I'm not a robot

Javascript practice exercises for beginners

Type systems can be classified as either strongly typed or weakly typed, with the latter being colloquially referred to as loosely typed. However, there is no universally accepted definition, leading some authors to avoid using these terms altogether in favor of more specific expressions like "type safety". A strongly typed language typically has stricter typing rules at compile time, resulting in fewer errors during compilation. In contrast, weakly typed languages have looser typing rules and may produce unpredictable results or perform implicit type conversion at runtime. Different classes of programming errors exist, leading some languages to implement "strong typing" disciplines to address these issues. Some languages make it possible to use a value of one type as if it were another, which is known as "weak typing." This can lead to coercion, where the language's type system is bypassed, or conversion, where a new object is created. For instance, the C programming language uses void* to allow for coercion. However, this can lead to issues with aliasing and optimization. Other languages, such as C++ and C#, allow for explicit type conversions through well-defined operators. Some languages also expose pointers as if they were numeric values, allowing users to perform arithmetic on them. This is known as "weakly typed." In contrast, languages like Java and Pascal require variables to have a declared type and support explicit casts only for arithmetic types. The concept of strong typing refers to the absence of unchecked runtime type errors. This property is also referred to as safety or type safety. Some definitions are contradictory, while others are conceptually independent or special cases of more "liberal" (less strict) definitions. Typed languages control methods of evading static type system in Java through virtual machine's type system, similar to C# and VB.NET but with an explicit disabling option. Pascal's strong type system includes array and string size as its type, making it difficult for programming tasks. However, Delphi resolves this issue. Languages such as Smalltalk, Ruby, Python, and Self have "strong" typing without runtime checks or implicit type conversion, while others like the Lisp family and Standard ML are statically checked with type declarations.