### 1. Generalización simbólica

TypeScript ejecuta las instrucciones de forma secuencial y permite controlar el flujo mediante estructuras clásicas como if, else, switch, for, while y do while.  
Las variables se declaran con let y const, manteniendo el mismo comportamiento que en JavaScript, pero con **tipos de datos explícitos**.  
Los bloques de código se delimitan con llaves {} y se organizan en funciones usando la palabra reservada function.  
Además, el lenguaje obliga a respetar reglas de tipado que definen el tipo de cada dato, como number, string, boolean, entre otros, lo que mejora la claridad y previene errores.

### 2. Creencias de los profesionales

La comunidad considera que TypeScript es una evolución natural de JavaScript, ya que combina su flexibilidad con un sistema de **tipado estático** que aporta seguridad y confiabilidad al código.  
Se valora especialmente que permite detectar errores antes de ejecutar el programa, reduciendo fallos en producción.  
También se cree que su integración directa con entornos modernos y su compatibilidad total con JavaScript lo convierten en una herramienta muy práctica para proyectos grandes.  
Otra creencia extendida es que el uso de tipos explícitos y estructuras claras fomenta mejores prácticas de programación y mejora la mantenibilidad del código a largo plazo.

## Parte B

### 1. Sintaxis y semántica

TypeScript comparte la sintaxis básica de JavaScript, pero agrega reglas semánticas más estrictas mediante el uso de tipos.  
Su definición oficial se basa en el estándar ECMAScript, con extensiones propias documentadas por Microsoft.  
La documentación en línea, como la de TypeScriptlang.org o MDN, sirve de referencia práctica para los desarrolladores que buscan escribir código más seguro y estructurado.

### 2. Comprobación del código

En TypeScript, el código se **verifica antes de ejecutarse** gracias al compilador tsc, que analiza el tipo de datos y las declaraciones.  
Esto permite detectar errores en tiempo de compilación, antes de que el programa se convierta a JavaScript y se ejecute en el navegador o en Node.js.  
Por esta razón, TypeScript ofrece una comprobación más robusta que JavaScript, sin perder la posibilidad de ejecutar el código de forma rápida una vez compilado.

### 3. Confiabilidad

TypeScript es considerado un lenguaje confiable porque combina la estabilidad del estándar ECMAScript con el control de tipos y herramientas modernas.  
El compilador detecta inconsistencias o errores de tipo antes de ejecutar el programa, lo que ayuda a evitar errores lógicos.  
Además, la amplia comunidad y el soporte oficial de Microsoft aseguran su mantenimiento y evolución constante.

### 4. Ortogonalidad

TypeScript presenta mayor ortogonalidad que JavaScript, ya que las reglas de tipado reducen combinaciones ambiguas o inconsistentes.  
Aun así, conserva cierta flexibilidad del lenguaje original, permitiendo usar any o inferencia de tipos cuando el programador lo necesita.  
Este equilibrio entre control y libertad le da potencia sin sacrificar seguridad.

### 5. Consistencia y uniformidad

TypeScript mantiene la estructura general de JavaScript, pero añade una sintaxis más consistente gracias al tipado obligatorio.  
Por ejemplo, las funciones deben especificar el tipo de retorno y de sus parámetros, lo que unifica el estilo de programación y evita ambigüedades.  
A diferencia de JavaScript, las herramientas del compilador imponen un formato más uniforme en la forma de declarar variables y funciones.

### 6. Extensibilidad y subconjuntos

TypeScript es altamente extensible: se integra fácilmente con librerías y frameworks modernos como Angular, React o Node.js.  
Además, puede configurarse para aceptar distintos niveles de restricción mediante el archivo tsconfig.json, lo que permite adaptar el lenguaje a las necesidades del proyecto.  
En cuanto a subconjuntos, cada versión del compilador añade nuevas características opcionales, y su código final puede limitarse a distintas versiones de ECMAScript para garantizar compatibilidad.

### 7. Portabilidad

El código TypeScript es muy portátil, ya que se compila a JavaScript estándar, lo que permite ejecutarlo en cualquier navegador moderno o entorno que soporte ECMAScript.  
Gracias a esto, un mismo programa puede funcionar en diferentes sistemas operativos sin modificaciones, manteniendo las ventajas del tipado estático y del paradigma estructurado.