|  |
| --- |
| Experiment No. 12 |
| Demonstrate the concept of Multi-threading |
| Date of Performance: |
| Date of Submission: |

**Experiment No. 12**

**Title:** Demonstrate the concept of Multi-threading

**Aim:** To study and implement the concept of Multi-threading

**Objective:** To introduce the concept of Multi-threading in python

**Theory:**

**Thread**

In computing, a **process** is an instance of a computer program that is being executed. Any process has 3 basic components:

* An executable program.
* The associated data needed by the program (variables, work space, buffers, etc.)
* The execution context of the program (State of process)

A **thread** is an entity within a process that can be scheduled for execution. Also, it is the smallest unit of processing that can be performed in an OS (Operating System).

In simple words, a **thread** is a sequence of such instructions within a program that can be executed independently of other code. For simplicity, you can assume that a thread is simply a subset of a process!

A thread contains all this information in a **Thread Control Block (TCB)**:

* **Thread Identifier:** Unique id (TID) is assigned to every new thread
* **Stack pointer:** Points to thread’s stack in the process. Stack contains the local variables under thread’s scope.
* **Program counter:** a register which stores the address of the instruction currently being executed by thread.
* **Thread state:** can be running, ready, waiting, start or done.
* **Thread’s register set:** registers assigned to thread for computations.
* **Parent process Pointer:** A pointer to the Process control block (PCB) of the process that the thread lives on.

**Code:**

import threading

def print\_cube(num):

print("Cube: {}".format(num \* num \* num))

def print\_square(num):

print("Square: {}".format(num \* num))

if \_\_name\_\_ == "\_\_main\_\_":

t1 = threading.Thread(target=print\_square, args=(10,))

t2 = threading.Thread(target=print\_cube, args=(10,))

t1.start()

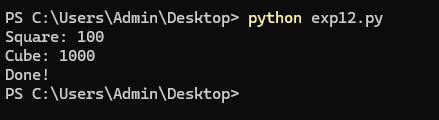
t2.start()

t1.join()

t2.join()

print("Done!")

**Output:**

****

**Conclusion:** The provided Python code demonstrates the implementation of multi-threading using the ‘threading’ module. Multi-threading is a programming concept where multiple threads (subprocesses) run concurrently within a single process, allowing for better resource utilization and improved performance, especially in tasks involving I/O operations or parallelizable computations.