Important Inbuilt Libraries

Math Library

The *math* module is a standard Python module providing mathematical functions. It includes a wide range of functions for basic arithmetic, trigonometry, logarithmic, exponential, and more. Here are some commonly used functions from the *math* module:

Basic Arithmetic Functions

1. 'math.ceil(x)`:

Returns the smallest integer greater than or equal to x.

Python3

```python
import math
result = math.ceil(4.2)
print(result)
```

Output - 5

2. $math.floor(x)$:

Returns the largest integer less than or equal to x.

Python3

```python
import math
 result = math.floor(4.9)
 print(result)
```

Output - 4

3. $math.trunc(x)$:

Returns the truncated integer value of x.

Python3

```
import math
result = math.trunc(4.9)
print(result)
```

Output - 4

4. $math.sqrt(x)$:

Returns the square root of x.

Python3

```
import math
result = math.sqrt(16)
print(result)
```

Output - 4.0

Trigonometric Functions

5. $math.sin(x)$, $math.cos(x)$, $math.tan(x)$:

Return the sine, cosine, and tangent of x (measured in radians).

Python3

```
import math
angle = math.radians(45)  # Convert degrees
sin_result = math.sin(angle)
cos_result = math.cos(angle)
tan_result = math.tan(angle)
print("sin_result :",sin_result, "cos_result
```

Output - sin_result : 0.7071067811865475 cos_result : 0.7071067811865476 tan_result : 0.9999999999999999

Logarithmic and Exponential Functions

6. $math.log(x, base)$:

Returns the natural logarithm of x with the specified base.

Python3

```python
import math
result = math.log(16, 2)
print(result)
```

output - 4.0

7. $math.exp(x)$:

Returns e raised to the power of x.

Python3

```python
import math
    result = math.exp(2)
    print(result)
```

Output - 7.3890560989306495

Constants

8. $math.pi$:

Mathematical constant representing the ratio of the circumference of a circle to its diameter.

9. $math.e$:

Mathematical constant representing the base of the natural logarithm.

Python3

```python
import math
print(math.pi)
print(math.e)
```

Output - 3.141592653589793

2.718281828459045

You can explore more functions and constants in the official Python documentation: https://docs.python.org/3/library/math.html

Random Library

The *random* module in Python is a standard library module that provides functions for generating random numbers, selecting random elements, and performing various randomization tasks. It is commonly used in applications involving simulations, games, statistical sampling, cryptography, and more.

Here are some commonly used functions from the *random* module:

Generating Random Numbers:

1. *random*():

Returns a random floating-point number in the range [0.0, 1.0).

Python3

```python
import random
value = random.random()
print(value)
```

Output - 0.14277691938900905

2. *randint*(a, b):

Returns a random integer between a and b (inclusive).

Python3

```python
import random
value = random.randint(1, 10)
print(value)
```

Output - 7

3. $uniform(a, b)$:

Returns a random floating-point number in the range [a, b) or [a, b] if b is included.

Python3

```python
import random
value = random.uniform(1.0, 5.0)
print(value)
```

Output - 1.1674375168224214

Randomizing Sequences:

4. $shuffle(sequence)$:

Randomly shuffles the elements of a sequence in place.

Python3

```python
import random
my_list = [1, 2, 3, 4, 5]
random.shuffle(my_list)
print(my_list)
```

Output - [5, 3, 2, 4, 1]

5. $choice(sequence)$:

Returns a random element from a sequence.

Python3

```python
import random
my_list = [1, 2, 3, 4, 5]
value = random.choice(my_list)
print(value)
```

Output - 4

6. $sample(sequence, k)$:

Returns a new list with k unique random elements sampled from the sequence without replacement.

Python3

```python
import random
my_list = [1, 2, 3, 4, 5]
sampled = random.sample(my_list, 3)
print(sampled)
```

Output - [2, 5, 4]

DateTime Library

In Python, the *datetime* module provides classes for working with dates and times. It allows you to represent dates, times, and intervals, and perform operations on them. Here are some key components of the *datetime* module:

1. *datetime* Class:

The *datetime* class represents a date and a time. It has attributes like year, month, day, hour, minute, second, and microsecond.

Python3

```python
from datetime import datetime
current_datetime = datetime.now()
print(current_datetime)
```

Output - 2023-11-30 08:17:47.597082

2. *date* Class:

The *date* class represents a date (year, month, and day) without a time component.

Python3

```
from datetime import date
current_date = date.today()
print(current_date)
```

Output - 2023-11-30

3. *time* Class:

The *time* class represents a time (hour, minute, second, and microsecond) without a date component.

Formatting and Parsing:

You can format *datetime* objects as strings and parse strings into *datetime* objects using the *strftime* and *strptime* methods.

Python3

```
from datetime import datetime
current_datetime=datetime.now()
# Format a datetime object as a string
formatted_date = current_datetime.strftime("
print(formatted_date)

# Parse a string into a datetime object
parsed_date = datetime.strptime("2023-01-01
print(parsed_date)
```

2023-01-01 12:30:00

Date Arithmetic:

Performing arithmetic operations with dates is possible using the *timedelta* class.

Python3

```
1 from datetime import datetime, timedelta
2 current_datetime=datetime.now()
3 future_date = current_datetime + timedelta(da
4 print(future_date)
```

Output - 2023-12-07 08:29:56.154919

For more details and examples, refer to the official Python documentation on the *datetime* module https://docs.python.org/3/library/datetime.html

Challenge Galore

Problem Statement - Write a Python program that calculates the remaining days, hours, and minutes until the event. Additionally, provide the percentage completion of the event based on the current time compared to the event's start time.

Python3

```
5
6  current_datetime = datetime.now()
7
8  time_difference = event_date - current_datet
9  remaining_days = time_difference.days
10 remaining_hours, remaining_seconds = divmod(
11 remaining_minutes = remaining_seconds // 60
12
13 # Calculate percentage completion
14 total_seconds_in_event = (event_date - event
15 elapsed_seconds = (current_datetime - event_
16 percentage_completion = (elapsed_seconds / t
17
18
19 print(f"Remaining time until the event: {rem
20 print(f"Percentage completion of the event:
```

Output - Remaining time until the event: 31 days, 9 hours, 0 minutes.

Percentage completion of the event: -3971%

Mark as Read

Report An Issue

If you are facing any issue on this page. Please let us know.