ASSIGNMENT - 8(PYTHON)

1. Is the Python Standard Library included with PyInputPlus?

No, the Python Standard Library is not included with PyInputPlus. PyInputPlus is a separate Python library that provides additional features for handling user input. It builds upon the functionality of the Python Standard Library's input() function but adds features like input validation, retry mechanisms, and more, making it easier to handle user input in a robust manner.

2. Why is PyInputPlus commonly imported with import pyinputplus as pypi?

Importing PyInputPlus as pypi is a common convention used by developers to make the code more concise and readable.

Here's why it's commonly imported this way:

Conciseness: By importing PyInputPlus as pypi, you can use shorter references to its functions and classes in your code, which can make your code easier to read and write.

Clarity: While import pyinputplus is perfectly valid, it requires you to prefix each usage of PyInputPlus functions and classes with pyinputplus, which can make the code look cluttered, especially if you're using PyInputPlus extensively. By importing it as pypi, you can avoid repetition and make the code more streamlined.

Avoiding Name Collisions: Importing PyInputPlus as pypi can also help avoid potential name collisions with other modules or functions in your codebase. Since pypi is a less common name, it's less likely to clash with existing names in your project.

3. How do you distinguish between inputInt() and inputFloat()?

inputInt():

inputInt() is used to prompt the user for an integer input.
It only accepts integer values as input. If the user enters a non-integer value, PyInputPlus will raise a ValidationException.
It returns an integer value.

inputFloat():

inputFloat() is used to prompt the user for a floating-point input.
It accepts both integer and floating-point values as input. If the user enters a non-numeric value, PyInputPlus will raise a ValidationException.
It returns a floating-point value.

4. Using PyInputPlus, how do you ensure that the user enters a whole number between 0 and 99?

You can use the inputInt() function from PyInputPlus along with the min and max parameters to ensure that the user enters a whole number between 0 and 99.

```
import pyinputplus as pypi

num = pypi.inputInt(prompt="Enter a number between 0 and 99: ", min=0, max=99)
print("You entered:", num)
```

In this example:

prompt="Enter a number between 0 and 99: " is the prompt displayed to the user.
min=0 specifies that the input must be greater than or equal to 0.
max=99 specifies that the input must be less than or equal to 99.

5. What is transferred to the keyword arguments allowRegexes and blockRegexes?

In PyInputPlus, the allowRegexes and blockRegexes keyword arguments are used to specify regular expressions that match strings that are allowed or blocked from being entered by the user.

allowRegexes: This keyword argument accepts a list of regular expressions. If any of these regular expressions match the user's input, the input is considered valid. Only inputs that match at least one regular expression from the allowRegexes list will be accepted.

blockRegexes: This keyword argument also accepts a list of regular expressions. If any of these regular expressions match the user's input, the input is considered invalid. Inputs that match any regular expression from the blockRegexes list will be rejected.

6. If a blank input is entered three times, what does inputStr(limit=3) do?

If a blank input is entered three times consecutively while using inputStr(limit=3) in PyInputPlus, it will raise a TimeoutException.

7. If blank input is entered three times, what does inputStr(limit=3, default='hello') do?

If blank input is entered three times consecutively while using inputStr(limit=3, default='hello') in PyInputPlus, the function will return the default value 'hello'.