Appendix A - Keywords
The following list contains keywords that are reserved for current or future use by the Cairo language.
There are three keyword categories:
- strict
- loose
- reserved

There is a fourth category, which are functions from the core library. While their names are not reserved,
they are not recommended to be used as names of any items to follow good practices.

Strict keywords
These keywords can only be used in their correct contexts.
They cannot be used as names of any items.
- `as` - Rename import
- `break` - Exit a loop immediately
- `const` - Define constant items
- `continue` - Continue to the next loop iteration
- `else` - Fallback for `if` and `if let` control flow constructs
- `enum` - Define an enumeration
- `extern` - Function defined at the compiler level that can be compiled to CASM
- `false` - Boolean false literal
- `fn` - Define a function
- `if` - Branch based on the result of a conditional expression
- `impl` - Implement inherent or trait functionality
- `implicits` - Special kind of function parameters that are required to perform certain actions
- `let` - Bind a variable
- `loop` - Loop unconditionally
- `match` - Match a value to patterns
- `mod` - Define a module
- `mut` - Denote variable mutability
- `nopanic` - Functions marked with this notation mean that the function will never panic.
- `of` - Implement a trait
- `pub` - Denote public visibility in items, such as struct and struct fields, enums, consts, traits and impl blocks, or modules
- `ref` - Parameter passed implicitly returned at the end of a function
- `return` - Return from function
- `struct` - Define a structure
- `trait` - Define a trait
- `true` - Boolean true literal
- `type` - Define a type alias
- `use` - Bring symbols into scope
- `while` - loop conditionally based on the result of an expression

---
Loose Keywords

These keywords are associated with a specific behaviour, but can also be used to define items.
- `self` - Method subject
- `super` - Parent module of the current module

Reserved Keywords
These keywords aren't used yet, but they are reserved for future use.
For now, it is possible to use them to define items, although it is highly recommended not to do so.
The reasoning behind this recommendation is to make current programs forward compatible with future versions of
Cairo by forbidding them to use these keywords.
- `Self`
- `do`
- `dyn`
- `for`
- `hint`
- `in`
- `macro`
- `move`
- `static_assert`
- `static`
- `try`
- `typeof`
- `unsafe`
- `where`
- `with`
- `yield`

Built-in Functions
The Cairo programming language provides several specific functions that serve a special purpose. We will not cover all of them in this book, but using the names of these functions as names of other items is not recommended.
- `assert` - This function checks a boolean expression, and if it evaluates to false, it triggers the panic function.
- `panic` - This function acknowledges the occurrence of an error and terminates the program.