**Python Assignment 6**

1. What are escape characters, and how do you use them?

Escape characters are special characters in a string that are used to represent certain actions or characters that are difficult to represent directly. They are used to include characters that cannot be easily typed or have special meanings within a string, such as newline characters, tab characters, quotes, and more.

Escape characters are typically represented by a backslash \ followed by a character or a sequence of characters. Here are some common escape characters and their meanings:

\\: Backslash

\': Single quote

\": Double quote

\n: Newline (line break)

\t: Tab

\r: Carriage return (return to the beginning of the line)

\b: Backspace

\f: Form feed (page break)

2. What do the escape characters n and t stand for?

\n (Newline):

The escape sequence \n represents a newline character. When it is encountered in a string, it creates a new line, effectively moving the cursor to the beginning of the next line. This is commonly used to format text into multiple lines.

\t (Tab):

The escape sequence \t represents a tab character. It adds horizontal spacing to the text and is often used to align content in columns.

3. What is the way to include backslash characters in a string?

To include a backslash \ character in a string, we need to use the escape character \\. The double backslash \\ is used to represent a single literal backslash within the string.

Here's an example:

text = "This is a backslash: \\"

print(text)

# Output: This is a backslash: \

4. The string "Howl's Moving Castle" is a correct value. Why isn't the single quote character in the

word Howl's not escaped a problem?

In Python strings, we can use either single quotes ' ' or double quotes " " to define string literals. This flexibility allows to use one type of quote to enclose a string that contains the other type of quote without the need for escaping.

In the string "Howl's Moving Castle", the outer quotes are double quotes, and the inner single quote is not causing a problem because Python treats it as part of the string and does not interpret it as the end of the string.

Similarly, we could define the same string using single quotes as well:

title = 'Howl\'s Moving Castle'

print(title)

# Output: Howl's Moving Castle

5. How do you write a string of newlines if you don't want to use the n character?

If we want to write a string with newlines without using the \n escape character, we can achieve this by using triple-quoted strings (''' or """) in Python. Triple-quoted strings allow us to write multiline strings without the need for explicit escape characters.

6. What are the values of the given expressions?

'Hello, world!'[1]

'Hello, world!'[0:5]

'Hello, world!'[:5]

'Hello, world!'[3:]

'Hello, world!'[1] yields 'e'.

'Hello, world!'[0:5] and 'Hello, world!'[:5] both yield 'Hello'.

'Hello, world!'[3:] yields 'lo, world!'.

7. What are the values of the following expressions?

'Hello'.upper()

'Hello'.upper().isupper()

'Hello'.upper().lower()

'Hello'.upper() yields 'HELLO'.

'Hello'.upper().isupper() yields True.

'Hello'.upper().lower() yields 'hello'.

8. What are the values of the following expressions?

'Remember, remember, the fifth of July.'.split()

'-'.join('There can only one.'.split())

'Remember, remember, the fifth of July.'.split() yields ['Remember,', 'remember,', 'the', 'fifth', 'of', 'July.'].

'-'.join('There can only one.'.split()) yields 'There-can-only-one.'.

9. What are the methods for right-justifying, left-justifying, and centering a string?

Right-justifying: The str.rjust(width, fillchar) method is used to right-justify a string within a field of the given width. It pads the string with the specified fillchar (default is space) on the left side until the desired width is reached.

Example:

text = 'Hello'

justified = text.rjust(10)

print(justified)

# Output: ' Hello'

Left-justifying: The str.ljust(width, fillchar) method is used to left-justify a string within a field of the given width. It pads the string with the specified fillchar (default is space) on the right side until the desired width is reached.

Example:

text = 'Hello'

justified = text.ljust(10)

print(justified)

# Output: 'Hello '

Centering: The str.center(width, fillchar) method is used to center a string within a field of the given width. It pads the string with the specified fillchar (default is space) on both sides until the desired width is reached.

Example:

text = 'Hello'

centered = text.center(10)

print(centered)

# Output: ' Hello '

10. What is the best way to remove whitespace characters from the start or end?

To remove whitespace characters from the start or end of a string in Python, we can use the str.strip() method. The strip() method removes leading (start) and trailing (end) whitespace characters (including spaces, tabs, and newline characters) from a string. It returns a new string with the whitespace removed.