Built-in Functions in Python - Python provides a rich set of built-in functions that simplify coding tasks. During coding, it is recommended to use first the built-in functions

Built-in Functions in String –

1. print () – used to print a message on screen

e.g 1 –	e.g 2 –

print("Hello, world!")
 	
Output – Hello World
	name = "Anurag"
	age = 25
	print(f"My name is {name} and I am {age} years old.")
	
	Output - My name is Anurag and I am 25 years old.

2. input () - input () function is used to take user input.

e.g 1 – Basic Use	e.g 2 - Converting Input to Int or Float

name = input("Enter your name: ")
print("Hello,", name)
 	
Output - Enter your name: Anurag
Hello Anurag
	age = int(input("Enter your age: ")) # For integers
	height = float(input("Enter your height in meters: ")) # For decimals
	
	Output - Enter your age: 30
	Enter your height in meters: 80.5

3. **len()** function returns the length of a string. Sample Program – text = "Python is awesome" length = len(text) print("Length of the string:", length)

Output - Length of the string: 17

4. **upper () & lower ()** – upper() converts all characters in the string to uppercase, lower() converts all characters in the string to lowercase**.** Program – text = "Python is awesome" uppercase = text.upper() lowercase = text.lower() print("Uppercase:", uppercase) print("Lowercase:", lowercase)

```
Output - Uppercase: PYTHON IS AWESOME
         Lowercase: python is awesome
```

5. **String Concatenation**—String concatenation means joining two or more strings to create a new single string. We don't have a direct function for string concatenation; we do this by using the "+ operator."

Program –

```
str1    =    "Hello"
str2 = "World"
result = str1 + " " + str2
print(result)
```

Output = Hello World

6. **replace ()** - Replace a specified substring with another substring. Program –

```
text = "Python is awesome"
new_text        =        text.replace("awesome",        "great")
print("Modified text:", new_text)
```

Output – Modified text: Python is great

7. **split()** - it splits a string into substrings and returns a list of them. Program –

```
text      =      "Python      is
awesome"      words      =
text.split()    print("Words:",
words)
```

Output - Words: ['Python', 'is', 'awesome']

8. **in operator** – in operator used to check if a specified value (such as a substring, element, or key) exists inside a sequence (such as a string, list, tuple, or dictionary). Program -

```
text = "Python is awesome"
substring  =  "is"  if  substring  in
text: print(substring, "found in the
text")
```

Output - is found in the text

9. **strip()** method is used to remove leading and trailing whitespace

Program – text = " Some spaces around “or text = ">>>Hello,

```
World! <<<"

stripped_text          =          text.strip()

print("Stripped text:", stripped_text)
```

Output - Stripped text: Some spaces around

Regular Expressions (or RegEx) in Python - are used for pattern matching within strings, such as searching, extracting, replacing, or validating text.

The re module in Python is used for working with regular expressions.

Examples of regex usage: matching emails, and phone numbers, or extracting data from text.

RegEx Functions –

- findall() - Returns a list containing all matches
- search() - It is used to search for a pattern in a given string. If a match is found, search() returns the matched object; otherwise, it returns None.
- Split()- divide a string into a list of substrings based on a specified delimiter. Sub()- replace parts of a string that match the pattern

Below are Examples of RegEx Expressions –

1. findall() –

```
import re

txt = "The rain in Spain"
x = re.findall("ai", txt)
print(x)
```

   Output – ['ai', 'ai']

2. search() – import re

```
text   =   "The   quick   brown   fox"
pattern = "brown"

search   =   re.search(pattern,   text)   if
search:        print("Pattern        found:",
search.group())
else:    print("Pattern    not
   found")
```

   Output - Pattern found: brown

   Break-up –

   - re.search() function searches for the first occurrence of the pattern "brown" in the string "The quick brown fox".
   - search.group() retrieves the exact text that matches the pattern.
   - if search: This check if the “search” variable holds a match object

3. split() – import re

   text = "apple,banana,orange,grape"
   pattern = ","

   split_result = re.split(pattern, text)
   print("Split result:", split_result)

   Output - Split result: ['apple', 'banana', 'orange', 'grape']

4. sub() import re text = "The quick

   brown fox jumps over the lazy brown

   dog" pattern = "brown" replacement =

   "red"

   new_text = re.sub(pattern, replacement, text)

   print("Modified text:", new_text)

   output - Modified text: The quick red fox jumps over the lazy red dog

==

Program of Integer Data Type

```
num1 = 10 num2 = 5 result1 = num1 // num2     #    Integer
Division
print("Integer Division:", result1)
result2 = num1 % num2                          # Modulus (Remainder)
print("Modulus (Remainder):", result2)
result3 = abs(-7)                              # Absolute Value
print("Absolute Value:", result3)
```

```
Output - Integer Division: 2
Modulus (Remainder): 0
Absolute Value: 7
```

Program of Float Data Type –

```
# Float variables
num1     =     5.0
num2 = 2.5
result1 = num1 + num2                 # Basic Arithmetic
print("Addition:", result1)
result2 = num1 - num2
print("Subtraction:", result2)
result3 = num1 * num2
print("Multiplication:", result3)
result4 = num1 / num2
print("Division:", result4)
result5 = round(3.14159265359, 2) # Rounds to 2 decimal places              # Rounding
print("Rounded:", result5)
```

==

Output – Addition: 7.5

```
Subtraction: 2.5

Multiplication: 12.5

Division: 2.0

Rounded: 3.14
```