code:

#include <stdio.h>

#define MAX\_TASKS 10

typedef struct {

int id;

int period;

int executionTime;

int deadline;

} Task;

// Function to sort tasks based on period

void sortTasks(Task tasks[], int n) {

int i, j;

Task temp;

for (i = 0; i < n - 1; i++) {

for (j = 0; j < n - i - 1; j++) {

if (tasks[j].period > tasks[j + 1].period) {

temp = tasks[j];

tasks[j] = tasks[j + 1];

tasks[j + 1] = temp;

}

}

}

}

// Rate-Monotonic scheduling algorithm

void rateMonotonic(Task tasks[], int n) {

int currentTime = 0;

int i;

printf("\nRate-Monotonic Scheduling\n");

printf("Task\tDeadline\tExecution Time\tPeriod\n");

for (i = 0; i < n; i++) {

printf("%d\t%d\t\t%d\t\t%d\n", tasks[i].id, tasks[i].deadline, tasks[i].executionTime, tasks[i].period);

}

printf("\nExecution Order: ");

for (i = 0; i < n; i++) {

printf("%d ", tasks[i].id);

}

printf("\n\n");

for (i = 0; i < n; i++) {

currentTime += tasks[i].executionTime;

printf("Task %d completed at time %d\n", tasks[i].id, currentTime);

}

}

// Earliest-Deadline First scheduling algorithm

void earliestDeadlineFirst(Task tasks[], int n) {

int currentTime = 0;

int i, smallestDeadlineTaskIndex;

printf("\nEarliest-Deadline First Scheduling\n");

printf("Task\tDeadline\tExecution Time\tPeriod\n");

for (i = 0; i < n; i++) {

printf("%d\t%d\t\t%d\t\t%d\n", tasks[i].id, tasks[i].deadline, tasks[i].executionTime, tasks[i].period);

}

printf("\nExecution Order: ");

for (i = 0; i < n; i++) {

smallestDeadlineTaskIndex = 0;

for (int j = 1; j < n; j++) {

if (tasks[j].deadline < tasks[smallestDeadlineTaskIndex].deadline) {

smallestDeadlineTaskIndex = j;

}

}

printf("%d ", tasks[smallestDeadlineTaskIndex].id);

currentTime += tasks[smallestDeadlineTaskIndex].executionTime;

tasks[smallestDeadlineTaskIndex].deadline += tasks[smallestDeadlineTaskIndex].period;

}

printf("\n\n");

for (i = 0; i < n; i++) {

printf("Task %d completed at time %d\n", tasks[i].id, currentTime);

}

}

// Proportional scheduling algorithm

void proportionalScheduling(Task tasks[], int n) {

int currentTime = 0;

int i;

printf("\nProportional Scheduling\n");

printf("Task\tDeadline\tExecution Time\tPeriod\n");

for (i = 0; i < n; i++) {

printf("%d\t%d\t\t%d\t\t%d\n", tasks[i].id, tasks[i].deadline, tasks[i].executionTime, tasks[i].period);

}

printf("\nExecution Order: ");

for (i = 0; i < n; i++) {

printf("%d ", tasks[i].id);

}

printf("\n\n");

for (i = 0; i < n; i++) {

currentTime += tasks[i].executionTime;

printf("Task %d completed at time %d\n", tasks[i].id, currentTime);

}

}

int main() {

Task tasks[MAX\_TASKS];

int n, i;

printf("Enter the number of tasks (max %d): ", MAX\_TASKS);

scanf("%d", &n);

printf("\nEnter the details for each task:\n");

for (i = 0; i < n; i++) {

printf("Task %d\n", i + 1);

tasks[i].id = i + 1;

printf("Enter the period: ");

scanf("%d", &tasks[i].period);

printf("Enter the execution time: ");

scanf("%d", &tasks[i].executionTime);

tasks[i].deadline = tasks[i].period;

}

sortTasks(tasks, n);

rateMonotonic(tasks, n);

earliestDeadlineFirst(tasks, n);

proportionalScheduling(tasks, n);

return 0;

}

Output:

