|  |  |  |  |
| --- | --- | --- | --- |
| МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ  **Кафедра програмних систем і технологій**  Дисципліна  **«Структури даних, аналіз і алгоритми комп'ютерної обробки інформації»**  **Лабораторна робота №2**  **«Жадібні алгоритми»** | | | |
| **Виконав**: | Самойленко Олександр Васильович | **Перевірив**: | Бичков Олексій Сергійович |
| **Група** | ІПЗ-21 | **Дата перевірки** |  |
| **Форма навчання** | денна | **Оцінка** |  |
| **Спеціальність** | 121 |
| 2022 | | | |

# Завдання:

1. У якості входу задано n робіт , кожна з котрих має довжину ℓj і крайній термін dj. Визначте запізнення λj(σ) роботи j у розкладі σ або як різницю Cj(σ) – dj між часом завершення роботи та останнім часом, або як 0, якщо Cj(σ) ≤dj. Написати програму для формування розкладу, що мінімізує максимальне запізнення.
2. В якості входу задано n робіт, кожна з яких має довжину ℓj і останній час dj. Визначте запізнення λj(σ) роботи j у розкладі σ або як різницю Cj(σ) –dj між часом завершення роботи та останнім часом, або як 0, якщо Cj(σ)≤dj. Написати програму для формування розкладу, що мінімізує сумарного запізнення
3. На вхід задано n робіт, кожна з яких має час початку s j і час закінчення t j . Дві роботи конфліктують, якщо вони накладаються у часі - якщо одна з них починається між часом початку та закінчення іншої. Написати програму побудови підмножини максимального розміру з роботами, які не мають конфліктів.

# Аналіз алгоритмів:

# №1 - 2

Для першого та другого завдань відсортуємо алгоритми за зростанням дедлайну. Відповідно, обератимемо завдання по черзі з відсортованого списку. Різниця між завданнями 1 та 2 у способі обрахунку запізнення: для першого завдання перевірятимемо чи поточне запізнення більше за попереднє і визначатимемо максимальне; для другого – обраховуватимемо суму усіх запізнень.

# №3

Для третього завдання аналогічно відсортуємо задачі за дедлайном та обиратимемо задачу, якщо вона не «залізає» на попередню.

# Код програми:

# №1 - 2

public class Job  
{  
 public int Id { get; set; }  
 public double Length { get; set; }  
 public double Deadline { get; set; }  
 public double StartTime { get; set; }  
}

public class WeightOptions  
{  
 public double MaxLateness { get; private set; } = Double.MinValue;  
 public double TotalLateness { get; private set; } = 0;  
  
 public double GetMaxLateness(double currentLateness)  
 {  
 if (currentLateness > 0 && currentLateness > MaxLateness)  
 {  
 MaxLateness = currentLateness;  
 return MaxLateness;  
 }  
  
 return MaxLateness > 0  
 ? MaxLateness  
 : 0;  
 }  
  
 public double GetTotalLateness(double currentLateness)  
 {  
 if(currentLateness > 0)  
 {  
 TotalLateness += currentLateness;  
 }   
 return TotalLateness;  
 }  
}

public static class Task1  
{  
 private static List<Job> GetUnorderedJobs()  
 {  
 var result = new List<Job>();  
 var exit = false;  
 var idIterator = 0;  
 while (!exit)  
 {  
 var line = Console.ReadLine();  
 if (string.IsNullOrWhiteSpace(line))  
 {  
 exit = true;  
 break;  
 }  
  
 var values = line.Split(' ', StringSplitOptions.TrimEntries);  
 if(values.Length != 2   
 || !double.TryParse(values[0], out var jobLength)  
 || !double.TryParse(values[1], out var jobDeadline))  
 {  
 continue;  
 }  
  
 idIterator++;  
 var newJob = new Job()  
 {  
 Id = idIterator,  
 Deadline = jobDeadline,  
 Length = jobLength  
 };  
 result.Add(newJob);  
 }  
  
 return result;  
 }  
 public static void Run(Func<double, double> getLateness)  
 {  
 Console.WriteLine("Please, enter all the tasks as: \n" +  
 "\tlength deadline" +  
 "\nTo end an input, enter a blank line");  
 var orderedJobs = GetUnorderedJobs()  
 .OrderBy(job => job.Deadline)  
 .ToList();  
 double currentTime = 0;  
 foreach (var job in orderedJobs)  
 {  
 var lateness = currentTime + job.Length - job.Deadline;  
 currentTime += job.Length;  
 Console.WriteLine($"Added job with id:{job.Id}" +  
 $"\tlen:{job.Length}" +  
 $"\tdeadline:{job.Deadline}" +  
 $"\tweighted late:{getLateness(lateness)}" +  
 $"\tcurrent time:{currentTime}");  
 }  
   
 }  
}

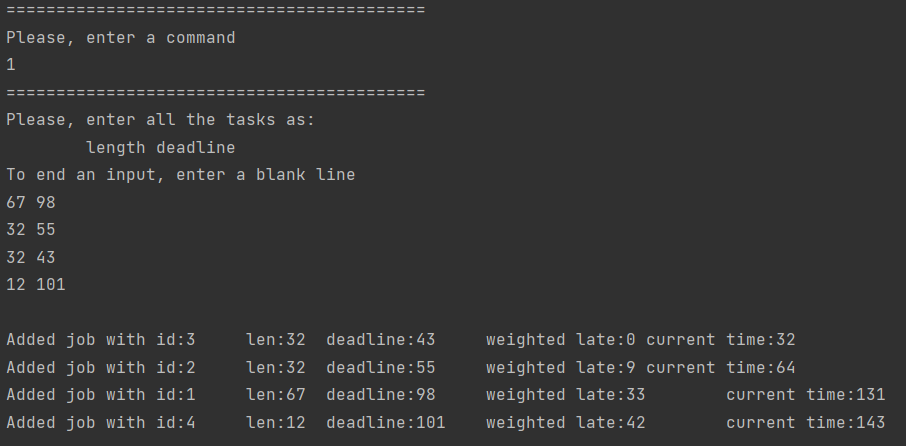
\*Основний метод для першого та другого завдань той самий, різняться лиш функції обрахунку запізнення, що є аргументами цього методу та містяться у класі WeightOptions

# №3

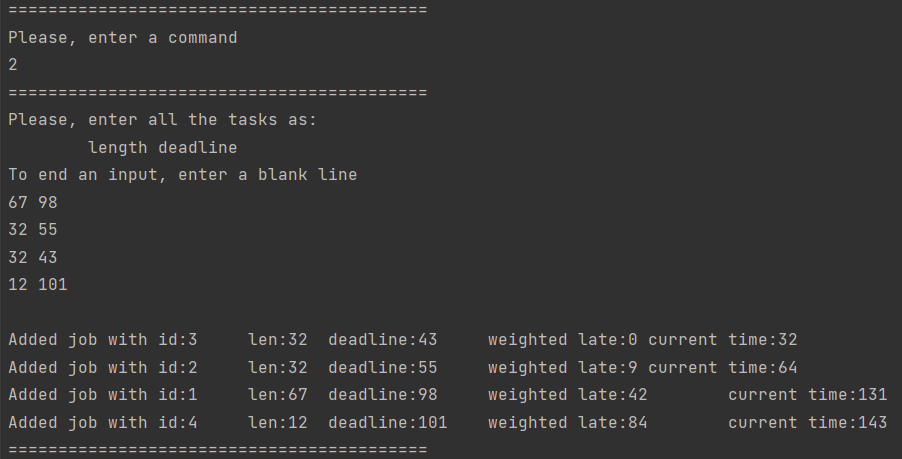
public static class Task2  
{  
 private static List<Job> GetUnorderedJobs()  
 {  
 var result = new List<Job>();  
 var exit = false;  
 int idIterator = 0;  
 while (!exit)  
 {  
 var line = Console.ReadLine();  
 if (string.IsNullOrWhiteSpace(line))  
 {  
 exit = true;  
 break;  
 }  
  
 var values = line.Split(' ', StringSplitOptions.**TrimEntries**);  
 if(values.Length != 2   
 || !double.TryParse(values[0], out var jobStart)  
 || !double.TryParse(values[1], out var jobDeadline))  
 {  
 continue;  
 }  
  
 idIterator++;  
 var newJob = new Job()  
 {  
 Id = idIterator,  
 Deadline = jobDeadline,  
 StartTime = jobStart  
 };  
 result.Add(newJob);  
 }  
  
 return result;  
 }  
  
 public static void Run()  
 {  
 Console.WriteLine("Please, enter all the tasks as: \n" +  
 "\tlength deadline" +  
 "\nTo end an input, enter a blank line");  
 var orderedJobs = GetUnorderedJobs()  
 .OrderBy(job => job.Deadline)  
 .ToList();  
 double currentTime = 0;  
 foreach (var job in orderedJobs)  
 {  
 if (job.StartTime < currentTime)  
 {  
 continue;  
 }  
 Console.WriteLine($"Added job with id:{job.Id}" +  
 $"\tstart:{job.StartTime}" +  
 $"\tdeadline:{job.Deadline}" +  
 $"\tprevious time:{currentTime}");  
 currentTime = job.Deadline;  
 }  
 }  
}

# Результат виконання:

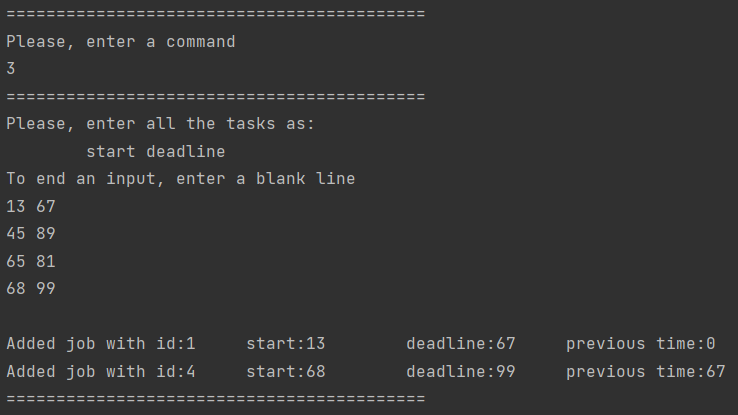
# №1



# №2



# №3



# Висновки:

Було виконано завдання лабораторної роботи номер 1, побудовано жадібні алгоритми знаходження розкладу різних типів.