**1. Is the Python Standard Library included with PyInputPlus?**

Ans:- No, the Python Standard Library is not included with PyInputPlus. PyInputPlus is a separate library that provides additional functionality for taking user input in a more controlled and user-friendly manner than the built-in input() function. While PyInputPlus builds upon the functionality of the standard library's input() function, it does not include the entire Python Standard Library.

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

Ans:- PyInputPlus is commonly imported with the alias "pypi" (import pyinputplus as pypi) for brevity and convenience. Using the alias "pypi," developers can use shorter names for PyInputPlus functions, making the code easier to read and write. However, it's worth noting that you can choose any alias you prefer.

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

Ans:- The functions inputInt() and inputFloat() are used to get integer and floating-point input, respectively, from the user. The difference between the two functions is in the type of value they accept. inputInt() only allows the user to enter an integer, and if they provide a non-integer value, it will keep prompting for a valid integer. On the other hand, inputFloat() allows the user to enter a floating-point number, and it will keep prompting until a valid float is entered..

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

Ans:- To ensure that the user enters a whole number between 0 and 99 using PyInputPlus, you can use the inputInt() function with the min and max arguments set to 0 and 99, respectively. Here's an example:

import pyinputplus as pypi

number = pypi.inputInt(prompt="Enter a whole number between 0 and 99: ", min=0, max=99)

print("You entered:", number)

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

1. Ans:- The keyword arguments allowRegexes and blockRegexes in PyInputPlus allow you to define regular expressions as lists of strings. The purpose of these arguments is to specify which patterns are allowed or blocked in the user input.

* allowRegexes: If provided, the user input must match at least one of the regular expressions in this list to be considered valid.
* blockRegexes: If provided, the user input must not match any of the regular expressions in this list to be considered valid.

You can use these arguments to control what kind of input is acceptable or rejected based on specific patterns or formats.

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

Ans- If a blank input is entered three times when using inputStr(limit=3), PyInputPlus raises a RetryLimitException. This means that after three attempts to get valid input (i.e., non-blank input), the function will give up and raise the exception. Here's an example:

import pyinputplus as pypi

try:

result = pypi.inputStr(prompt="Enter something: ", limit=3)

print("You entered:", result)

except pypi.RetryLimitException:

print("You failed to provide non-blank input three times.")

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

Ans:- If blank input is entered three times when using inputStr(limit=3, default='hello'), PyInputPlus will return the default value 'hello' instead of raising a RetryLimitException. In this case, if the user fails to provide non-blank input three times, the function will return the default value. Here's an example:

import pyinputplus as pypi

result = pypi.inputStr(prompt="Enter something: ", limit=3, default='hello')

print("You entered:", result)