#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <unistd.h>

#include <ctype.h>

#define max 5

struct TASK

{

int TaskId;

char TaskTitle[100];

int TaskDuration;

char Status[100];

};

struct TASK task[10] = {

{532, "ew34s", 3, "Idle"},

{2352, "erw53r", 3, "Idle"},

{325, "hg43gw", 3, "Idle"},

{235, "fr33es", 3, "Idle"},

{123, "f33fss", 3, "Idle"},

{324532, "fad32g", 3, "Idle"},

{12414, "ff43sd", 3, "Idle"},

{242, "gd34fa", 3, "Idle"},

{1444, "fa34fa", 3, "Idle"},

{414, "ga41ag", 3, "Idle"}};

int taskcount = 10;

struct TASK queue[max];

int front = -1, rear = -1;

void enqueue(struct TASK task)

{

if (rear == max - 1)

{

printf("\nQueue is full\n");

// calculate wait time

int waitTimeMin = queue[front].TaskDuration, waitTimeMax = 0;

for (int i = front; i <= rear; i++)

{

waitTimeMax += queue[i].TaskDuration;

}

printf("Wait time is between %d and %d seconds\n", waitTimeMin, waitTimeMax);

}

else if (front == -1 && rear == -1)

{

front = rear = 0;

queue[rear].TaskDuration = task.TaskDuration;

queue[rear].TaskId = task.TaskId;

strcpy(queue[rear].TaskTitle, task.TaskTitle);

strcpy(queue[rear].Status, "Queued");

}

else

{

rear++;

queue[rear].TaskDuration = task.TaskDuration;

queue[rear].TaskId = task.TaskId;

strcpy(queue[rear].TaskTitle, task.TaskTitle);

strcpy(queue[rear].Status, "Queued");

}

}

void dequeue()

{

if (front == -1 && rear == -1 || front == rear + 1)

{

printf("\nQueue is empty\n");

}

else

{

printf("Task %d is running\n", queue[front].TaskId);

sleep(queue[front].TaskDuration);

printf("Task %d is completed\n", queue[front].TaskId);

front++;

// make status completed

for (int i = 0; i < taskcount; i++)

{

if (task[i].TaskId == queue[front - 1].TaskId)

{

strcpy(task[i].Status, "Completed");

}

}

}

}

int main()

{

int choice;

while (1)

{

printf("\n1. Schedule task\n");

printf("2. Run task\n");

printf("3. Display scheduled Task\n");

printf("4. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice)

{

case 1:

printf("\nEnter the task id of the tasks to be scheduled\n");

for (int i = 0; i < taskcount; i++)

{

printf("%d ", task[i].TaskId);

}

printf("\n");

int id;

scanf("%d", &id);

for (int i = 0; i < taskcount; i++)

{

if (task[i].TaskId == id)

{

if (strcmp(task[i].Status, "Idle") == 0)

{

enqueue(task[i]);

strcpy(task[i].Status, "Queued");

}

else

{

printf("Task is already queued or completed\n");

}

}

}

break;

case 2:

dequeue();

break;

case 3:

printf("\nQueued tasks are:\n");

for (int i = front; i <= rear; i++)

{

printf("\nTaskId: %d", queue[i].TaskId);

printf("\tTaskTitle: %s", queue[i].TaskTitle);

printf("\nTaskDuration: %d", queue[i].TaskDuration);

printf("\tStatus: %s", queue[i].Status);

printf("\n\n");

}

break;

case 4:

exit(0);

break;

default:

printf("Invalid choice\n");

}

}

return 0;

}