Aplicando lo aprendido 1.

Nombre y apellido: Maia Belén Bulacio.

Ejercicio 1

Parte A

1. Generalización simbólica: ¿Cuáles son las reglas escritas del lenguaje?

En JavaScript, las reglas escritas del lenguaje se refieren a su sintaxis, semántica y estructuras gramaticales, las cuales están definidas formalmente por el estándar **ECMAScript (ECMA-262)** que especifica cómo debe comportarse JavaScript.

-Sensibilidad a mayúsculas y minúsculas.

-Uso de **;** al terminar una sentencia y **{}** para definir bloques.

-Tipos de datos primitivos (string, number, boolean, etc.)

-Estructuras de control (if, for, while)

-Funciones, clases, objetos

-Reglas de declaración de variables (var, let, const)

Gramática: La especificación ECMA proporciona una gramática formal (BNF) para el análisis sintáctico.

2. Creencias de los profesionales: ¿Qué características particulares del lenguaje se cree que sean "mejores" que en otros lenguajes?

Lo que se considera mejor de JavaScript en comparación con otros lenguajes es la facilidad de aprendizaje, la versatilidad, el dinamismo que permite cambios en el momento de ejecución, el desarrollo rápido que permite construir funcionalidades de manera ágil, la capacidad de correr en multiplataformas (navegadores, servidores, móviles, etc); además tiene una gran cantidad de herramientas de desarrollo modernas, librerías y frameworks.

Parte B

1. ¿Tiene una sintaxis y una semántica bien definida? ¿Existe documentación oficial?
   Si, la sintaxis esta definida por el estándar ECMA-262. Además, existe una página donde está la documentación oficial que es la confiable para la comunidad de desarrolladores MDN Web Docs (Mozilla Developer Network).

2. ¿Es posible comprobar el código producido en ese lenguaje?

   Si, se pueden usar herramientas básicas como la consola del navegador y herramientas de análisis estático como ESLint, TypeScript (para tipado) y linters. Sin embargo, debido a que es dinámico y no fuertemente tipado, no siempre es posible detectar todos los errores en tiempo de compilación.

3. ¿Es confiable?

Su confiabilidad depende de las metodologías y usos del programador ya que, al ser un lenguaje tan flexible, es posible cometer errores sutiles pero que impidan la correcta ejecución del programa.

4. ¿Es ortogonal?

   En JavaScript hay cierta flexibilidad, pero también inconsistencias como null y undefined que se comportan distinto en algunas operaciones. Además, algunas estructuras no son intercambiables (por ejemplo, for...in vs for...of).

5. ¿Cuáles son sus características de consistencia y uniformidad?

   La consistencia en JavaScript se refiere a mantener la uniformidad y la coherencia en la forma en que se escribe y estructura el código. Como por ejemplo la modularidad: el escribir código en módulos ayuda a mantener la consistencia al encapsular la funcionalidad y organizar el código en partes reutilizables.
   Las características de uniformidad de JavaScript se basan en que gran parte de sus elementos comparten un mismo modelo. Sin embargo, su uniformidad no es absoluta porque existen excepciones y comportamientos especiales, como el tratamiento distinto de null y undefined o la coerción de tipos, que rompen esa homogeneidad.

6. ¿Es extensible? ¿Hay subconjuntos de ese lenguaje?

   Sí, es extensible, se pueden crear librerías, frameworks (React, Angular, etc.), y prototipos personalizados. El lenguaje permite agregar propiedades y métodos a objetos en tiempo de ejecución.

   Algunos subconjuntos son: ES5, ES6, ESNext, Strict Mode (un subconjunto más restringido y seguro de JavaScript). También existe TypeScript que es un superconjunto de JavaScript.

7. El código producido, ¿es transportable?

   Sí, es transportable porque JavaScript corre en cualquier navegador moderno y en entornos como Node.js sin depender del sistema operativo. Pero tiene algunas diferencias mínimas de implementación entre navegadores.