الطالب :ميرنا أحمد طعمينا

الرقم الجامعي : 202210569

المشرفة: د.بيان السقى

using System;

using System.Collections.Generic;

class Program

{

class project

{

public string Name { get; set; }

public int Priority { get; set; } // 1-High, 2-Medium, 3-Low

public DateTime CreationDate { get; set; }

public project(string name, int priority, DateTime creationDate)

{

Name = name;

Priority = priority;

CreationDate = creationDate;

}

}

// Define a linked list node for completed tasks

class CompletedTaskNode

{

public project project { get; set; }

public CompletedTaskNode Next { get; set; }

public CompletedTaskNode(project project)

{

project=project;

Next = null;

}

}

static project[] tasks = new project[100]; // Array to store tasks

static int taskCount = 0; // Counter for active tasks

static CompletedTaskNode completedHead = null; // Head of the completed tasks linked list

static Queue<project> urgentTasks = new Queue<project>(); // Queue for urgent tasks

static void Main()

{

while (true)

{

Console.WriteLine("menu:");

Console.WriteLine("1. Add task");

Console.WriteLine("2. Display tasks");

Console.WriteLine("3. Delete task");

Console.WriteLine("4. Sort tasks by priority");

Console.WriteLine("5. Sort tasks by date");

Console.WriteLine("6. Complete a task");

Console.WriteLine("7. Display completed tasks");

Console.WriteLine("8. Add an urgent task");

Console.WriteLine("9. Display urgent tasks");

Console.WriteLine("10. Exit");

Console.Write("Select an option: ");

if (int.TryParse(Console.ReadLine(), out int option))

{

switch (option)

{

case 1:

AddTask();

break;

case 2:

DisplayTasks();

break;

case 3:

DeleteTask();

break;

case 4:

SortTasksByPriority();

break;

case 5:

SortTasksByDate();

break;

case 6:

CompleteTask();

break;

case 7:

DisplayCompletedTasks();

break;

case 8:

AddUrgentTask();

break;

case 9:

DisplayUrgentTasks();

break;

case 0:

return; // Exit the program

default:

Console.WriteLine("Invalid option. Please try again.");

break;

}

}

else

{

Console.WriteLine("Invalid input. Please enter a number.");

}

}

}

static void AddTask()

{

Console.Write("Enter task name: ");

string name = Console.ReadLine();

Console.Write("Enter priority (1-High, 2-Medium, 3-Low): ");

int priority = Convert.ToInt32(Console.ReadLine());

DateTime creationDate = DateTime.Now;

tasks[taskCount] = new project(name, priority, creationDate);

taskCount++;

Console.WriteLine("Task added successfully.");

}

static void DisplayTasks()

{

Console.WriteLine("\nActive Tasks:");

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

{

var task = tasks[i];

Console.WriteLine($"[{i}] Name: {task.Name}, Priority: {task.Priority}, Date: {task.CreationDate}");

}

}

static void DeleteTask()

{

DisplayTasks();

Console.Write("Enter the task number to delete: ");

int index = int.Parse(Console.ReadLine());

if (index >= 0 && index < taskCount)

{

for (int i = index; i < taskCount - 1; i++)

{

tasks[i] = tasks[i + 1];

}

tasks[--taskCount] = null;

Console.WriteLine("Task deleted ");

}

else

{

Console.WriteLine("enteer task num");

}

}

static void SortTasksByPriority()

{

Array.Sort(tasks, 0, taskCount, Comparer<project>.Create((x, y) => x.Priority.CompareTo(y.Priority)));

Console.WriteLine("Tasks sorted by priority.");

}

static void SortTasksByDate()

{

Array.Sort(tasks, 0, taskCount, Comparer<project>.Create((x, y) => x.CreationDate.CompareTo(y.CreationDate)));

Console.WriteLine("Tasks sorted by date.");

}

static void CompleteTask()

{

DisplayTasks();

Console.Write("Enter the task number to complete: ");

int index = int.Parse(Console.ReadLine());

if (index >= 0 && index < taskCount)

{

var completedTask = tasks[index];

// Remove from array and shift left

for (int i = index; i < taskCount - 1; i++)

{

tasks[i] = tasks[i + 1];

}

tasks[--taskCount] = null;

// Add to completed linked list

var newNode = new CompletedTaskNode(completedTask);

newNode.Next = completedHead;

completedHead = newNode;

Console.WriteLine("Task marked as completed.");

}

else

{

Console.WriteLine("Invalid task number.");

}

}

static void DisplayCompletedTasks()

{

Console.WriteLine("\nCompleted Tasks:");

var currentNode = completedHead;

while (currentNode != null)

{

var task = currentNode.project;

Console.WriteLine($"Name: {task.Name}, Priority: {task.Priority}, Date: {task.CreationDate}");

currentNode = currentNode.Next;

}

}

static void AddUrgentTask()

{

Console.Write("Enter urgent task name: ");

string name = Console.ReadLine();

Console.Write("Enter priority (1-High, 2-Medium, 3-Low): ");

int priority = int.Parse(Console.ReadLine());

var urgentTask = new project(name, priority, DateTime.Now);

urgentTasks.Enqueue(urgentTask);

Console.WriteLine("urgent task added");

}

static void DisplayUrgentTasks()

{

Console.WriteLine("\nUrgent Tasks:");

foreach (var task in urgentTasks)

{

Console.WriteLine($"Name: {task.Name}, Priority: {task.Priority}, Date: {task.CreationDate}");

}

}

}