# Appendix C: String Operators and Methods 10

This documentation was generated from the Python documentation available by typing *help*(*str*) in the Python shell. In the documentation found here the variables *s* and *t* are references to strings (Table 10.1).

**Table 10.1** String operators and methods

| Operator | Returns | Comments |
|---|---|---|
| s+t | str | Return a new string which is the concatenation of s and t |
| s in t | bool | Returns True if s is a substring of t and False otherwise |
| s==t | bool | Returns True if s and t refer to strings with the same sequence of characters |
| s>=t | bool | Returns True if s is lexicographically greater than or equal to t |
| s<=t | bool | Returns True if s is lexicographically less than or equal to t |
| s>t | bool | Returns True if s is lexicographically greater than t |
| s<t | bool | Returns True if s is lexicographically less than t |
| s != t | bool | Returns True if s is lexicographically not equal to t |
| s[i] | str | Returns the character at index i in the string. If i is negative then it returns the character at index len(s)−i |
| s[[i]:[j]] | str | Returns the slice of characters starting at index i and extending to index j−1 in the string. If i is omitted then the slice begins at index 0. If j is omitted then the slice extends to the end of the list. If i is negative then it returns the slice starting at index len(s)+i (and likewise for the slice ending at j) |
| s ∗ i | str | Returns a new string with s repeated i times |
| i ∗ s | str | Returns a new string with s repeated i times |
| chr(i) | str | Return the ASCII character equivalent of the integer i |
| float(s) | float | Returns the float contained in the string s |
| int(s) | int | Returns the integer contained in the string s |
| len(s) | int | Returns the number of characters in s |
| ord(s) | int | Returns the ASCII decimal equivalent of the single character string s |

(continued)

K.D. Lee, *Python Programming Fundamentals*,
Undergraduate Topics in Computer Science, DOI 10.1007/978-1-4471-6642-9_10

**Table 10.1** (continued)

| Method | Returns | Comments |
|---|---|---|
| repr(s) | | Returns a string representation of s. This adds an extra pair of quotes to s |
| str(s) | str | Returns a string representation of s. In this case you get just the string s |
| s.capitalize() | str | Returns a copy of the string s with the first character upper case |
| s.center(width[, fillchar]) | str | Returns s centered in a string of length width. Padding is done using the specified fill character (default is a space) |
| s.count(sub[, start[, end]]) | int | Returns 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.encode([encoding[, errors]]) | bytes | Encodes s using the codec registered for encoding. encoding defaults to the default encoding. errors may be given to set a different error handling scheme. Default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register_error that can handle UnicodeEncodeErrors |
| s.endswith(suffix[, start[, end]]) | bool | Returns True if s ends with the specified suffix, False otherwise. With optional start, test s beginning at that position. With optional end, stop comparing s at that position. suffix can also be a tuple of strings to try |
| s.expandtabs([tabsize]) | str | Returns a copy of s where all tab characters are expanded using spaces. If tabsize is not given, a tab size of 8 characters is assumed |
| s.find(sub[, start[, end]]) | int | Returns the lowest index in s where substring sub is found, such that sub is contained within s[start:end]. Optional arguments start and end are interpreted as in slice notation. Return −1 on failure |
| s.format(*args, **kwargs) | str | |
| s.index(sub[, start[, end]]) | int | Like s.find() but raise ValueError when the substring is not found |
| s.isalnum() | bool | Returns True if all characters in s are alphanumeric and there is at least one character in s, False otherwise |
| s.isalpha() | bool | Returns True if all characters in s are alphabetic and there is at least one character in s, False otherwise |
| s.isdecimal() | bool | Returns True if there are only decimal characters in s, False otherwise |

(continued)

**Table 10.1** (continued)

| Method | Returns | Comments |
|---|---|---|
| s.isdigit() | bool | Returns True if all characters in s are digits and there is at least one character in s, False otherwise |
| s.isidentifier() | bool | Returns True if s is a valid identifier according to the language definition |
| s.islower() | bool | Returns True if all cased characters in s are lowercase and there is at least one cased character in s, False otherwise |
| s.isnumeric() | bool | Returns True if there are only numeric characters in s, False otherwise |
| s.isprintable() | bool | Returns True if all characters in s are considered printable in repr() or s is empty, False otherwise |
| s.isspace() | bool | Returns True if all characters in s are whitespace and there is at least one character in s, False otherwise |
| s.istitle() | bool | Returns True if s is a titlecased string and there is at least one character in s, i.e. upper- and titlecase characters may only follow uncased characters and lowercase characters only cased ones. Return False otherwise |
| s.isupper() | bool | Returns True if all cased characters in s are uppercase and there is at least one cased character in s, False otherwise |
| s.join(sequence) | str | Returns a string which is the concatenation of the strings in the sequence. The separator between elements is s |
| s.ljust(width[, fillchar]) | str | Returns s left-justified in a Unicode string of length width. Padding is done using the specified fill character (default is a space) |
| s.lower() | str | Returns a copy of the string s converted to lowercase |
| s.lstrip([chars]) | str | Returns a copy of the string s with leading whitespace removed. If chars is given and not None, remove characters in chars instead |
| s.partition(sep) | (h,sep,t) | Searches for the separator sep in s, and returns the part before it, the separator itself, and the part after it. If the separator is not found, returns s and two empty strings |
| s.replace (old, new[, count]) | str | Returns a copy of s with all occurrences of substring old replaced by new. If the optional argument count is given, only the first count occurrences are replaced |
| s.rfind(sub[, start[, end]]) | int | Returns the highest index in s where substring sub is found, such that sub is contained within s[start:end]. Optional arguments start and end are interpreted as in slice notation.<br>Returns −1 on failure |
| s.rindex(sub[, start[, end]]) | int | Like s.rfind() but raise ValueError when the substring is not found |

(continued)

**Table 10.1** (continued)

| Method | Returns | Comments |
|---|---|---|
| s.rjust(width[, fillchar]) | str | Returns s right-justified in a string of length width. Padding is done using the specified fill character (default is a space) |
| s.rpartition(sep) | (t,sep,h) | Searches for the separator sep in s, starting at the end of s, and returns the part before it, the separator itself, and the part after it. If the separator is not found, returns two empty strings and s |
| s.rsplit([sep[, maxsplit]]) | string list | Returns a list of the words in s, using sep as the delimiter string, starting at the end of the string and working to the front. If maxsplit is given, at most maxsplit splits are done. If sep is not specified, any whitespace string is a separator |
| s.rstrip([chars]) | str | Returns a copy of the string s with trailing whitespace removed. If chars is given and not None, removes characters in chars instead |
| s.split([sep[, maxsplit]]) | string list | Returns a list of the words in s, using sep as the delimiter string. If maxsplit is given, at most maxsplit splits are done. If sep is not specified or is None, any whitespace string is a separator and empty strings are removed from the result |
| s.splitlines([keepends]) | string list | Returns a list of the lines in s, breaking at line boundaries. Line breaks are not included in the resulting list unless keepends is given and true |
| s.startswith (prefix[, start[, end]]) | bool | Returns True if s starts with the specified prefix, False otherwise. With optional start, test s beginning at that position. With optional end, stop comparing s at that position. Prefix can also be a tuple of strings to try |
| s.strip([chars]) | str | Returns a copy of the string s with leading and trailing whitespace removed. If chars is given and not None, removes characters in chars instead. |
| s.swapcase() | str | Returns a copy of s with uppercase characters converted to lowercase and vice versa |
| s.title() | str | Returns a titlecased version of s, i.e. words start with title case characters, all remaining cased characters have lower case |
| s.translate(table) | str | Returns a copy of the string s, where all characters have been mapped through the given translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None. Unmapped characters are left untouched. Characters mapped to None are deleted |
| s.upper() | str | Returns a copy of s converted to uppercase |
| s.zfill(width) | str | Pad a numeric string s with zeros on the left, to fill a field of the specified width. The string s is never truncated |