CSI 3001- CLOUD COMPUTING METHODOLOGIES

LAB ASSESSMENT 3

NAME: VIGNESH S

REGNO: 22MIC0152

QUESTION 1-I:

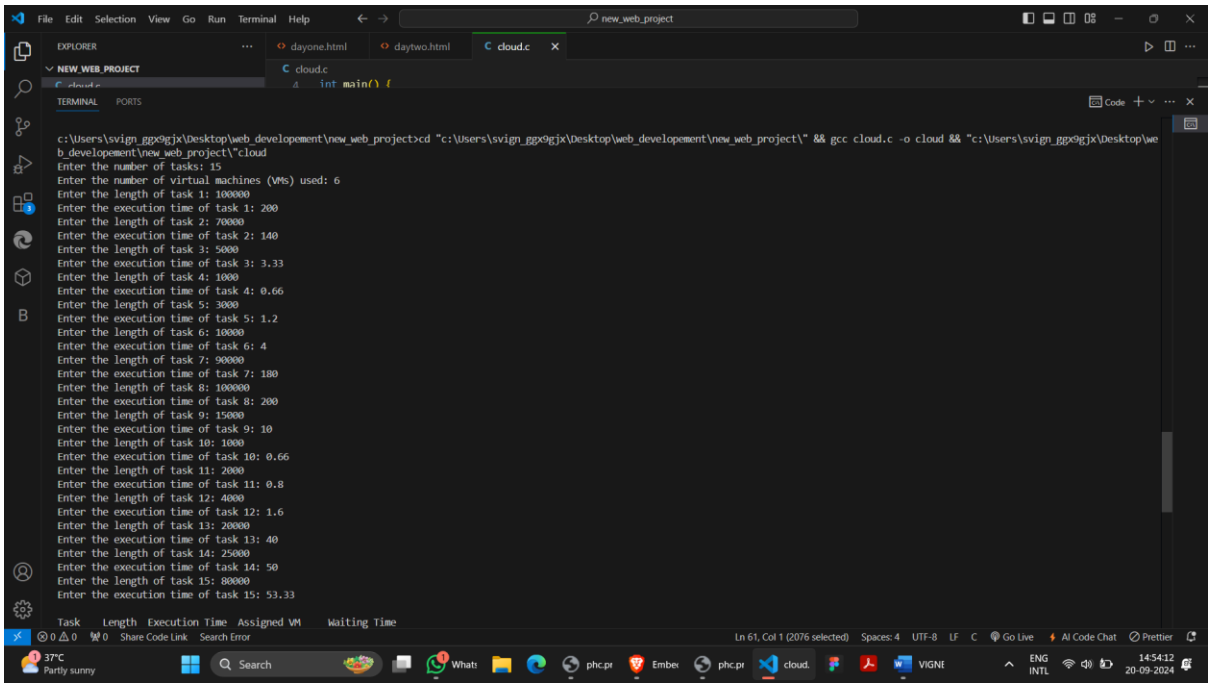
In a cloud computing environment, a data center provides virtual machines (VMs) to serve various tasks that are submitted by users. Consider a scenario where 8 tasks, T1, T2, T3, T4, T5, T6, T7, and T8, arrive in a queue and are assigned to 3 virtual machines (VM1, VM2, and VM3) based on a First-Come, First-Served (FCFS) scheduling algorithm. Each task has a specific length (in seconds) that denotes the time required to complete it. Compute total waiting time and average waiting time.

CODE:

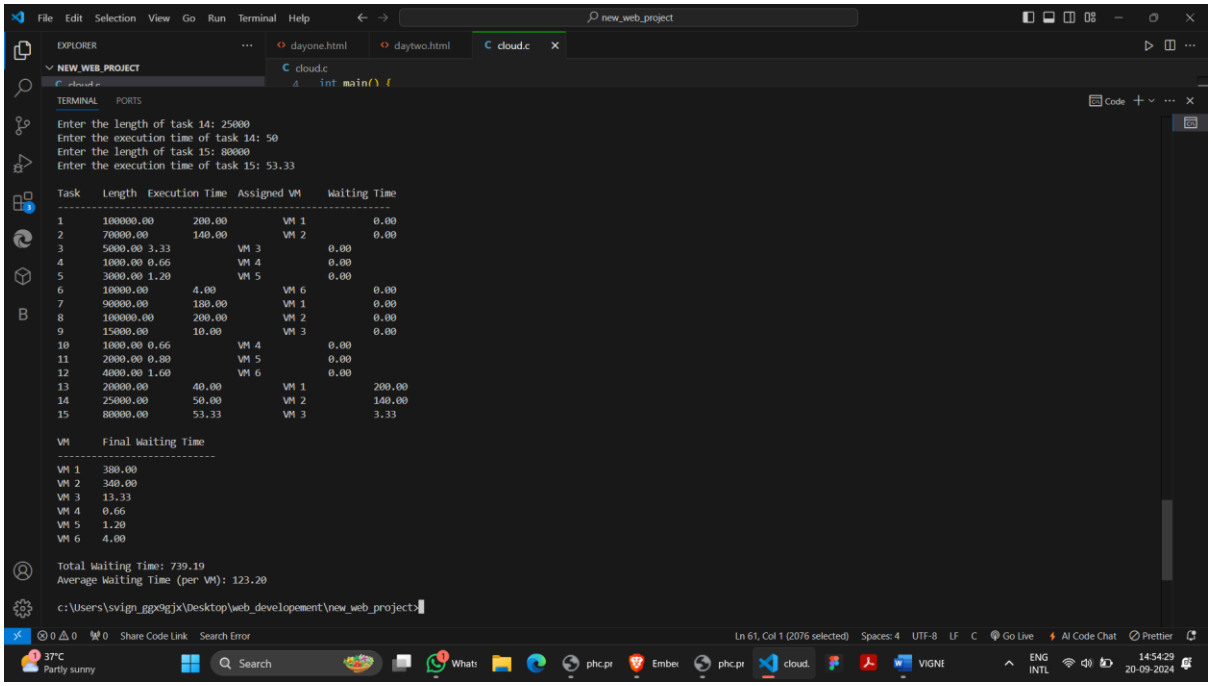
```
#include <stdio.h>

#include <stdlib.h>


int main() {
   int num_tasks, num_vms;
   printf("Enter the number of tasks: ");
   scanf("%d", &num_tasks);
   printf("Enter the number of virtual machines (VMs) used: ");
   scanf("%d", &num_vms);
   double *task_length = (double *)malloc(num_tasks * sizeof(double));
   double *execution_time = (double *)malloc(num_tasks * sizeof(double));
   double *vm_waiting_time = (double *)malloc(num_vms * sizeof(double));
   double *vm_last_task_end = (double *)malloc(num_vms * sizeof(double));
   for (int i = 0; i < num_vms; i++) {
      vm_waiting_time[i] = 0;
      vm_last_task_end[i] = 0;
   }
   for (int i = 0; i < num_tasks; i++) {
      printf("Enter the length of task %d: ", i + 1);
      scanf("%lf", &task_length[i]);
      printf("Enter the execution time of task %d: ", i + 1);
      scanf("%lf", &execution_time[i]);
   }
```

```
    printf("\nTask\tLength\tExecution Time\tAssigned VM\tWaiting Time\n");
    printf("-------------------------------------------------------\n");
    for (int i = 0; i < num_tasks; i++) {
        int vm_index = i % num_vms;
        printf("%d\t%.2lf\t%.2lf\t\tVM %d\t\t%.2lf\n", i + 1, task_length[i], execution_time[i],
vm_index + 1, vm_waiting_time[vm_index]);
        vm_waiting_time[vm_index] += vm_last_task_end[vm_index];
        vm_last_task_end[vm_index] = execution_time[i];
    }
    printf("\nVM\tFinal Waiting Time\n");
    printf("---------------------------\n");
    double total_waiting_time = 0;
    for (int i = 0; i < num_vms; i++) {
        printf("VM %d\t%.2lf\n", i + 1, vm_waiting_time[i]);
        total_waiting_time += vm_waiting_time[i];
    }
    double average_waiting_time = total_waiting_time / num_vms;
    printf("\nTotal Waiting Time: %.2lf\n", total_waiting_time);
    printf("Average Waiting Time (per VM): %.2lf\n", average_waiting_time);
   free(task_length);
    free(execution_time);
    free(vm_waiting_time);
    free(vm_last_task_end);
    return 0;
}
```

OUTPUT 1:

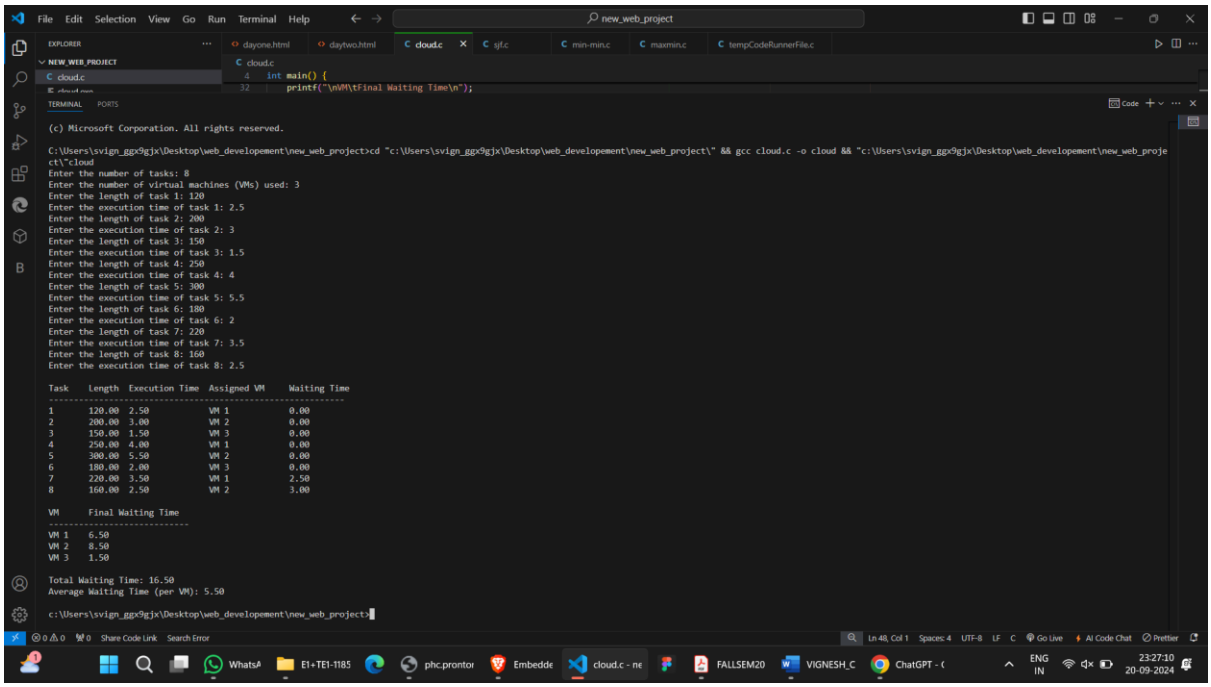


```
c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>cd "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\" && gcc cloud.c -o cloud && "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\"cloud
Enter the number of tasks: 15
Enter the number of virtual machines (VMs) used: 6
Enter the length of task 1: 100000
Enter the execution time of task 1: 200
Enter the length of task 2: 70000
Enter the execution time of task 2: 140
Enter the length of task 3: 5000
Enter the execution time of task 3: 3.33
Enter the length of task 4: 1000
Enter the execution time of task 4: 0.66
Enter the length of task 5: 3000
Enter the execution time of task 5: 1.2
Enter the length of task 6: 10000
Enter the execution time of task 6: 4
Enter the length of task 7: 90000
Enter the execution time of task 7: 180
Enter the length of task 8: 100000
Enter the execution time of task 8: 200
Enter the length of task 9: 15000
Enter the execution time of task 9: 10
Enter the length of task 10: 1000
Enter the execution time of task 10: 0.66
Enter the length of task 11: 2000
Enter the execution time of task 11: 0.8
Enter the length of task 12: 4000
Enter the execution time of task 12: 1.6
Enter the length of task 13: 20000
Enter the execution time of task 13: 40
Enter the length of task 14: 25000
Enter the execution time of task 14: 50
Enter the length of task 15: 80000
Enter the execution time of task 15: 53.33

Task    Length  Execution Time  Assigned VM     Waiting Time
```



```
Enter the length of task 14: 25000
Enter the execution time of task 14: 50
Enter the length of task 15: 80000
Enter the execution time of task 15: 53.33

Task    Length  Execution Time  Assigned VM     Waiting Time
------------------------------------------------------------
1       100000.00       200.00          VM 1            0.00
2       70000.00        140.00          VM 2            0.00
3       5000.00 3.33            VM 3            0.00
4       1000.00 0.66            VM 4            0.00
5       3000.00 1.20            VM 5            0.00
6       10000.00        4.00            VM 6            0.00
7       90000.00        180.00          VM 1            0.00
8       100000.00       200.00          VM 2            0.00
9       15000.00        10.00           VM 3            0.00
10      1000.00 0.66            VM 4            0.00
11      2000.00 0.80            VM 5            0.00
12      4000.00 1.60            VM 6            0.00
13      20000.00        40.00           VM 1            200.00
14      25000.00        50.00           VM 2            140.00
15      80000.00        53.33           VM 3            3.33

VM      Final Waiting Time
--------------------------
VM 1    380.00
VM 2    340.00
VM 3    13.33
VM 4    0.66
VM 5    1.20
VM 6    4.00

Total Waiting Time: 739.19
Average Waiting Time (per VM): 123.20

c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>
```

OUTPUT 2:


File Edit Selection View Go Run Terminal Help
new_web_project
EXPLORER
NEW_WEB_PROJECT
cloud.c
dayone.html
daytwo.html
cloud.c
sjf.c
min-min.c
maxmin.c
tempCodeRunnerFile.c
int main() {
printf("\nVM\tFinal Waiting Time\n");
TERMINAL
PORTS
(c) Microsoft Corporation. All rights reserved.
C:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>cd "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\" && gcc cloud.c -o cloud && "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\"cloud
Enter the number of tasks: 8
Enter the number of virtual machines (VMs) used: 3
Enter the length of task 1: 120
Enter the execution time of task 1: 2.5
Enter the length of task 2: 200
Enter the execution time of task 2: 3
Enter the length of task 3: 150
Enter the execution time of task 3: 1.5
Enter the length of task 4: 250
Enter the execution time of task 4: 4
Enter the length of task 5: 300
Enter the execution time of task 5: 5.5
Enter the length of task 6: 180
Enter the execution time of task 6: 2
Enter the length of task 7: 220
Enter the execution time of task 7: 3.5
Enter the length of task 8: 160
Enter the execution time of task 8: 2.5
Task Length Execution Time Assigned VM Waiting Time
1 120.00 2.50 VM 1 0.00
2 200.00 3.00 VM 2 0.00
3 150.00 1.50 VM 3 0.00
4 250.00 4.00 VM 1 0.00
5 300.00 5.50 VM 2 0.00
6 180.00 2.00 VM 3 0.00
7 220.00 3.50 VM 1 2.50
8 160.00 2.50 VM 2 3.00
VM Final Waiting Time
VM 1 6.50
VM 2 8.50
VM 3 1.50
Total Waiting Time: 16.50
Average Waiting Time (per VM): 5.50
c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>

QUESTION 1-II:

A cloud service provider is managing a set of virtual machines (VMs) to handle usersubmitted tasks. The system employs Shortest Job First (SJF) scheduling to optimize task processing. There are 3 VMs available (VM1, VM2, and VM3), and 8 tasks (T1, T2, T3, T4, T5, T6, T7, and T8) with varying execution lengths, which arrive at different times. The tasks need to be assigned to the VMs based on their lengths, with shorter tasks being executed first. Tasks are non-preemptive, meaning once a task begins executing on a VM, it cannot be interrupted until completion. The arrival time for the tasks is ignored, and they are considered ready for execution at the same time. The tasks are assigned to VMs based on availability and their length. Compute total waiting time and average waiting time.

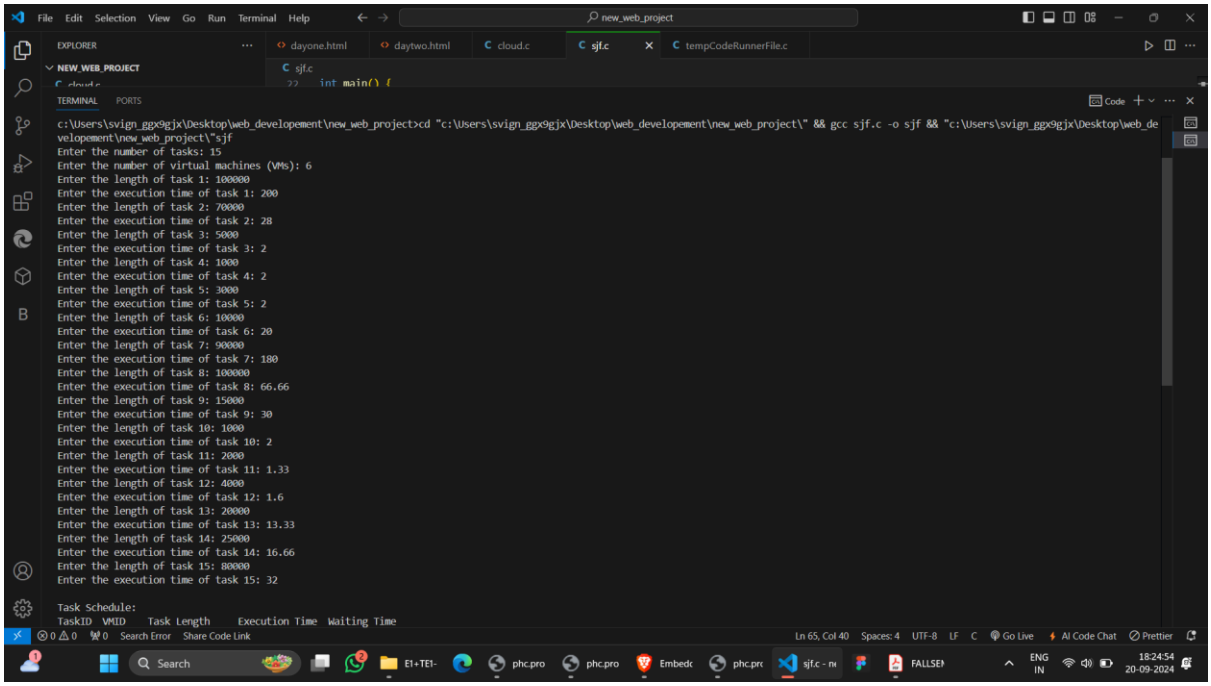
CODE:

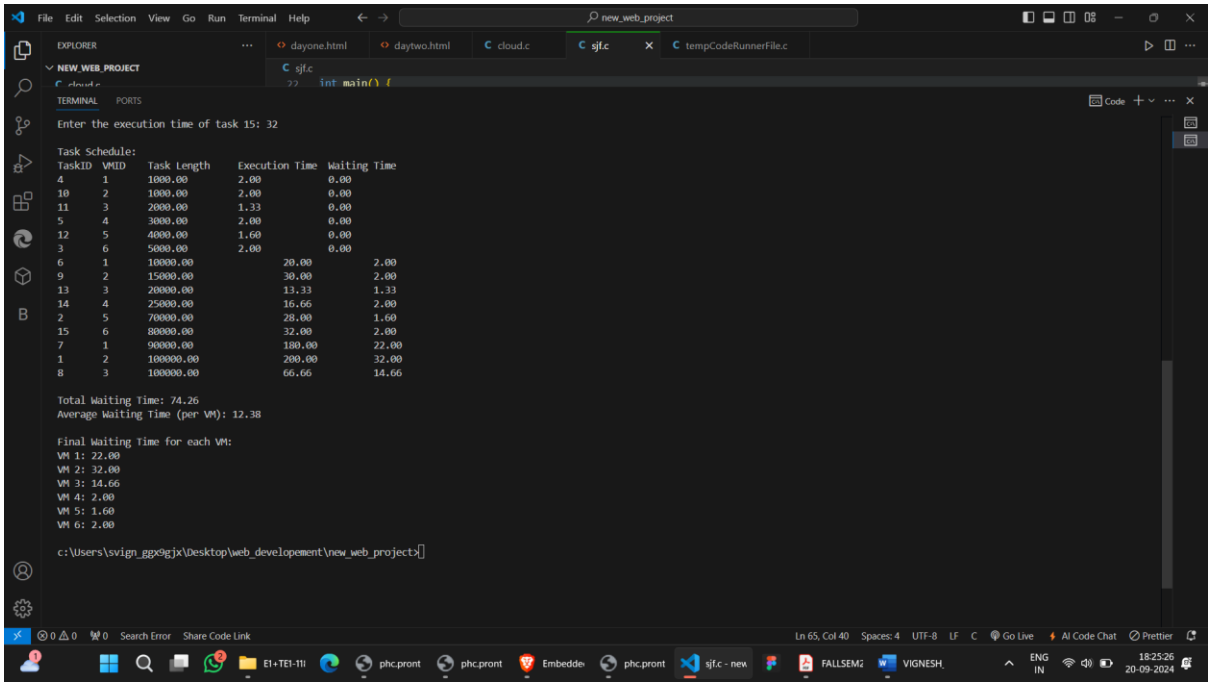
```
#include <stdio.h>
struct Task {
    int taskId;
    float length;
    float executionTime;
    int vmId;
};
void sortTasksByLength(struct Task tasks[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - 1 - i; j++) {
            if (tasks[j].length > tasks[j + 1].length) {
                struct Task temp = tasks[j];
                tasks[j] = tasks[j + 1];
                tasks[j + 1] = temp;
            }
        }
    }
}
int main() {
    int n, vmCount;
    printf("Enter the number of tasks: ");
    scanf("%d", &n);
    printf("Enter the number of virtual machines (VMs): ");
    scanf("%d", &vmCount);
    struct Task tasks[n];
    float vmWaitingTime[vmCount];
```

```
    float previousExecutionTime[vmCount];
    for (int i = 0; i < vmCount; i++) {
        vmWaitingTime[i] = 0.0;
        previousExecutionTime[i] = 0.0;
    }
    for (int i = 0; i < n; i++) {
        tasks[i].taskId = i + 1;
        printf("Enter the length of task %d: ", i + 1);
        scanf("%f", &tasks[i].length);
        printf("Enter the execution time of task %d: ", i + 1);
        scanf("%f", &tasks[i].executionTime);
    }
    sortTasksByLength(tasks, n);
    printf("\nTask Schedule:\n");
    printf("TaskID\tVMID\tTask Length\tExecution Time\tWaiting Time\n");
    for (int i = 0; i < n; i++) {
        int vmId = i % vmCount;
        tasks[i].vmId = vmId + 1;
        float waitingTime = (previousExecutionTime[vmId] > 0) ? vmWaitingTime[vmId] +
previousExecutionTime[vmId] : 0;
        printf("%d\t%d\t%.2f\t\t%.2f\t\t%.2f\n", tasks[i].taskId, tasks[i].vmId, tasks[i].length,
tasks[i].executionTime, waitingTime);
        vmWaitingTime[vmId] += previousExecutionTime[vmId];
        previousExecutionTime[vmId] = tasks[i].executionTime;
    }
    float totalWaitingTime = 0.0;
    for (int i = 0; i < vmCount; i++) {
        totalWaitingTime += vmWaitingTime[i];
    }
    float averageWaitingTime = totalWaitingTime / vmCount;
    printf("\nTotal Waiting Time: %.2f\n", totalWaitingTime);
    printf("Average Waiting Time (per VM): %.2f\n", averageWaitingTime);


    printf("\nFinal Waiting Time for each VM:\n");
```

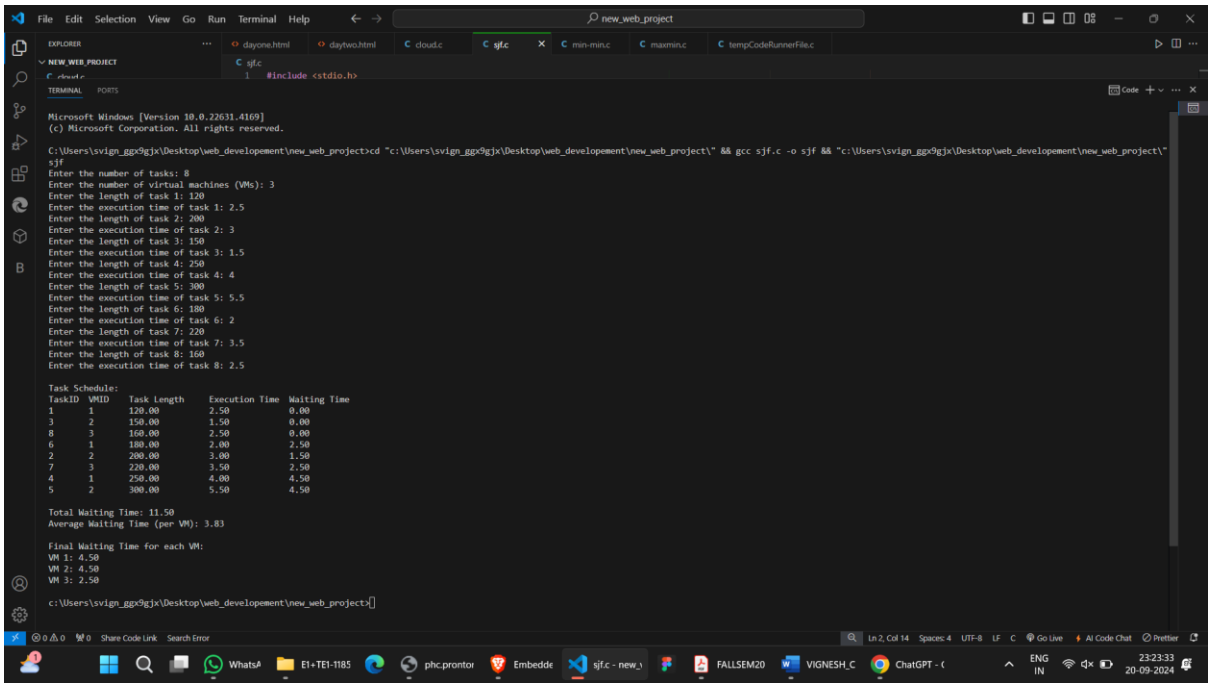
```
    for (int i = 0; i < vmCount; i++) {
        printf("VM %d: %.2f\n", i + 1, vmWaitingTime[i]);
    }
}
```

OUTPUT 1:





OUTPUT 2:


File Edit Selection View Go Run Terminal Help
new_web_project
EXPLORER
NEW_WEB_PROJECT
dayone.html
daytwo.html
cloud.c
sjf.c
min-min.c
maxmin.c
tempCodeRunnerFile.c
#include <stdio.h>
TERMINAL
PORTS
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.
C:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>cd "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\" && gcc sjf.c -o sjf && "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\"
sjf
Enter the number of tasks: 8
Enter the number of virtual machines (VMs): 3
Enter the length of task 1: 120
Enter the execution time of task 1: 2.5
Enter the length of task 2: 200
Enter the execution time of task 2: 3
Enter the length of task 3: 150
Enter the execution time of task 3: 1.5
Enter the length of task 4: 250
Enter the execution time of task 4: 4
Enter the length of task 5: 300
Enter the execution time of task 5: 5.5
Enter the length of task 6: 180
Enter the execution time of task 6: 2
Enter the length of task 7: 220
Enter the execution time of task 7: 3.5
Enter the length of task 8: 160
Enter the execution time of task 8: 2.5
Task Schedule:
TaskID VMID Task Length Execution Time Waiting Time
1 1 120.00 2.50 0.00
3 2 150.00 1.50 0.00
8 3 160.00 2.50 0.00
6 1 180.00 2.00 2.50
2 2 200.00 3.00 1.50
7 3 220.00 3.50 2.50
4 1 250.00 4.00 4.50
5 2 300.00 5.50 4.50
Total Waiting Time: 11.50
Average Waiting Time (per VM): 3.83
Final Waiting Time for each VM:
VM 1: 4.50
VM 2: 4.50
VM 3: 2.50
c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>
Share Code Link
Search Error
Ln 2, Col 14
Spaces: 4
UTF-8
LF
C
Go Live
AI Code Chat
Prettier
ENG
IN
23:23:33
20-09-2024

QUESTION 2:

A cloud computing environment has multiple Virtual Machines (VMs) available to process a set of tasks submitted by users. Each task has an associated execution time, which represents the time required for the task to complete once it starts on a VM. The system uses two distinct scheduling strategies, Max-Min and Min-Min, to assign tasks to VMs. In both scheduling strategies, the tasks are assigned to the VMs based on their execution times, but the selection process differs. Compute overall completion time

CODE:

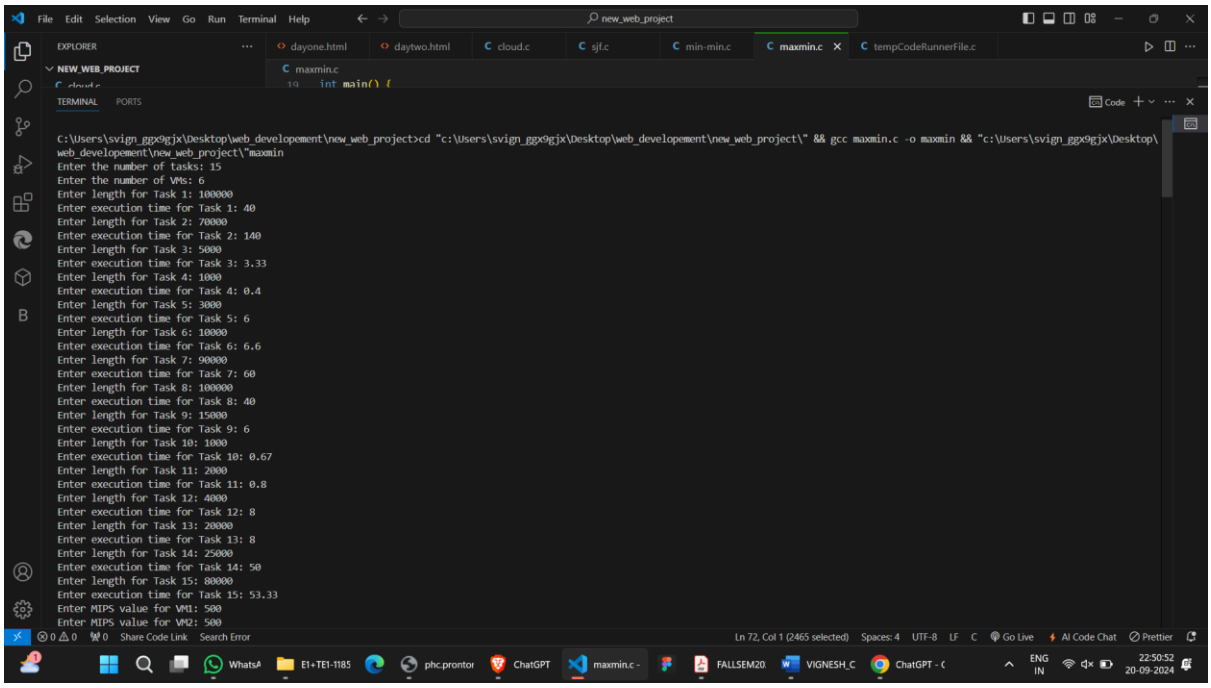
MAX-MIN:

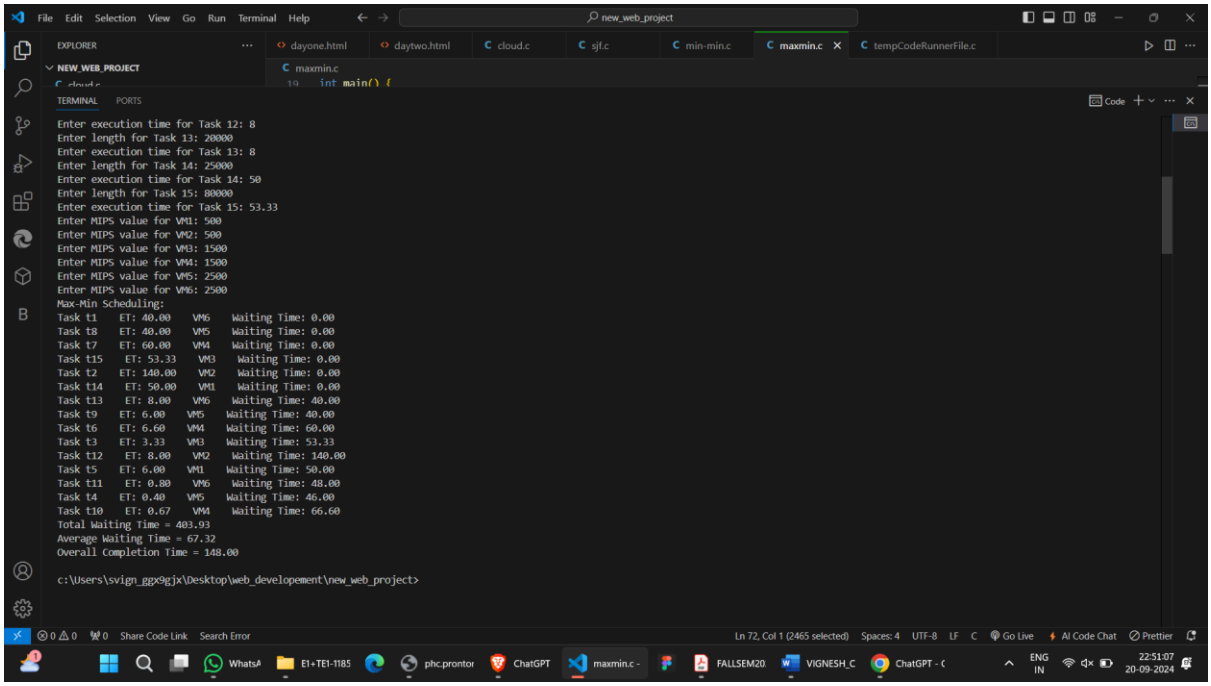
```
#include <stdio.h>
#include <stdlib.h>
typedef struct {
    int id;
    double length;
    double et;
    double waitingTime;
} Task;
int compareTasks(const void* a, const void* b) {
    Task* taskA = (Task*)a;
    Task* taskB = (Task*)b;
    return (taskB->length > taskA->length) - (taskB->length < taskA->length);
}
int compareVMs(const void* a, const void* b) {
    int* vmA = (int*)a;
    int* vmB = (int*)b;
    return (*vmB - *vmA);
}
int main() {
    int numTasks, numVMs;
    printf("Enter the number of tasks: ");
    scanf("%d", &numTasks);
    printf("Enter the number of VMs: ");
    scanf("%d", &numVMs);

    Task* tasks = (Task*)malloc(numTasks * sizeof(Task));
    double* vmMips = (double*)malloc(numVMs * sizeof(double));
```

```
    double* vmCompletionTimes = (double*)calloc(numVMs, sizeof(double));
    int* vmIndices = (int*)malloc(numVMs * sizeof(int));
    for (int i = 0; i < numTasks; i++) {
        printf("Enter length for Task %d: ", i + 1);
        scanf("%lf", &tasks[i].length);
        printf("Enter execution time for Task %d: ", i + 1);
        scanf("%lf", &tasks[i].et);
        tasks[i].id = i + 1;
        tasks[i].waitingTime = 0;
    }
    for (int i = 0; i < numVMs; i++) {
        printf("Enter MIPS value for VM%d: ", i + 1);
        scanf("%lf", &vmMips[i]);
        vmIndices[i] = i;
    }
    qsort(tasks, numTasks, sizeof(Task), compareTasks);
    qsort(vmIndices, numVMs, sizeof(int), compareVMs);
    printf("Max-Min Scheduling:\n");
    double abb = 0;
    double overallCompletionTime = 0;
    for (int i = 0; i < numTasks; i++) {
        int vmIndex = vmIndices[i % numVMs];
        Task* task = &tasks[i];
        task->waitingTime = vmCompletionTimes[vmIndex];
        vmCompletionTimes[vmIndex] += task->et;
        printf("Task t%d    ET: %.2f    VM%d    Waiting Time: %.2f\n", task->id, task->et, vmIndex + 1,
task->waitingTime);
        if (i >= numTasks - numVMs) {
            abb += task->waitingTime;
        }
    }
    for (int i = 0; i < numVMs; i++) {
        if (vmCompletionTimes[i] > overallCompletionTime) {
            overallCompletionTime = vmCompletionTimes[i];
```

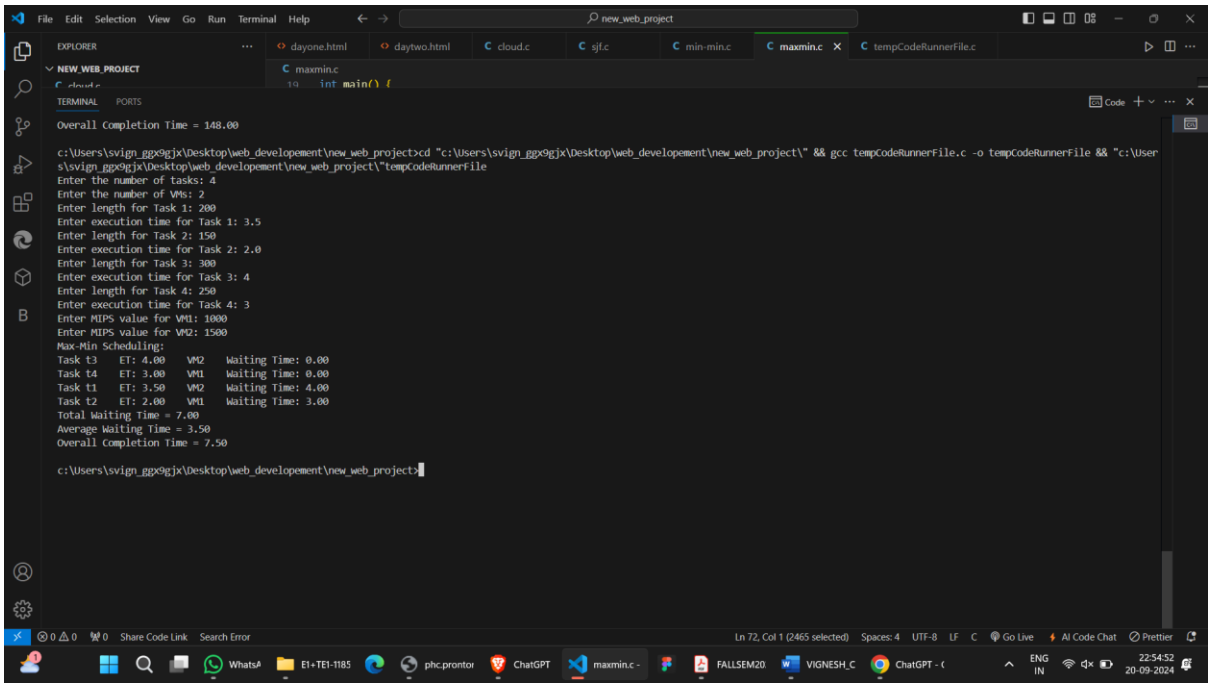
```
        }
    }
    double averageWaitingTime = abb / numVMs;
    printf("Total Waiting Time = %.2f\n", abb);
    printf("Average Waiting Time = %.2f\n", averageWaitingTime);
    printf("Overall Completion Time = %.2f\n", overallCompletionTime);
    free(tasks);
    free(vmMips);
    free(vmCompletionTimes);
    free(vmIndices);
}

}
```

OUTPUT 1:


C:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>cd "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\" && gcc maxmin.c -o maxmin && "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\"maxmin
Enter the number of tasks: 15
Enter the number of VMs: 6
Enter length for Task 1: 100000
Enter execution time for Task 1: 40
Enter length for Task 2: 70000
Enter execution time for Task 2: 140
Enter length for Task 3: 5000
Enter execution time for Task 3: 3.33
Enter length for Task 4: 1000
Enter execution time for Task 4: 0.4
Enter length for Task 5: 3000
Enter execution time for Task 5: 6
Enter length for Task 6: 10000
Enter execution time for Task 6: 6.6
Enter length for Task 7: 90000
Enter execution time for Task 7: 60
Enter length for Task 8: 100000
Enter execution time for Task 8: 40
Enter length for Task 9: 15000
Enter execution time for Task 9: 6
Enter length for Task 10: 1000
Enter execution time for Task 10: 0.67
Enter length for Task 11: 2000
Enter execution time for Task 11: 0.8
Enter length for Task 12: 4000
Enter execution time for Task 12: 8
Enter length for Task 13: 20000
Enter execution time for Task 13: 8
Enter length for Task 14: 25000
Enter execution time for Task 14: 50
Enter length for Task 15: 80000
Enter execution time for Task 15: 53.33
Enter MIPS value for VM1: 500
Enter MIPS value for VM2: 500


Enter execution time for Task 12: 8
Enter length for Task 13: 20000
Enter execution time for Task 13: 8
Enter length for Task 14: 25000
Enter execution time for Task 14: 50
Enter length for Task 15: 80000
Enter execution time for Task 15: 53.33
Enter MIPS value for VM1: 500
Enter MIPS value for VM2: 500
Enter MIPS value for VM3: 1500
Enter MIPS value for VM4: 1500
Enter MIPS value for VM5: 2500
Enter MIPS value for VM6: 2500
Max-Min Scheduling:
Task t1 ET: 40.00 VM6 Waiting Time: 0.00
Task t8 ET: 40.00 VM5 Waiting Time: 0.00
Task t7 ET: 60.00 VM4 Waiting Time: 0.00
Task t15 ET: 53.33 VM3 Waiting Time: 0.00
Task t2 ET: 140.00 VM2 Waiting Time: 0.00
Task t14 ET: 50.00 VM1 Waiting Time: 0.00
Task t13 ET: 8.00 VM6 Waiting Time: 40.00
Task t9 ET: 6.00 VM5 Waiting Time: 40.00
Task t6 ET: 6.60 VM4 Waiting Time: 60.00
Task t3 ET: 3.33 VM3 Waiting Time: 53.33
Task t12 ET: 8.00 VM2 Waiting Time: 140.00
Task t5 ET: 6.00 VM1 Waiting Time: 50.00
Task t11 ET: 0.80 VM6 Waiting Time: 48.00
Task t4 ET: 0.40 VM5 Waiting Time: 46.00
Task t10 ET: 0.67 VM4 Waiting Time: 66.60
Total Waiting Time = 403.93
Average Waiting Time = 67.32
Overall Completion Time = 148.00
c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>

OUTPUT 2:


Overall Completion Time = 148.00
c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>cd "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\"tempCodeRunnerFile
Enter the number of tasks: 4
Enter the number of VMs: 2
Enter length for Task 1: 200
Enter execution time for Task 1: 3.5
Enter length for Task 2: 150
Enter execution time for Task 2: 2.0
Enter length for Task 3: 300
Enter execution time for Task 3: 4
Enter length for Task 4: 250
Enter execution time for Task 4: 3
Enter MIPS value for VM1: 1000
Enter MIPS value for VM2: 1500
Max-Min Scheduling:
Task t3 ET: 4.00 VM2 Waiting Time: 0.00
Task t4 ET: 3.00 VM1 Waiting Time: 0.00
Task t1 ET: 3.50 VM2 Waiting Time: 4.00
Task t2 ET: 2.00 VM1 Waiting Time: 3.00
Total Waiting Time = 7.00
Average Waiting Time = 3.50
Overall Completion Time = 7.50
c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>

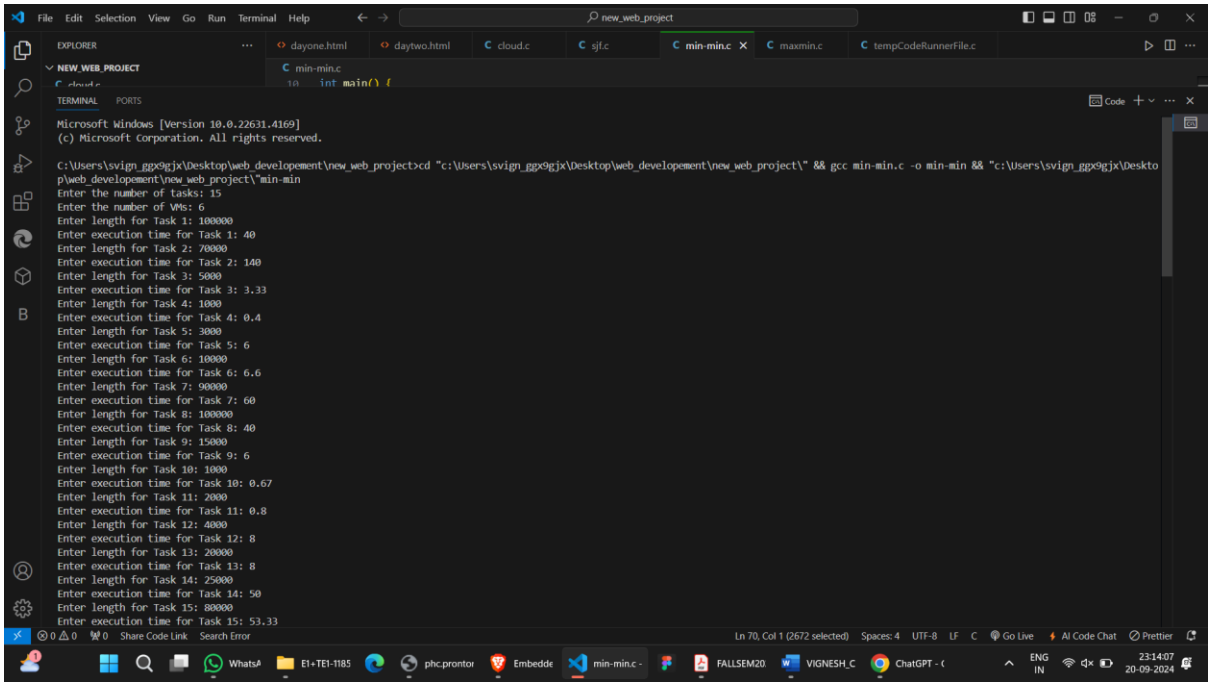
MIN-MIN:

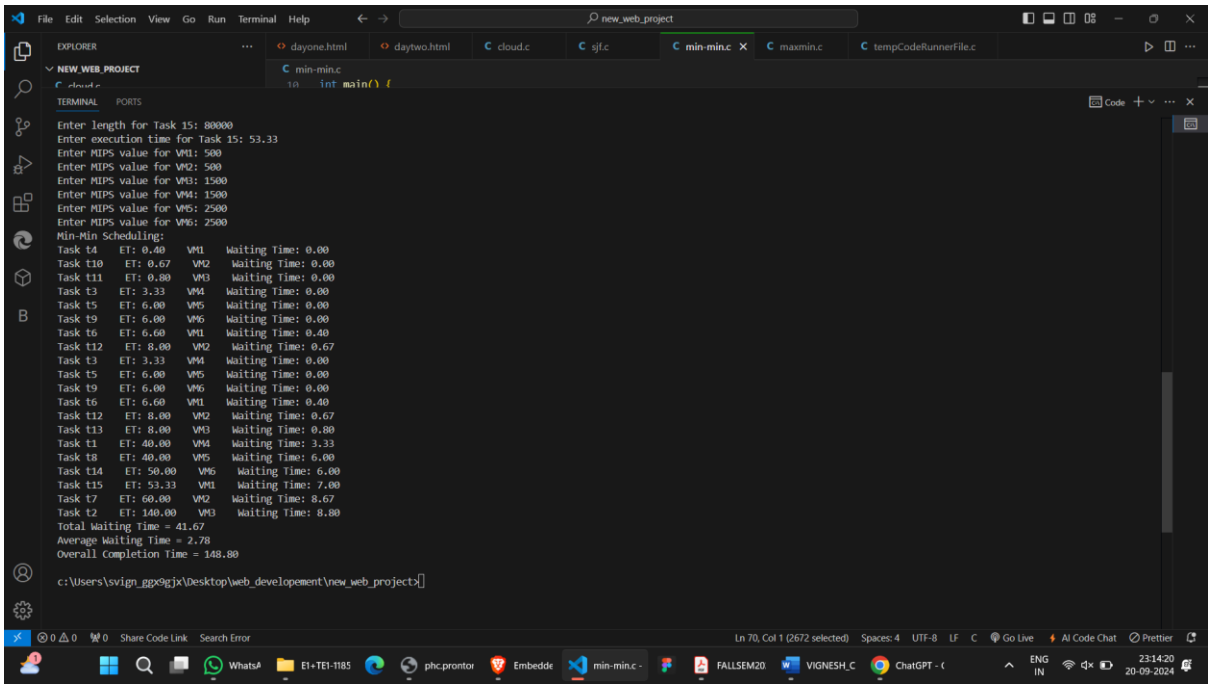
```
#include <stdio.h>
#include <stdlib.h>
typedef struct {
    int id;
    double length;
    double et;
    double waitingTime;
    int assignedVM;
} Task;
int main() {
    int numTasks, numVMs;
    printf("Enter the number of tasks: ");
    scanf("%d", &numTasks);
    printf("Enter the number of VMs: ");
    scanf("%d", &numVMs);
    Task* tasks = (Task*)malloc(numTasks * sizeof(Task));
    double* vmMips = (double*)malloc(numVMs * sizeof(double));
    double* vmCompletionTimes = (double*)calloc(numVMs, sizeof(double));
    for (int i = 0; i < numTasks; i++) {
        printf("Enter length for Task %d: ", i + 1);
        scanf("%lf", &tasks[i].length);
        printf("Enter execution time for Task %d: ", i + 1);
        scanf("%lf", &tasks[i].et);
        tasks[i].id = i + 1;
        tasks[i].waitingTime = 0;
        tasks[i].assignedVM = -1;
    }
    for (int i = 0; i < numVMs; i++) {
        printf("Enter MIPS value for VM%d: ", i + 1);
        scanf("%lf", &vmMips[i]);
    }
    printf("Min-Min Scheduling:\n");
    double overallCompletionTime = 0;
```

```
    double totalWaitingTime = 0;
    for (int i = 0; i < numTasks; i++) {
        int bestTaskIndex = -1;
        int bestVMIndex = -1;
        double bestCompletionTime = 1e9;
        for (int t = 0; t < numTasks; t++) {
            if (tasks[t].assignedVM == -1) {
                for (int v = 0; v < numVMs; v++) {
                    double completionTime = vmCompletionTimes[v] + tasks[t].et;
                    if (completionTime < bestCompletionTime) {
                        bestCompletionTime = completionTime;
                        bestTaskIndex = t;
                        bestVMIndex = v;
                    }
                }
            }
        }
        tasks[bestTaskIndex].assignedVM = bestVMIndex;
        tasks[bestTaskIndex].waitingTime = vmCompletionTimes[bestVMIndex];
        vmCompletionTimes[bestVMIndex] += tasks[bestTaskIndex].et;
        printf("Task t%d    ET: %.2f    VM%d    Waiting Time: %.2f\n", tasks[bestTaskIndex].id,
tasks[bestTaskIndex].et, bestVMIndex + 1, tasks[bestTaskIndex].waitingTime);
        totalWaitingTime += tasks[bestTaskIndex].waitingTime;
    }
    for (int i = 0; i < numVMs; i++) {
        if (vmCompletionTimes[i] > overallCompletionTime) {
            overallCompletionTime = vmCompletionTimes[i];
        }
    }
    double averageWaitingTime = totalWaitingTime / numTasks;
    printf("Total Waiting Time = %.2f\n", totalWaitingTime);
    printf("Average Waiting Time = %.2f\n", averageWaitingTime);
    printf("Overall Completion Time = %.2f\n", overallCompletionTime);
    free(tasks);
```

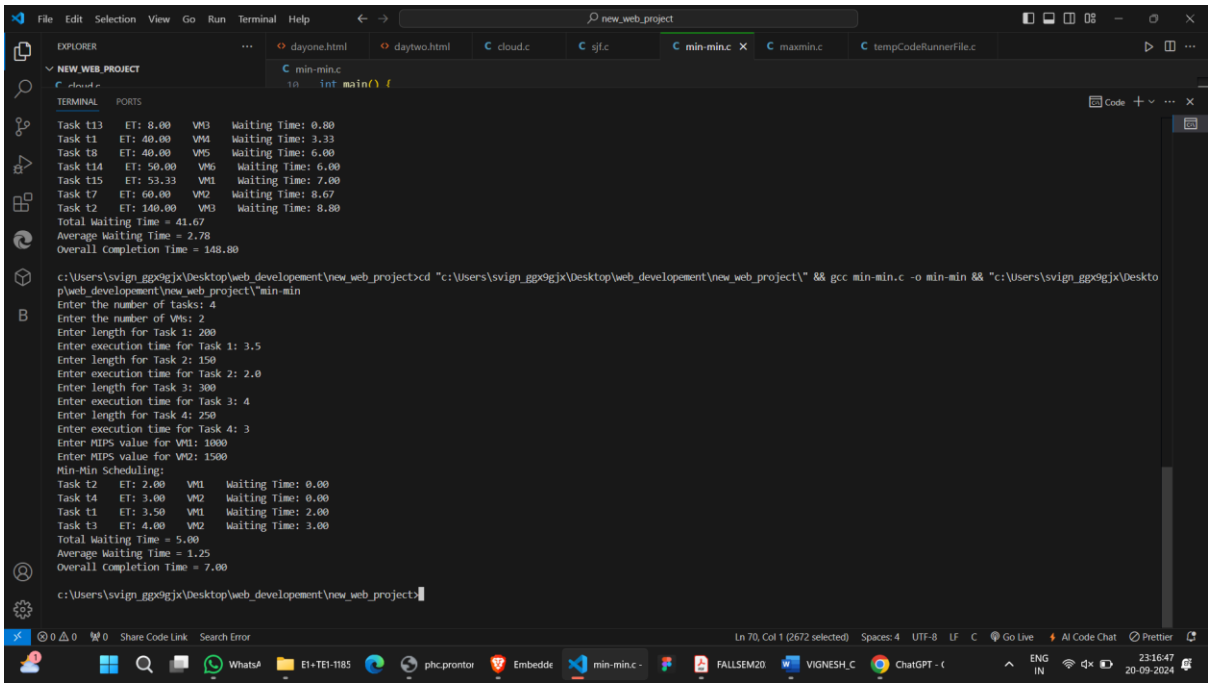
```
    free(vmMips);
    free(vmCompletionTimes);
}
```

OUTPUT 1:





OUTPUT 2:


Task t13 ET: 8.00 VM3 Waiting Time: 0.80
Task t1 ET: 40.00 VM4 Waiting Time: 3.33
Task t8 ET: 40.00 VM5 Waiting Time: 6.00
Task t14 ET: 50.00 VM6 Waiting Time: 6.00
Task t15 ET: 53.33 VM1 Waiting Time: 7.00
Task t7 ET: 60.00 VM2 Waiting Time: 8.67
Task t2 ET: 140.00 VM3 Waiting Time: 8.80
Total Waiting Time = 41.67
Average Waiting Time = 2.78
Overall Completion Time = 148.80
c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>cd "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\" && gcc min-min.c -o min-min && "c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project\"min-min
Enter the number of tasks: 4
Enter the number of VMs: 2
Enter length for Task 1: 200
Enter execution time for Task 1: 3.5
Enter length for Task 2: 150
Enter execution time for Task 2: 2.0
Enter length for Task 3: 300
Enter execution time for Task 3: 4
Enter length for Task 4: 250
Enter execution time for Task 4: 3
Enter MIPS value for VM1: 1000
Enter MIPS value for VM2: 1500
Min-Min Scheduling:
Task t2 ET: 2.00 VM1 Waiting Time: 0.00
Task t4 ET: 3.00 VM2 Waiting Time: 0.00
Task t1 ET: 3.50 VM1 Waiting Time: 2.00
Task t3 ET: 4.00 VM2 Waiting Time: 3.00
Total Waiting Time = 5.00
Average Waiting Time = 1.25
Overall Completion Time = 7.00
c:\Users\svign_ggx9gjx\Desktop\web_developement\new_web_project>