1. Is the Python Standard Library included with PyInputPlus?

Ans->PyInputPlus is not a part of the Python Standard Library, so you must install it separately using Pip. To install PyInputPlus, run pip install --user pyinputplus from the command line.  To check if PyInputPlus installed correctly, import it in the interactive shell:

>>> **import pyinputplus**

If no errors appear when you import the module, it has been successfully installed.

PyInputPlus has several functions for different kinds of input:

**inputStr()** Is like the built-in input() function but has the general PyInputPlus features. You can also pass a custom validation function to it

**inputNum()** Ensures the user enters a number and returns an int or float, depending on if the number has a decimal point in it

**inputChoice()** Ensures the user enters one of the provided choices

**inputMenu()** Is similar to inputChoice(), but provides a menu with numbered or lettered options

**inputDatetime()** Ensures the user enters a date and time

**inputYesNo()** Ensures the user enters a “yes” or “no” response

**inputBool()** Is similar to inputYesNo(), but takes a “True” or “False” response and returns a Boolean value

**inputEmail()** Ensures the user enters a valid email address

**inputFilepath()** Ensures the user enters a valid file path and filename, and can optionally check that a file with that name exists

**inputPassword()** Is like the built-in input(), but displays \* characters as the user types so that passwords, or other sensitive information, aren’t displayed on the screen

These functions will automatically reprompt the user for as long as they enter invalid input:

>>> **import pyinputplus as pyip**  
>>> **response = pyip.inputNum()**  
**five**  
'five' is not a number.  
**42**  
>>> **response**  
42

1. Why is PyInputPlus often imported as pyip with import pyinputplus?

Ans-> You can **import** the module with **import pyinputplus** as **pyip** so that you can enter a shorter name when calling the module's functions. **PyInputPlus** has functions for entering a variety of input, including strings, numbers, dates, yes/no, True/False, emails, and files.

Example :

>>> **response = input('Enter a number: ')**  
Enter a number: **42**  
>>> **response**  
'42'  
>>> **import pyinputplus as pyip**  
>>> **response = pyip.inputInt(prompt='Enter a number: ')**  
Enter a number: **cat**  
'cat' is not an integer.  
Enter a number: **42**  
>>> **response**  
42

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

Ans-> **inputInt()** : Accepts an integer value. This also takes additional parameters 'min', 'max', 'greaterThan' and 'lessThan' **for** bounds. Returns an int. **inputFloat()** : Accepts a floating-point numeric value.

Example -:

pyinputplus.inputInt(prompt=”, default=None, blank=False, timeout=None, limit=None, strip=None, allowRegexes=None, blockRegexes=None, applyFunc=None, postValidateApplyFunc=None, min=None, max=None, lessThan=None, greaterThan=None)

Prompts the user to enter an integer value. Returns the integer as an int value.

Run help(pyinputplus.parameters) for an explanation of the common parameters.

• min (None, float): If not None, the minimum accepted numeric value, including the minimum argument.

• max (None, float): If not None, the maximum accepted numeric value, including the maximum argument.

• greaterThan (None, float): If not None, the minimum accepted numeric value, not including the greaterThan argument.

• lessThan (None, float): If not None, the maximum accepted numeric value, not including the lessThan argument.

>>> import pyinputplus as pyip

>>> response = pyip.inputInt()

42

>>> response

42

>>> type(response)

<class ‘int’>

>>> response = pyip.inputInt(min=4)

4

>>> response

4

>>> response = pyip.inputInt(min=4)

3

Number must be at minimum 4. -5

Number must be at minimum 4. 5

>>> response

5

>>> response = pyip.inputInt(blockRegexes=[r'[13579]$'])

43

This response is invalid.

41

This response is invalid.

42

>>> response

42

>>> response = pyip.inputInt()

42.0

>>> response

42

>>> type(response)

<class ‘int’>

pyinputplus.inputFloat(prompt=”, default=None, blank=False, timeout=None, limit=None, strip=None, allowRegexes=None, blockRegexes=None, applyFunc=None, postValidateApplyFunc=None, min=None, max=None, lessThan=None, greaterThan=None)

Prompts the user to enter a floating point number value. Returns the number as a float.

Run help(pyinputplus.parameters) for an explanation of the common parameters.

• min (None, float): If not None, the minimum accepted numeric value, including the minimum argument.

• max (None, float): If not None, the maximum accepted numeric value, including the maximum argument.

• greaterThan (None, float): If not None, the minimum accepted numeric value, not including the greaterThan argument.

• lessThan (None, float): If not None, the maximum accepted numeric value, not includingthe lessThan argument.

>>> import pyinputplus as pyip

>>> response = pyip.inputFloat()

42

>>> response

42.0

>>> type(response)

<class ‘float’>

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

Ans-> pip install pyinputplus

import pyinputplus as pyip

# integer input with

# specific bounds

inp = pyip.inputInt(prompt = "Enter a whole number... ",

min = 0, lessThan = 100 )

print(inp)

output:-

Enter a whole number... 222

Number must be less than 100

Enter a whole number...

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

Ans->If you specify both an allowRegexes and blockRegexes argument, the allow list overrides the block list. For example, enter the following into the interactive shell, which allows 'caterpillar' and 'category' but blocks anything else that has the word 'cat' in it:

>>> import pyinputplus as pyip  
>>> response = pyip.inputStr(allowRegexes=[r'caterpillar', 'category'],  
blockRegexes=[r'cat'])  
cat  
This response is invalid.  
catastrophe  
This response is invalid.  
category  
>>> response  
'category'

Specify a list of regular expression strings that a PyInputPlus function won’t accept by using the blockRegexes keyword argument.

>>> import pyinputplus as pyip  
>>> response = pyip.inputNum(blockRegexes=[r'[02468]$'])  
42  
This response is invalid.  
44  
This response is invalid.  
43  
>>> response  
43

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

#### Ans->*The blank Keyword Argument*

By default, blank input isn’t allowed unless the blank keyword argument is set to True:

>>> import pyinputplus as pyip  
>>> response = pyip.inputStr(limit = 3)  
(blank input entered here)  
Blank values are not allowed.  
(blank input entered here)

Blank values are not allowed.

(blank input entered here)

Blank values are not allowed.

str  
>>> response  
42  
>>> response = pyip.inputStr(blank=True)  
(blank input entered here)  
>>> response  
''

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

Ans >

|  |
| --- |
| import pyinputplus as pyip    # integer input with  # limited number of tries  inp = pyip.inputInt(prompt = "Enter an Integer... ",                      default = ‘hello’, limit = 3)    print(inp) |

**Output :**

Enter an Integer... hello

'hello' is not an integer.

Enter an Integer... abc

'abc' is not an integer.

Enter an Integer...

Blank values are not allowed.

hello