Python Review EXERCISES

Work with your neighbor.

1. What is the value of \( x \)?

\[
x = 15 \mod 4 - 1 * 8
\]

2. Given a variable \( x \), write an expression that evaluates to \text{True} \text{ if } \( x \) is even and \text{False} otherwise.

3. Given a variable \( x \), write an expression that evaluates to \text{True} \text{ if } \( x \) is even and divisible by 5 and \text{False} otherwise.

4. Assume that string \( s \) has length greater than 2. Show two ways to produce the second to the last character of \( s \).

5. Given the expression

\[
\text{text} = \text{\textasciitilde"Hello World!"}
\]

Write the result of the following expressions:

\[
\text{text}[2:5]
\]

\[
\text{text}[::3]
\]

\[
\text{text} * 2
\]
6. Write an expression that takes a string `text` and creates a new string where the first and last characters of `text` are swapped. Assume that `text` has a length of 2 or greater.

7. Suppose you are given a string variable `text` that consists of arbitrary characters and a single “−” character. An example would be the following:

   ```
   text = "ababab-ccc"
   ```

   Create a new string from `text` that swaps all the characters before and after the “−” and removes the hyphen. Given the definition of `text` above, the created string would be
   ```
   "cccababab"
   ```

   Assume that the hyphen is not the last character of `text`.

   Hint: consult the reference page for a string method that will help in solving this problem.

8. Given the expression

   ```
   text = "?!?!?!?!!?!!??!how are you?!?!!??!?!?!"
   ```

   Write an expression that operates on `text` and evaluates to
   ```
   "HOW ARE YOU"
   ```
Reference Sheet

Arithmetic expressions
Python supports the usual arithmetic operators such as +, -, *, %, /, and //. The arithmetic operators follow the usual PEMDAS rules of precedence.

Logical operators
Python supports the usual logical operators: and, or, and not.

Strings
In Python, strings are denoted by characters enclosed in single or double quotes. For example,

```python
s = "abcdefg"
text = 'Hello all!'
```

Strings are indexed (or subscripted) from position 0 and are indexed using brackets so that \texttt{s[i]} results in the \texttt{i}th element of string \texttt{s}.

Given the assignments above,

```python
s[3]
```

is the string 'd'.

The following table lists a few of the Python string operations.

<table>
<thead>
<tr>
<th>Operation or method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{s.find(substring)}</td>
<td>Returns the index where the start of the given substring appears in string \texttt{s} (Returns -1 if not found.)</td>
</tr>
<tr>
<td>\texttt{s[index1: index2]}</td>
<td>Returns the characters in string \texttt{s} from index1 (inclusive) to index2 (exclusive); if index2 is omitted, grabs till end of string</td>
</tr>
<tr>
<td>\texttt{s.lower()}, \texttt{s.upper()}</td>
<td>Returns a new string with characters in \texttt{s} converted to lowercase or uppercase letters</td>
</tr>
<tr>
<td>\texttt{s.startswith(substring)}</td>
<td>Returns True if \texttt{s} starts with string \texttt{substring} and False otherwise</td>
</tr>
<tr>
<td>\texttt{s.join(substring)}</td>
<td>Returns a string which is the concatenation of the individual single length substrings in \texttt{substring}.</td>
</tr>
<tr>
<td>\texttt{s1 + s2}</td>
<td>Concatenates strings \texttt{s1} and \texttt{s2}</td>
</tr>
<tr>
<td>\texttt{len(s)}</td>
<td>Returns the length of string \texttt{s}</td>
</tr>
</tbody>
</table>
Solutions

Problem 1
-5

Problem 2
x % 2 == 0

Problem 3
x % 2 == 0 and x % 5 == 0

Problem 4
text[-2]

text[len(text) - 2]

Problem 5
'llo'
'Hel'
'Hello World!Hello World!'

Problem 6
text[-1] + text[1:-1] + text[0]

Problem 7
i = text.find("-")
text[i+1:] + text[:i]

Problem 8
text.strip("?!").upper()