**Python Assignment 6**

**By Abhishek Sachan**

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

Escape characters are special characters that are used to represent certain characters or sequences of characters that cannot be typed directly in a string or text document. They are used to escape or modify the behavior of other characters in the text.

1. \n - Represents a new line character
2. \t - Represents a tab character
3. \r - Represents a carriage return character
4. ' - Represents a single quote character
5. " - Represents a double quote character

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

\n represent new line character

\t represent a tab character

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

If you want to include a backslash character () in a string, you need to use an escape character to tell Python to interpret the backslash as a literal character, rather than as the start of an escape sequence.

To include a backslash character in a Python string, you can use the escape character sequence "\". For example, to create a string that contains the word "Hello" followed by a backslash character, you could use the following code:

my\_string = "Hello\\"

In this code, the double backslash represents a literal backslash character. When the string is printed or displayed, it will appear as "Hello".

**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?**

The single quote character in the string "Howl's Moving Castle" is not a problem because the string is enclosed in double quotes, which is a valid way to create a string in Python. When a string is enclosed in double quotes, single quote characters can be used inside the string without being escaped, and vice versa.

This is because Python allows both single quotes and double quotes to be used interchangeably to create strings. This means that if a string is enclosed in double quotes, any single quotes within the string are treated as literal characters, and do not need to be escaped.

However, if you wanted to enclose the same string in single quotes, you would need to escape the single quote character in the word "Howl's" in order to avoid a syntax error. For example:

my\_string = 'Howl\'s Moving Castle'

In this case, the backslash character is used to escape the single quote character, which tells Python to interpret it as a literal character rather than the end of the string.

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

If you want to write a string of newlines in Python without using the "\n" character, you can use multi-line strings with the triple quotes syntax (either triple single quotes or triple double quotes). This syntax allows you to create a string that spans multiple lines without using any escape characters.

Here's an example of how you can create a string of newlines using the triple quotes syntax:

my\_string = '''This is a string  
that spans  
multiple lines'''

6. What are the values of the given expressions?

Expressions Output

‘Hello, world!’[1] it will be print e ‘Hello, world!’[0:5] it will be print Hello

‘Hello, world!’[:5] it will we be print Hello

‘Hello, world!’[3:] it will be print llo, world!

**7. What are the values of the following expressions?**

**Expression Output**

**‘Hello’.upper() HELLO**

**‘Hello’.upper().isupper() True**

**‘Hello’.upper().lower() 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() would return a list of strings, where each string is a word in the original sentence. The output would be: ['Remember,', 'remember,', 'the', 'fifth', 'of', 'July.']

"'-'.join('There can only one.'.split())" would split the string "There can only one." into individual words, then join them back together using a hyphen (-) as a separator. The output would be: "There-can-only-one."

Therefore, the final output of the two expressions would be:

* The first expression doesn't actually have a print statement, so it wouldn't output anything.
* The second expression would output the string "There-can-only-one."

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

* For right-justifying we can use text.rjust(width)
* For left justifying we can use text.ljust(width)
* For center a starting we can use text.center(width)

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

**strip()** method: This method removes whitespace characters from both the start and end of a string.

The **lstrip()** method removes whitespace characters from the start of a string,

while the **rstrip()** method removes whitespace characters from the end of a string.

Another way to remove whitespace characters from the start or end of a string is to use regular expressions. The **re** module in Python provides several functions to work with regular expressions. For example:

import re

text = " Hello, World! "

stripped\_text = re.sub(r'^\s+|\s+$', '', text)

print(stripped\_text)