1. Is the Python Standard Library included with PyInputPlus?

Ans – PyInputPlus is a Python module that provides input() and raw\_input() like functions with additional validation features, such as re-prompting the user if they enter invalid input, validating for numeric, boolean, date, time, or yes/no responses, and specifying regexes for whitelists or blacklists of responses.

The Python Standard Library is a collection of modules that are built-in to the Python interpreter and provide a wide range of functionality, such as file handling, web development, data processing, and more.

PyInputPlus is not included with the Python Standard Library. It is a third-party module that needs to be installed separately using pip or another package manager.

=======================================================================

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

Ans - PyInputPlus is a Python module that provides input() and raw\_input() like functions with additional validation features.

It is often imported with the alias pyip to save typing and make the code more concise.

For example, instead of writing pyinputplus.inputNum(), we can write pyip.inputNum().

This is a common practice in Python programming to use aliases for modules that have long names.

============================================================================

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

Ans- The difference between inputInt() and inputFloat() is in the data type and the validation of the user input.

inputInt() is a PyInputPlus function that prompts the user to enter an integer value, which is a whole number without any decimal point.

inputFloat() is similar to inputInt(), but it prompts the user to enter a floating-point value, which is a number that can have a decimal point.

Both functions can take additional parameters to specify the minimum, maximum, or range of the valid input values.

For example, if we want to ask the user to enter a number between 1 and 10, we can use inputInt() or inputFloat(), depending on whether we want to accept decimal numbers or not.

# Using inputInt()

import pyinputplus as pyip

number = pyip.inputInt(prompt="Enter a number between 1 and 10: ", min=1, max=10)

print("You entered:", number)

# Using inputFloat()

import pyinputplus as pyip

number = pyip.inputFloat(prompt="Enter a number between 1 and 10: ", min=1, max=10)

print("You entered:", number)

If we run these codes and enter 4.3, the first one will give you an error and ask you to enter a valid integer, while the second one will accept it as a valid floating-point number.

===========================================================================

1. 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, we can use the inputInt() function with the min and max parameters set to 0 and 99 respectively.

For Ex.-

import pyinputplus as pyip

num = pyip.inputInt(prompt="Please enter a number between 0 and 99: ", min=0, max=99)

print("You entered:", num)

This code will prompt the user to enter a number between 0 and 99, and will re-prompt if the input is invalid, such as a non-integer, a negative number, or a number greater than 99

==========================================================================

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

Ans- The keyword arguments allowRegexes and blockRegexes are used to specify the validation rules for user input in PyInputPlus functions.

They take a list of regular expression strings as their values. A regular expression is a sequence of characters that defines a search pattern for text.

The allowRegexes argument contains a list of regular expressions that match the input that we want to allow. Only input that matches at least one of the regular expressions in this list will be accepted by the PyInputPlus function.

For example, if we want to allow only alphanumeric input, we can use allowRegexes=[r'\w+'].

The blockRegexes argument contains a list of regular expressions that match the input that we want to block. Any input that matches any of the regular expressions in this list will be rejected by the PyInputPlus function.

For example, if we want to block any input that contains spaces, you can use blockRegexes=[r'\s'].

We can use both allowRegexes and blockRegexes together to create more complex validation rules. However, note that allowRegexes will override blockRegexes, meaning that any input that matches both lists will be allowed.

For example, if we want to allow only six-digit numbers, but block any number that starts with zero, we can use allowRegexes=[r'\d{6}'] and blockRegexes=[r'^0'].

=============================================================================

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

Ans- If a blank input is entered three times, the inputStr(limit=3) function in Python will return the default value specified by the default parameter, if it is provided. If the default parameter is not provided, it will raise a RetryLimitException error, which indicates that the user has exceeded the maximum number of retries allowed.

For example, if we want to ask the user to enter their name, but allow them to skip it by pressing Enter three times, we can use the following code:

import pyinputplus as pyip

name = pyip.inputStr(prompt="Please enter your name: ", default="Anonymous", limit=3)

print("Your name is:", name)

If the user enters a non-blank input, such as “John”, it will be accepted as valid input. If the user enters a blank input three times in a row, the name variable will be set to “Anonymous” and the program will continue without raising an error.

========================================================================

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

Ans- If a blank input is entered three times, the inputStr(limit=3, default='hello') function in Python will return the default value ‘hello’, which is specified by the default parameter. This is because the limit parameter specifies the maximum number of retries allowed for invalid input, and if the user enters a blank input three times in a row, it will be considered invalid input. The default parameter is used to provide a fallback value in case the user does not provide any valid input within the specified limit.

For example, if we want to ask the user to enter their username, but allow them to skip it by pressing Enter three times, you can use the following code:

import pyinputplus as pyip

username = pyip.inputStr(prompt="Please enter your username: ", default="hello", limit=3)

print("Your username is:", username)

If the user enters a non-blank input, such as “John”, it will be accepted as valid input. If the user enters a blank input three times in a row, the username variable will be set to “hello” and the program will continue without raising an error.