Computer Systems Modeling and Verification
(USEEN1)

Working with lists

Exercises taken from the *Python Programming Primer*[1] (and updated with type hints).

Textbook *Introduction to Python for Computational Science and Engineering* (2022), Chapters 3-4.

Corresponding lecture slides from *Computational Science and Engineering in Python*:

- Sequences
- Loops
- Some things revisited

Note. Interfaces (called *abstract base classes* in Python) should be used to provide types to arguments instead of concrete implementations. For instance, use `Sequence[A]` instead of `list[A]` whenever the argument is not modified by the function, and `MutableSequence[A]` otherwise.

Exercises. Define the following functions.

1. A function `min_max(xs: Sequence[int]) -> tuple[int, int]` that computes the minimum value `xmin` of the elements in the list xs, and the maximum value `xmax` of the elements in the list, and returns a tuple `(xmin, xmax)`.

   Example:

   ```
   >>> min_max([0, 1, 2, 10, -5, 3])
   (-5, 10)
   ```

2. A function `range_squared(n: int) -> list[int]` that takes an non-negative integer value `n` and that returns the list `[0, 1, 4, 9, 16, 25, ..., (n-1)²]`. If `n` is zero, the function should return the empty list.

   Example:

   ```
   >>> range_squared(3)
   [0, 1, 4]
   ```

3. A function `count(element: int, seq: Sequence[int]) -> int` that counts how often the given element `element` occurs in the given sequence `seq`, and returns this integer value. For example, `count(2, list(range(5)))` should return `1`.

   Change the type of elements to `str`, and test again with the following example:

   ```
   >>> count('dog', ['dog', 'cat', 'mouse', 'dog'])
   2
   ```

4. A function `seq_sqrt(xs: Sequence[int]) -> list[float]` which takes a list of non-negative numbers `xs` with elements `[x0, x1, x2, ..., xn]`, and returns the list `[sqrt(x0), sqrt(x1), sqrt(x2), ..., sqrt(xn)]`. In other words, the function takes a list of numbers, and returns a list of the same length that contains the square root for each number in the list.

5. A function `mean(xs: Sequence[int]) -> float` that takes a sequence `xs` of numbers, and returns the (arithmetic) mean (i.e. the average value):

1. *Python Programming Primer*, Hans Fangohr *et al.* University of Southampton (2016)

Example:

```
>>> mean([0, 1, 2])
1.0
```

6. A function `reversed(xs: Sequence[int]) -> list[int]` that takes a sequence `xs` of integer values, computes the reverse of this sequence and returns it in the form of a list.

7. A function `concat(xs1: Sequence[int], xs2: Sequence[int]) -> list[int]` that takes two sequences of integer values and returns the concatenation of these sequences in the form of a list (you are not allowed to use the concatenation operator `+` in this exercise).