas:
- El alumno ingresa tres notas (una por cada actividad).
- Se calcula la media usando paréntesis para asegurarse de que la suma se haga antes de la división.
- Mostrar por pantalla: Resultado
- Fin