DYOL
Design a Language
Know the Consequences

Garbage Collection
Automatic release of memory is impossible for cyclic data structures. Languages that want to support them, have a garbage collector – a runtime compiler component that occasionally marks data structures that have become inaccessible and then sweeps them away, freeing the memory. GC can compromise language responsiveness and performance.

Concrete Syntax
The way to describe the concrete representation of the programs. The concrete syntax is used by humans to read, write, create and understand sentences of the language. Usually the only languages that do not have concrete syntax are those intended for internal intermediate representation. Some languages have more than one.

Alphabet
The basic alphabet is often taken for granted, especially for textual languages, but it is an important design aspect. In some languages (APL being the extreme) the alphabet is extremely broad, with specific symbols being used for built-in operators, which shifts the visual feel of the language closer to mathematics. In other languages keywords are taken from English, which limits language appeal to some groups of users (and may lead to reimplementations with translated keywords).

Operator Precedence
To avoid excessive use of parentheses, a language can provide a default convention of disambiguating constructs with 3+ entities bound by binary operators. In arithmetic expressions, the precedence usually follows mathematical laws.

Type Analysis
Components can be identified, explicitly or automatically, to belong to a particular type. Among other things, the type determines applicability and compatibility of components with one another. In complex scenarios (like a monadic bind) how to understand components can only fit together in one possible way. Type equivalence rules can be based on names, structure, scopes, etc.

Synonyms and similar terms

- Type System
- A complementary set of techniques to type analysis, the main difference in the definition of type constructions, bottom up in syntax and top down in analysis. Types
- The sort of rules combining all the types available in a software language, into one system with subtypes, conversions, etc.

Full deck at http://slebok.github.io/dyol
Vadim Zaytsev aka @grammarware