11

Interpreters

Exercise 11.1. Draw the parse tree for each of the following Python expressions and provide the value of each expression.

a. $1 + 2 + 3 * 4$

Solution.

b. $3 > 2 + 2$

Solution.

c. $3 * 6 >= 15 == 12$

Solution.

d. $(3 * 6 >= 15) == \mathsf{True}$

Solution.

Exercise 11.2. Do comparison expressions have higher or lower precedence than addition expressions? Explain why using the grammar rules.

Solution.

Exercise 11.3. Define the sequence of factorials as an infinite list using delayed evaluation.

Solution.

Exercise 11.4. Describe the infinite list defined by each of the following definitions. (Check your answers by evaluating the expressions in LazyCharme.)

a. (**define** *p* (*cons* 1 (*merge-lists* *p* *p* +)))

Solution.

b. (**define** *t* (*cons* 1 (*merge-lists* *t* (*merge-lists* *t* *t* +) +)))

Solution.

c. (**define** *twos* (*cons* 2 *twos*))

Solution.

d. (**define** *doubles* (*merge-lists* (*ints-from* 1) *twos* *))

Solution.

Eratosthenes

Exercise 11.5. [⋆⋆] A simple procedure known as the *Sieve of Eratosthenes* for finding prime numbers was created by Eratosthenes, an ancient Greek mathematician and astronomer. The procedure imagines starting with an (infinite) list of all the integers starting from 2. Then, it repeats the following two steps forever:

1. Circle the first number that is not crossed off; it is prime.
2. Cross off all numbers that are multiples of the circled number.

To carry out the procedure in practice, of course, the initial list of numbers must be finite, otherwise it would take forever to cross off all the multiples of 2. But, with delayed evaluation, we can implement the Sieve procedure on an effectively infinite list.

Implement the sieve procedure using lists with lazy evaluation. You may find the *list-filter* and *merge-lists* procedures useful, but will probably find it necessary to define some additional procedures.

Solution.