

Strings never change in place: every method returns a new string, so keep the result (`s = s.strip()`). Examples use `s = "python"` .

CREATE <code>"hi" == 'hi'</code> → <code>True</code> single or double quotes, same string <code>"""Dear Ada, Thanks."""</code> triple quotes keep the line breaks <code>r"C:\new\data"</code> raw: backslashes stay as typed <code>"ab" * 3</code> → <code>'ababab'</code> <code>"py" + "thon"</code> → <code>'python'</code> <code>str(42)</code> → <code>'42'</code> any value to text	CLEAN UP <code>" hi \n".strip()</code> → <code>'hi'</code> also <code>lstrip()</code> and <code>rstrip()</code> <code>"--hi--".strip("--")</code> → <code>'hi'</code> <code>"report.csv".removesuffix(".csv")</code> → <code>'report'</code> <code>"v1.2".removeprefix("v")</code> → <code>'1.2'</code> <code>"a-b-c".replace("-", " ")</code> → <code>'a b c'</code> <code>"a-b-c".replace("-", "", 1)</code> → <code>'ab-c'</code> only the first 1	ESCAPES <code>len("\a\nb")</code> → <code>3</code> <code>\n</code> is one character: a new line <code>"tab\there"</code> <code>\t</code> tab, <code>\\</code> a backslash <code>"It's"</code> → <code>"It's"</code> or escape it: <code>It\'s</code> <code>"caf\u00e9"</code> → <code>'café'</code> <code>\u</code> + 4 hex digits
INDEX AND SLICE <code>s = "python"</code> <code>s[0]</code> → <code>'p'</code> first <code>s[-1]</code> → <code>'n'</code> last <code>s[1:4]</code> → <code>'yth'</code> start included, stop excluded <code>s[:3]</code> → <code>'pyt'</code> <code>s[3:]</code> → <code>'hon'</code> <code>s[::2]</code> → <code>'pto'</code> every second character <code>s[::-1]</code> → <code>'nohtyp'</code> reversed <code>len(s)</code> → <code>6</code>	SPLIT AND JOIN <code>"a,b,c".split(",")</code> → <code>['a', 'b', 'c']</code> no argument: any run of whitespace <code>"a b\tc".split()</code> → <code>['a', 'b', 'c']</code> no argument: any run of whitespace <code>"k=v=w".split("=", 1)</code> → <code>['k', 'v=w']</code> split once <code>"a b c".rsplit(" ", 1)</code> → <code>['a b', 'c']</code> split once, from the right <code>"one\ntwo".splitlines()</code> → <code>['one', 'two']</code> <code>", ".join(["a", "b"])</code> → <code>'a, b'</code> the separator comes first <code>"-".join(str(n) for n in [1, 2])</code> → <code>'1-2'</code> join needs strings <code>"key=val".partition("=")</code> → <code>('key', '=', 'val')</code>	LOOP OVER TEXT <code>s = "python"</code> <code>for ch in s:</code> <code>print(ch)</code> one character at a time <code>[i for i, ch in enumerate(s) if ch == "o"]</code> → <code>[4]</code> positions of a character <code>"".join(sorted("cab"))</code> → <code>'abc'</code> <code>sorted(set("banana"))</code> → <code>['a', 'b', 'n']</code> the distinct characters <code>max(["fig", "apple"], key=len)</code> → <code>'apple'</code> longest word <code>sorted(["b", "A", "c"], key=str.lower)</code> → <code>['A', 'b', 'c']</code> sort ignoring case
SEARCH AND TEST <code>s = "python"</code> <code>"th" in s</code> → <code>True</code> <code>s.find("o")</code> → <code>4</code> index of the first match <code>s.find("z")</code> → <code>-1</code> <code>-1</code> when it's not there <code>s.index("z")</code> → <code>raises ValueError</code> like find, but raises <code>"banana".count("a")</code> → <code>3</code> <code>s.startswith("py")</code> → <code>True</code> <code>"a.csv".endswith((".csv", ".tsv"))</code> → <code>True</code> a tuple tests several <code>"2026".isdigit()</code> → <code>True</code> <code>"a1".isalnum()</code> → <code>True</code> <code>"\t".isspace()</code> → <code>True</code>	F-STRINGS <code>name, pi, big, rate = "Ada", 3.14159, 1234567, 0.256</code> <code>f"Hi {name}!"</code> → <code>'Hi Ada!'</code> <code>f"{pi:.2f}"</code> → <code>'3.14'</code> 2 decimal places <code>f"{big:,}"</code> → <code>'1,234,567'</code> thousands separator <code>f"{rate:.1%}"</code> → <code>'25.6%'</code> <code>f"{7:03d}"</code> → <code>'007'</code> pad with zeros <code>f"{name:>6}"</code> → <code>' Ada'</code> right-align in 6; < left, ^ centre <code>f"{name:^7}"</code> → <code>'**Ada**'</code> centre, fill with * <code>f"{pi:.1f}"</code> → <code>'pi=3.1'</code> = prints the name too: handy for debugging <code>f"{255:x}"</code> → <code>'ff'</code> hex; b for binary	HANDY IN THE STANDARD LIBRARY <code>import string, textwrap</code> <code>textwrap.shorten("A long title here", 12)</code> → <code>'A long [...]'</code> <code>import textwrap</code> <code>string.ascii_lowercase[:5]</code> → <code>'abcde'</code> <code>import string</code>; also digits, punctuation <code>" ".join(w.capitalize() for w in "ada lovelace".split())</code> → <code>'Ada Lovelace'</code> each word capitalised, rest lowered
CASE <code>"hi there".title()</code> → <code>'Hi There'</code> <code>"hi there".capitalize()</code> → <code>'Hi there'</code> <code>"HI".upper()</code> → <code>'HI'</code> <code>"Hi".lower()</code> → <code>'hi'</code> <code>"Straße".casefold()</code> → <code>'strasse'</code> for comparing text, not for display	CONVERT <code>int("42")</code> → <code>42</code> <code>float("3.5")</code> → <code>3.5</code> <code>list("abc")</code> → <code>['a', 'b', 'c']</code> <code>ord("A")</code> → <code>65</code> <code>chr(97)</code> → <code>'a'</code> <code>"café".encode()</code> → <code>b'caf\xca3\xa9'</code> str to bytes, UTF-8 <code>b"hi".decode()</code> → <code>'hi'</code>	WATCH OUT <code>s = "python"</code> <code>s[0] = "P"</code> <code>TypeError: strings can't change. Build a new one: "P" + s[1:]</code> <code>s.upper()</code> returns a new string and s stays the same; write <code>s = s.upper()</code> <code>"o'neil's".title()</code> → <code>"O'Neil'S"</code> <code>title()</code> capitalises after any non-letter <code>"10" < "9"</code> → <code>True</code> text compares character by character: <code>int()</code> first <code>"www.site.com".strip("w.")</code> strip removes characters, not a word: use <code>removeprefix("www.")</code> <code>"Total: " + 5</code> <code>TypeError: use f"Total: {5}"</code> <code>text += piece</code> <code># in a long loop</code> slow: collect the pieces in a list, then <code>"".join(pieces)</code>