Python Basic- Assignment- 06

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

Answer:

Escape characters are characters that have special meaning in Python. They are used to indicate when a character should be interpreted differently than it normally would. For example, the backslash (\) is an escape character that is used to indicate special characters such as newlines (\n) or tabs (\t). Escape characters can also be used to add special characters to a string, such as quotes (\"). To use an escape character in a string, we must prefix it with a backslash (\). For example:

#input:

```
my_string = "This is a string with a \"quote\" in it"

print(my_string)
```

#Output:

```
This is a string with a "quote" in it
```

The backslash (\) in front of the quote (\") tells python to interpret the quote as part of the string, rather than as the end of the string.

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

Answer:

The escape characters n and t stand for the newline (\n) and tab (\t) characters respectively. The newline character is used to denote the end of a line in a text file or the end of a command in a terminal. The tab character is used to create an indentation or a tabular format in a text document.

#input:

```
my_string = "My name is Mahmud. \nI am a student of full stack data science 2.0.\tI am enjoying it"

print(my_string)
```

#output:

```
My name is Mahmud.

I am a student of full stack data science 2.0.      I am enjoying it
```

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

Answer:

To include backslash characters in a string, you must use a double backslash (\\) or a raw string (r"string"). A double backslash will escape the backslash character, allowing it to be included in the string as a literal character. A raw string is a string with an ‘r’ prefix, which tells python to interpret backslash characters as literal characters instead of escape characters.

Example(\\):

#input:

```
my_string = "His name is Rahim \\ Karim"

print(my_string)
```

#output:

```
His name is Rahim \ Karim
```

Example(r"string"):

#input:

```
my_string1 = r"His name is Rahim\Karim"

print(my_string1)
```

#output:

```
His name is Rahim\Karim
```

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

Answer:

The single quote character does not need to be escaped in python because strings are surrounded by either single or double quotes, and the same type of quote is used to denote the beginning and end of the string. Therefore, since the single quote character is enclosed in double quotes, it is not treated as a special character and does not need to be escaped.

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

Answer:

We can use the triple quotation marks (''' or """) to write a multiple line string.

For example:

```
myString = """
This is
a string
of newlines
"""
print(myString)
```

This will print the following string:

```
This is

a string

of newlines
```

6. What are the values of the given expressions?

Answer:

'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?

Answer:

'Hello'.upper()

HELLO

'Hello'.upper().isupper()

True

'Hello'.upper().lower()

‘hello’

8. What are the values of the following expressions?

Answer:

'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 centering a string?

Answer:

Right-justifying a string means that the text is aligned to the right side of a container. This can be done using the rjust() method. This will take a string and a width as arguments and will return a right-justified string within the specified width.

#input:

```
a = "apple"
b = a.rjust(20)
print(b, "is a name of fruit.")
```

#output:

```
               apple is a name of fruit.
```

Left-justifying a string means that the text is aligned to the left side of a container. This can be done using the ljust() method. This will take a string and a width as arguments and will return a left-justified string within the specified width.

#input:

```
a = "apple"
b = a.ljust(20)
print(b, "is a name of fruit.")
```

#output:

```
apple                is a name of fruit.
```

Centering a string means that the text is aligned in the middle of a container. This can be done using the center() method. This will take a string and a width as arguments and will return a centered string within the specified width.

#input:

```
a = "apple"
b = a.center(20)
print(b, "is a name of fruit.")
```

#output:

```
       apple         is a name of fruit.
```

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

Answer:

The best way to remove whitespace characters from the start or end in python is to use the .strip() method. This method will remove any leading or trailing whitespace characters from a string, such as spaces, tabs, and newlines.

#input:

```
string = "  Hello World!     "
string = string.strip()
print(string)
```

#Output:

```
Hello World!
```