Strings
Comparing strings

To compare two strings:

• Compare the first characters of each string against each other using their ASCII values.
  • If the first character of \( s_1 \) < first character of \( s_2 \), then \( s_1 < s_2 \).
  • If the first character of \( s_2 \) < first character of \( s_1 \), then \( s_2 < s_1 \).
  • Otherwise, the characters must be equal. Repeat with the next character.

• If you run out of characters at the same time, the two strings are obviously equal.

• If you run out of characters in one string before the other, then shorter string < longer string. (Just like “me” comes before “met” in the dictionary.)
The **for** statement

The general form of a **for** statement is:

```python
for <var> in <some kind of series>:
```

The easiest way to explain this is with an example:

```python
for i in [1, 2, 3]:
    print(i)
```

Produces the output:

1
2
3

- The thing in [ ] is called a *list*
- The number of times you go through the loop = the number of items in the list
- Each time you go through the loop, you assign the value of the next item to the variable in the **for** statement
- Don’t forget the colon
- Indentation is important!
There are a number of useful methods (functions) that apply to strings.

`isalpha()` returns True if there is at least one character in this string and all characters are alphabetic.

`isdigit()` returns True if this string contains only number characters.

`islower()` returns True if there is at least one character in this string and all characters are lowercase.

`isupper()` returns True if there is at least one character in this string and all characters are uppercase.

`isspace()` returns True if this string only contains whitespace characters.
String Editing Methods

upper() return a copy of the string with all letters converted to uppercase

lower() return a copy of the string with all letters converted to lowercase

capitalize() return a copy of the string with the first character uppercase and the rest lowercase

title() return a copy of the string with the first letter of each word capitalized

swapcase() return a copy of the string in which all lowercase letters are made uppercase, and vice versa
replace(old, new) replace all instances of old in string with new

lstrip() remove all leading (left) whitespace from string
rstrip() remove all trailing (right) whitespace from string
strip() remove all leading and trailing whitespace from string

center(length) creates a new string of the given length
with the given string centered between spaces
ljust(length) creates a new string of the given length
consisting of the given string followed by spaces
rjust(length) creates a new string of the given length
consisting of spaces followed by the given string
Methods for Searching Substrings

endswith(s1) returns True if the string ends with the substring s1.

startswith(s1) returns True if the string starts with the substring s1.

find(s1) returns the lowest index where s1 starts in the string, or -1 if it is not found at all.

rfind(s1) returns the highest index where s1 starts in the string, or -1 if it is not found at all.

count(substring) returns the number of non-overlapping occurrences of the substring.
A string is a palindrome if it reads the same backwards and forwards. "mom", "dad", and "noon" are all palindromes.

Write a program that prompts the user to enter a string and reports whether or not the string is a palindrome. Ignore blanks at the start and end of the string.

Have the program check whether the first character is the same as the last character. If they are, move to the second character and the second to the last character. Continue until either:

• a mismatch is found
• all characters have been checked
• the string has an odd number of characters, and all characters except the middle character have been checked.