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

* Escape characters represent characters in string values that would otherwise be difficult or impossible to type into code.

|  |  |
| --- | --- |
| **Code** | **Result** |
| \' | Single Quote |
| \\ | Backslash |
| \n | New Line |
| \r | Carriage Return |
| \t | Tab |
| \b | Backspace |
| \f | Form Feed |
| \ooo | Octal value |
| \xhh | Hex value |

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

* Print (“Hello\nWorld!!!”)
  + Hello
  + World!!!
* Print (“Hello\tWorld!!!”)
  + Hello World!!!

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

* Print ("This will insert one \\ (backslash).")
  + This will insert one \ (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?

* The single quote in Howl's is fine because you've used double quotes to mark the beginning and end of the string.

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

* Multiline strings allow you to use newlines in strings without the \n escape character.

6. What are the values of the given expressions?

* 'Hello, world!'[1] 🡪 “e”
* 'Hello, world!'[0:5] 🡪 “Hello”
* 'Hello, world!'[:5] 🡪 “Hello”
* 'Hello, world!'[3:] 🡪”lo, world!”

7. What are the values of the following expressions?

* '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()

🡪 ['Remember,', 'remember,', 'the', 'fifth', 'of', 'July.']

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

🡪 'There-can-only-one.'

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

* The rjust(), ljust(), and center() string methods, respectively

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

* The lstrip() and rstrip() methods remove whitespace from the left and right ends of a string, respectively.