## Q1. What is multithreading in python? Why is it used? Name the module used to handle threads in python ?

In [1]:

```
# Ans:-Multithreading refers to concurrently executing multiple threads by rapidly switching the
#      control of the CPU between threads (called context switching).

#     USED MULTITHREADING IN PYTHON
#       Multithreading in Python streamlines the efficient utilization of resources as the threads
#       share the same memory and data space. It also allows the concurrent appearance of multiple tasks and reduces the response time.
#       This improves the performance.

#     There are two main modules which can be used to handle threads in Python:
#     The thread module, and. The threading module.
```

## Q2. Why threading module used? Write the use of the following functions

1. activeCount ()

2.currentThread()

3. enumerate()

In [2]:

```
# Ans:-Python threading allows you to have different parts of your program run concurrently and can simplify your design.
#      1.activecount():-active_count() is an inbuilt method of the threading module,
#                       it is used to return the number of Thread objects that are active at any instant
#      2.currentThread():-Returns the number of thread objects in the caller's thread control.
#      3.enumerate():-Enumerate is a built-in function in python that allows you to keep track of the number of iterations (loops) in a loop
```

## Q3. Explain the following functions?

1.run()

2.start()

3.join()

4.isAlive()

In [3]:

```
# run():-open a command-line and type in the word python , or python3 if you have both versions, followed by the path to your script,
# start():- method is an inbuilt method of the Thread class of the threading module, it is used to start a thread's activity
# join():- Join in Python is an in-built method used to join an iterable's elements,
#          separated by a string separator, which is specified by you.
# isAlive():-The Thread. is_alive() method is an inbuilt method of the Thread class of the threading module,
#            it is used to check whether that thread is alive or not.
```

## Q4. 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

In [10]:

```
import threading
def cube(num):
    print("cube.{}".format(num*num*num))
def square(num):
    print("square.{}".format(num*num))
if __name__ =="__main__":
    t1 =[ threading.Thread(target=square, args=(10,))]
    t2 = [threading.Thread(target=cube, args=(10,))]
```

In [11]:

```
for a in t1:
    a.start()
```

```
square.100
```

In [12]:

```
cube(12)
```

```
cube.1728
```

In [13]:

```
square(12)
```

```
square.144
```

## 5. State advantages and disadvantages of multithreading ?

In [15]:

```
# Ans:-Advantagees of multithreading
#       Enhanced performance by decreased development time
#       Simplified and streamlined program coding
#       Improvised GUI responsiveness
#        Simultaneous and parallelized occurrence of tasks
#         Better use of cache storage by utilization of resources
#         Decreased cost of maintenance
#         Better use of CPU resource

#    Disadvanntages of multithreading
#        Complex debugging and testing processes
#        Overhead switching of context
#       Increased potential for deadlock occurrence
#       Increased difficulty level in writing a program
#       Unpredictable results
```

## Q6.What is Deadlock and Race Condition ?

In [16]:

```
#Ans-Race Condition And Deadlock
# A race condition occurs when two threads use the same variable at a given time.
# Deadlock exists when two threads seek one lock simultaneously.
# This situation will stop both threads from processing or executing the functions.
```

In [ ]: