1 Values

1.1 Literals

Literals are expressions that are hard-coded into the code. They take one of four forms.

1.2 Numeric Literals

These must start with a digit, a plus sign, a minus sign, or a period.

1.3 Boolean Literals

Either `true` or `false`.

1.4 Character and String Literals

These must start and end with a single quote `'`. What is in between is interpreted as a string. If a character cannot be typed, it can be inputted as `'\uXXXX'` where `XXXX` represents the 4 digit unicode designation of the desired character. There are three distinct cases:

1. `''`: This is automatically a string literal representing an empty string.
2. A single character: depending on the context, this is interpreted as a string or character.
3. Multiple characters: always a string.

1.5 Examples

- `2`, `-56543234565`, `41`, `-.02345654321`, `12.`

2 Statements

There are a limited number of valid statement forms. All start with a capital letter and end with a period.

2.1 Definition

2.1.1 Declaration

A minimal declaration simply provides a variable with a name and associates it with a type.

```
Define a[n] <type> called <name>.
```

This is equivalent to the Java `<type> <name>;`

2.1.2 Field Initialization

A variable can also have its fields initialized, including the field `value`, which represents the value of the entire structure.

```
Define a[n] <type> called <name> with a[n] <field1> of <value1>,
a[n] <field2> of <value2>, and a[n] <field3> of <value3>.
```

Commas and **and** are all technically unnecessary, but included to insure readability. Similarly, **a** and **an** are equivalent but both are included to avoid statements like `Define a integer called x.`

2.1.3 Examples

```
Define an integer called x.

Define a string called name with a value of ‘‘41++’’.

Define a matrix called M with a width of 3 and a height of
2.  Define a matrix called M2 with a value of M.
```