IARE

IARE
INSTITUTE OF AERONAUTICAL ENGINEERING
(An Autonomous Institute affiliated to JNTUH, Hyderabad)
Dundigal, Hyderabad - 500 043

LABORATORY WORK SHEET

Name of the Student : Muhd. Fauzan zohaib

Class : B.tech CSE Semester : III

Roll Number									
2	3	9	5	1	A	0	5	5	H

Course Code : ACS010 Course Name : OS Laboratory

Name of the Course Faculty : Mr. M Hari krishna Faculty ID : IARE10836

Exercise Number : 01 Week Number : 01 Date :

DAY TO DAY EVALUATION:

Marks	Aim / Preparation	Algorithm / Procedure	Source Code	Program Execution	Viva - Voce	Total
		Performance in the Lab	Calculations and Graphs	Results and Error Analysis		
Max. Marks	4	4	4	4	4	20
Obtained	4	4	4	4	4	20

Signature of Faculty

START WRITING FROM HERE :

1. Getting started exercises

1.1 The Busy printer

Code :-

```
def calculateTotalTime(arrivalTimes, pages):
    currentTime = 0
    totalTime = 0
    for i in range(len(arrivalTimes)):
        if(arrivalTimes[i] != i):
            totalTime += (arrivalTimes[i] - arrivalTimes[i-1])-1)
    totalTime = totalTime + pages[i]
```

ROLL NUMBER :

```
    return totalTime

arrivalTimes = [0, 1, 3, 5]
pages = [10, 5, 3, 7]
totalTime = calculateTotalTime(arrivalTimes, pages)
print(f"The total time taken to complete all printing tasks
                is : {totalTime} seconds")
```

Output:-

The total time taken to complete all printing tasks is : 27 seconds.

1.2 The Software Developer's Tasks

code:-

```
def calculate_total_time(execution_times):
    total_time = sum(execution_times)
    return total_time

execution_times = [3,5,7,4]
total_time = calculate_total_time(execution_times)
print(f" The total time required to complete all tasks is:
                {total_time} hours ")
```

output:-

The total time required to complete all tasks is : 19 hours

ROLL NUMBER :

1.3 Managing a Restaurant's Orders

code :-

```
from collections import deque

def calculate_total_time (orders, time_quantum):
    queue = deque(orders)
    total_time = 0
    while queue:
        current_order = queue.popleft()
        if current_order > time_quantum:
            total_time += time_quantum
            remaining_order_time = current_order - time_quantum
            queue.append(remaining_order_time)
        else:
            total_time += current_order
    return total_time

orders = [5, 3, 8, 6]
time_quantum = 4
total_time = calculate_total_time(orders, time_quantum)
print(f"The total time required to complete all order is : {total_time} minutes.")
```

output :-

The total time required to complete all order is : 22 minutes.

ROLL NUMBER :

1.4 Emergency room prioritization

Code :-

```
from queue import priority Queue

def calculate_total_time (patients):
    pq = Priority Queue ()
    for priority, treatment_time in patients:
        pq.put ((priority, treatment_time))
    total_time = 0
    while not pq.empty()
        _, treatment_time = pq.get ()
        total_time += treatment_time
    return total_time.

patients = [(1, 10), (2, 8), (3, 15), (4, 5)]
total_time = calculate_total_time (patients)
print (f" The total time required to treat all patients is:
                                    {total_time} minutes ")
```

Output :-

The total time required to treat all patients is :
38 minutes