



MULTITHREADING AND MULTIPROCESSING



MULTITHREADING

The threading module is a Python module that provides a number of functions and classes for creating and managing threads. Threads are lightweight processes that can run concurrently with the main thread of execution. This can be useful for tasks that can be broken down into smaller subtasks, which can then be executed by different threads.

The threading module provides a number of functions for creating and managing threads. These functions include:

Thread() : This function creates a new thread object. The thread object can then be started using the **start()** method.

start() : This method starts the thread object. The thread will then run concurrently with the main thread of execution.

join() : This method waits for the thread object to finish executing.

run() : This method is the entry point for the thread. The code that you want the thread to execute should be placed in this method.

MULTITHREADING

The threading module also provides a number of classes for managing threads. These classes include:

Thread : This class represents a thread object.

Lock : This class provides a lock that can be used to synchronize access to shared resources.

Semaphore : This class provides a semaphore that can be used to control access to shared resources.

Condition : This class provides a condition variable that can be used to wait for certain conditions to be met.



python.py

```
import threading
def test(id):
    print("prog start %d" % id)
thread = [threading.Thread(target=test , args=(i,) )for i in
range(10)]
for t in thread :
    t.start()
```

MULTITHREADING



python.py

```
import time
def test1(id) :
    for i in range(10) :
        print("test1 %d printing %d %s" %(id,i,time.ctime()))
        time.sleep(1)
```

WITHOUT THREAD



python.py

```
test1(0)
```

VS

WITH THREAD




python.py

```
thread1 = [threading.Thread(target=test1 , args = (i,))
for i in range(3)]
for t in thread1:
    t.start()
```

MULTIPROCESSING

The multiprocessing module in Python provides a number of functions and classes that allow you to write parallel code. This can be useful for speeding up computationally intensive tasks, or for running multiple tasks simultaneously.



```
import multiprocessing
def test():
    print("this is my multiprocessing prog")

if __name__ == '__main__':
    m = multiprocessing.Process(target=test)
    print("this is my main prog")
    m.start()
    m.join()
```

EXAMPLE



python.py

```
import multiprocessing

def square(x):
    return x * x

if __name__ == '__main__':
    pool = multiprocessing.Pool()
    results = pool.map(square, range(10))
    pool.close()
    pool.join()

    print(results)
```

PRACTICE QUESTIONS

1. What is multithreading in python? Why is it used? Name the module used to handle threads in python
2. Why threading module used? write the use of the following functions
 - a. `activeCount()`
 - b. `currentThread()`
 - c. `enumerate()`
3. Explain the following functions
 - `run()`
 - `start()`
 - `join()`
 - `isAlive()`
4. Write a python program to create two threads. Thread one must print the list of squares and thread two must print the list of cubes
5. State advantages and disadvantages of multithreading
6. Explain deadlocks and race conditions.

SUBSTITUTION METHOD

Example 1

Substitution Method

(Assumption
↳ Already
Provided
Recurrence Relation)

$$T(n) = \begin{cases} 1 & n=1 \\ T(n-1) + n & n > 1 \end{cases}$$

$$T(1) = 1$$

Base
Case
Condition

$$T(n) = T(n-1) + n \quad \text{--- (2) (Substitute the Recursive term)}$$

$$T(n-1) = T(n-2) + n-1 \quad \text{--- (1)}$$

$$T(n) = T(n-2) + n-1 + n$$

$$= T(n-3) + n-2 + n-1 + n$$

$$\begin{matrix} \text{k times} \\ \downarrow \end{matrix} \quad \begin{matrix} n-k = 1 \\ \underline{n-1 = k} \end{matrix}$$

$$= T(n-k) + (n-k+1) + (n-k+2) + \dots + n-2 + n-1 + n$$

$$= T(n-(n-1)) + (n-(n-1)+1) + (n-(n-1)+2) + \dots + n-2 + n-1 + n$$

$$= T(1) + 2 + 3 + \dots + n-2 + n-1 + n$$

$$\Rightarrow \underline{T(1) + 2 + 3 + \dots + n-2 + n-1 + n}$$

↪ Mathematical Series

$$1 + 2 + 3 + \dots + n-2 + n-1 + n$$

(Sum of n natural numbers)

$$n(n+1)/2 = (n^2+n)/2 \gg 1$$

$$\Rightarrow \underline{O(n^2)}$$

SUBSTITUTION METHOD

Recurrence Relation \rightarrow Iteration

Substitution Method

$$T(n) = \begin{cases} 1 & n = 1 \rightarrow \text{Base case} \\ T(n-1) + \frac{1}{n} & n > 1 \end{cases}$$

Definition

$$T(n) = T(n-1) + \frac{1}{n}$$

↓
Recursive Term

$$T(n-1) = T(n-1-1) + \frac{1}{n-1}$$

$$= T(n-2) + \frac{1}{n-1}$$

$$T(n) = T(n-2) + \frac{1}{n-1} + \frac{1}{n}$$

n-2+1

$$T(n) = T(n-3) + \frac{1}{n-2} + \frac{1}{n-1} + \frac{1}{n}$$

n-2+1

$T(1) = 1$

$n-k = 1$

$n-1 = k$

$$T(n) = T(n-k) + \frac{1}{n-k+1} + \frac{1}{n-k+2} + \dots + \frac{1}{n-1} + \frac{1}{n}$$

$$\Rightarrow T(n - (n-1)) + \frac{1}{n - (n-1) + 1} + \frac{1}{n - (n-1) + 2} + \dots + \frac{1}{n-1} + \frac{1}{n}$$

$$\Rightarrow T(1) + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{n-1} + \frac{1}{n}$$

↓

$$\Rightarrow 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{n-1} + \frac{1}{n}$$

$$\Rightarrow \underline{\underline{O(\log n)}}$$

SUBSTITUTION METHOD

Substitution Method

Recurrence Relation

$$T(n) = \begin{cases} 1 & n=1 \rightarrow \text{Base case condition} \\ T\left(\frac{n}{2}\right) + n & n>1 \end{cases}$$

+ Recursive Term

$$T(n) = T\left(\frac{n}{2}\right) + n \quad \text{1st time}$$

$$T\left(\frac{n}{2}\right) = T\left(\frac{n}{2^2}\right) + \frac{n}{2}$$

Pattern

$$T(n) = T\left(\frac{n}{2^k}\right) + \frac{n}{2} + \frac{n}{2} + \dots + n \quad \text{2nd time}$$

$T(1) = 1$

$k \rightarrow ??$

$n = 2^k$
 $k = \log_2 n$

$$T(n) = T\left(\frac{n}{2^k}\right) + \frac{n}{2^{k-1}} + \frac{n}{2^{k-2}} + \dots + n$$

$\frac{n}{2^k} = 1 \rightarrow \text{base case condition}$

$$\frac{1}{2^{\log_2 n}} \Rightarrow n \frac{1}{2^{\log_2 n}} = n$$

$T(n) = T\left(\frac{n}{2^{\log_2 n}}\right) + \left\{ \frac{n}{2^{\log_2 n - 1}} + \frac{n}{2^{\log_2 n - 2}} + \dots + \frac{n}{2} + n \right\}$

Mathematical Series (Basic)

$$= T(1) + n \left(\left(\frac{1}{2}\right)^0 + \left(\frac{1}{2}\right)^1 + \dots + \left(\frac{1}{2}\right)^{\log_2 n - 1} \right)$$

\hookrightarrow GP Series

$a = \left(\frac{1}{2}\right)^0 = 1$

$T(1) = 1$

$r = \frac{1}{2}$

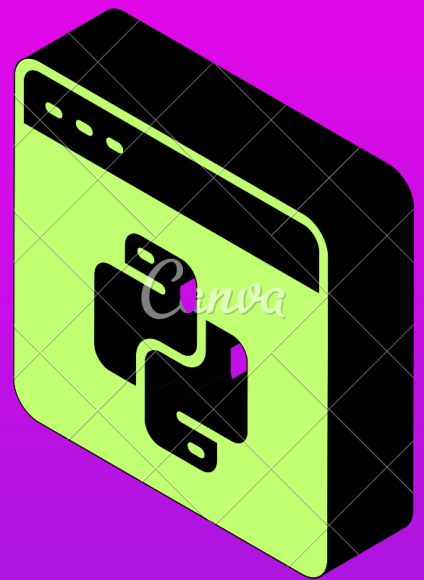
$\frac{a(1-r^n)}{1-r}$

$$\Rightarrow 1 + \left\{ \frac{(1 - \frac{1}{2}^{\log_2 n})}{1 - \frac{1}{2}} \right\} \cdot n$$

$$\Rightarrow 1 + \frac{(1)^n}{(1)} \rightarrow \text{Higher term}$$

$$\Rightarrow O(n) \checkmark$$

FOLLOW FOR MORE



FOLLOW ME

