6.4 Combinatoric functions

The `itertools.combinations(iterable, r)` returns an iterator giving all possible *r*-tuple combinations of the elements contained in *iterable*.

```
itertools.combinations([1, 2, 3, 4, 5], 2) =>
  (1, 2), (1, 3), (1, 4), (1, 5),
  (2, 3), (2, 4), (2, 5),
  (3, 4), (3, 5),
  (4, 5)

itertools.combinations([1, 2, 3, 4, 5], 3) =>
  (1, 2, 3), (1, 2, 4), (1, 2, 5), (1, 3, 4), (1, 3, 5), (1, 4, 5),
  (2, 3, 4), (2, 3, 5), (2, 4, 5),
  (3, 4, 5)
```

The elements within each tuple remain in the same order as *iterable* returned them. For example, the number 1 is always before 2, 3, 4, or 5 in the examples above. A similar function, `itertools.permutations(iterable, r=None)`, removes this constraint on the order, returning all possible arrangements of length *r*:

```
itertools.permutations([1, 2, 3, 4, 5], 2) =>
  (1, 2), (1, 3), (1, 4), (1, 5),
  (2, 1), (2, 3), (2, 4), (2, 5),
  (3, 1), (3, 2), (3, 4), (3, 5),
  (4, 1), (4, 2), (4, 3), (4, 5),
  (5, 1), (5, 2), (5, 3), (5, 4)

itertools.permutations([1, 2, 3, 4, 5]) =>
  (1, 2, 3, 4, 5), (1, 2, 3, 5, 4), (1, 2, 4, 3, 5),
  ...
  (5, 4, 3, 2, 1)
```

If you don't supply a value for *r* the length of the iterable is used, meaning that all the elements are permuted.

Note that these functions produce all of the possible combinations by position and don't require that the contents of *iterable* are unique:

```
itertools.permutations('aba', 3) =>
  ('a', 'b', 'a'), ('a', 'a', 'b'), ('b', 'a', 'a'),
  ('b', 'a', 'a'), ('a', 'a', 'b'), ('a', 'b', 'a')
```

The identical tuple `('a', 'a', 'b')` occurs twice, but the two 'a' strings came from different positions.

The `itertools.combinations_with_replacement(iterable, r)` function relaxes a different constraint: elements can be repeated within a single tuple. Conceptually an element is selected for the first position of each tuple and then is replaced before the second element is selected.

```
itertools.combinations_with_replacement([1, 2, 3, 4, 5], 2) =>
  (1, 1), (1, 2), (1, 3), (1, 4), (1, 5),
  (2, 2), (2, 3), (2, 4), (2, 5),
  (3, 3), (3, 4), (3, 5),
  (4, 4), (4, 5),
  (5, 5)
```