Short Python function/method descriptions:

`__builtins__`:
- `float(x) -> float`
  Convert a string or number to a floating point number, if possible.
- `int(x) -> int`
  Convert x to an integer, if possible. A floating point argument will be truncated towards zero.
- `len(x) -> int`
  Return the length of list, tuple, or string x.
- `print(value) -> NoneType`
  Print the value.
- `range([start], stop, [step]) -> list-like-object of int`
  Return the integers starting with start and ending with stop - 1 with step specifying the amount to increment (or decrement). If start is not specified, the sequence starts at 0. If step is not specified, the values are incremented by 1.
- `str(x) -> str`
  Return an object converted to its string representation, if possible.

`str`:
- `x in s -> bool`
  Produce True if and only if x is in string s.
- `S.count(sub[, start[, end]]) -> int`
  Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.
- `S.find(sub[, i]) -> int`
  Return the lowest index in S (starting at S[i], if i is given) where the string sub is found, or -1 if sub does not occur in S.
- `S.isalpha() -> bool`
  Return True if and only if all characters in S are alphabetic and there is at least one character in S.
- `S.isalnum() -> bool`
  Return True if and only if all characters in S are alphanumeric and there is at least one character in S.
- `S.isdigit() -> bool`
  Return True if and only if all characters in S are digits and there is at least one character in S.
- `S.islower() -> bool`
  Return True if and only if all cased characters in S are lowercase and there is at least one cased character in S.
- `S.isupper() -> bool`
  Return True if and only if all cased characters in S are uppercase and there is at least one cased character in S.
- `S.lower() -> str`
  Return a copy of the S converted to lowercase.
- `S.upper() -> str`
  Return a copy of S converted to uppercase.

`list`:
- `x in L -> bool`
  Produce True if and only if x is in list L
- `L.append(object) -> NoneType`
  Append object to end of list L.
- `L.extend(iterable) -> NoneType`
  Extend list L by appending elements from the iterable. Strings and lists are iterables whose elements are characters and list items respectively.
- `L.insert(index, object) -> NoneType`
  Insert object into list L at index.