Exercises:

1. Modify Example 1 to display strings via two independent threads:

thread1: “Hello! StudentName ”,

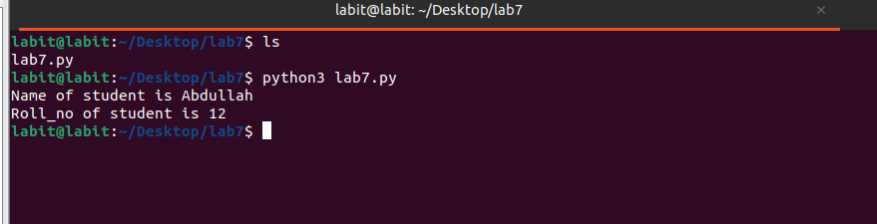
thread2: “Student roll no is: ”

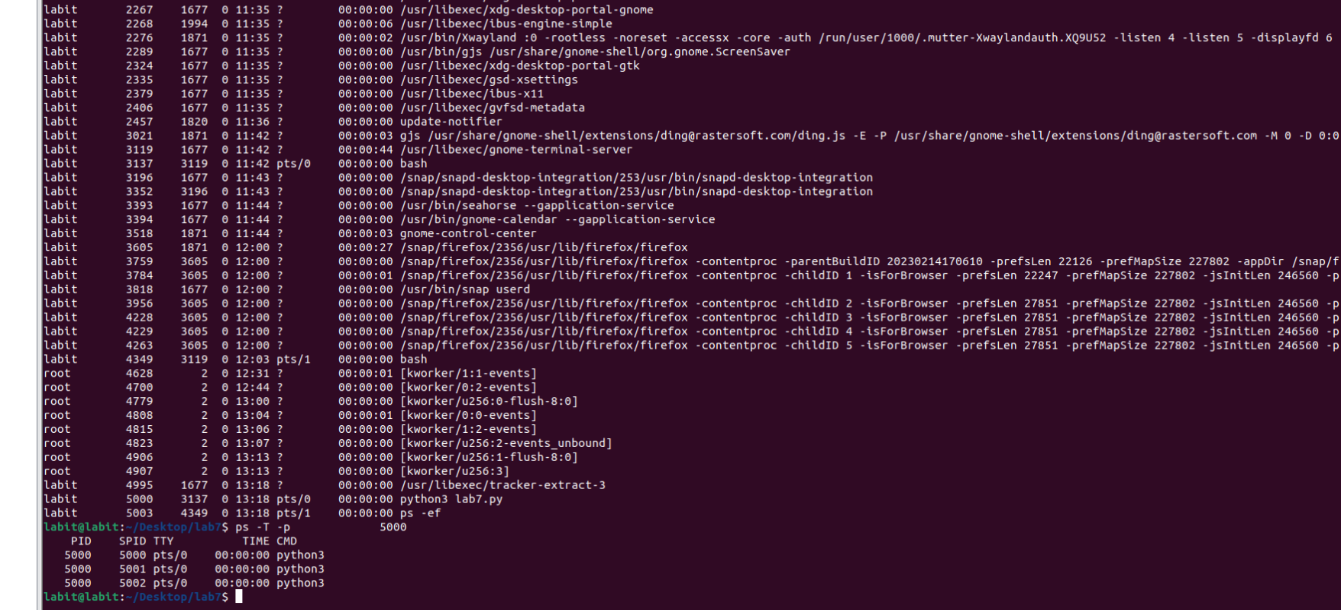
Ans:

Lab7.py

|  |
| --- |
| import threading  import time  def student\_name(name):  print("Name of student is", name)  time.sleep(30)  def student\_id(id):  print("Roll\_no of student is", id)  time.sleep(30)  t1 = threading.Thread(target=student\_name, args=("Abdullah",))  t2 = threading.Thread(target=student\_id, args=("12",))  t1.start()  t2.start()  t1.join()  t2.join() |

OUTPUT:





2. Write a Python program that performs basic arithmetic operations (addition, subtraction,

multiplication, and division) using multithreading. Each operation should be handled by a separate,

independent thread, allowing the calculations to be executed concurrently.

Ans:

Cal.py

|  |
| --- |
| import threading  import time  def add(a, b):  print(f"Addition of {a} and {b} is: {a + b}")  time.sleep(30)  def subtract(a, b):  print(f"Subtraction of {a} and {b} is: {a - b}")  time.sleep(30)  def multiply(a, b):  print(f"Multiplication of {a} and {b} is: {a \* b}")  time.sleep(30)  def divide(a, b):  if b != 0:  print(f"Division of {a} by {b} is: {a / b}")  else:  print("Cannot divide by zero.")  time.sleep(30)  a = 10  b = 5  t1 = threading.Thread(target=add, args=(a, b))  t2 = threading.Thread(target=subtract, args=(a, b))  t3 = threading.Thread(target=multiply, args=(a, b))  t4 = threading.Thread(target=divide, args=(a, b))  t1.start()  t2.start()  t3.start()  t4.start()  t1.join()  t2.join()  t3.join()  t4.join()  print("All arithmetic operations completed.") |

OUTPUT:

