**8 ПЕРЕЧИСЛЕНИЯ. СТРУКТУРЫ**

Задание 1. Дан фрагмент текста, запрашиваемый у пользователя. Написать программу заменяющие в нем гласные русские буквы на «\*».

Листинг программы:

class Program

{

//Структура Train

private struct Train

{

public int Id { get; init; }

public string Destination { get; init; }

public DateTime Date { get; init; }

public Train(int id, string destination, DateTime date)

{

Id = id;

Destination = destination;

Date = date;

}

}

//Кол-во train-ов в массиве

private const int TrainsCount = 8;

static void Main(string[] args)

{

//Создание массива train-ов

Train[] trains = new Train[TrainsCount];

for (int i = 0; i < trains.Length; i++)

{

trains[i] = InputTrain();

}

//Сортировка train-ов по их индексам

Array.Sort(trains, (a, b) => a.Id.CompareTo(b.Id));

//Поиск поезда по id

try

{

Console.Write("Введите id чтобы найти поезд: ");

var id = int.Parse(Console.ReadLine());

var train = GetTrainById(id, trains);

OutputTrain(train);

}

catch (KeyNotFoundException exception)

{

//Если не удалось найти train - выводим на экран сообщение ошибки

Console.WriteLine(exception.Message);

}

}

private static void OutputTrain(Train train)

{

Console.WriteLine(

$"Id: { train.Id };" +

$" Destination: { train.Destination }; " +

$"Date: { train.Date.Date }");

}

private static Train GetTrainById(int id, Train[] trains)

{

foreach (var train in trains)

{

if (train.Id == id)

return train;

}

//Если не удалось найти train по id - выбрасываем ошибку с сообщением не удалось найти

throw new KeyNotFoundException("Поезд не найден");

}

private static Train InputTrain()

{

Console.Write("Введите id: ");

var id = int.Parse(Console.ReadLine());

Console.Write("Введите место назначения: ");

var destination = Console.ReadLine();

Console.Write("Введите время отправления: ");

var date = DateTime.Parse(Console.ReadLine());

return new Train(id, destination, date);

}

}

Таблица 8.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Введите id: 14  Введите место назначения: Tokyo  Введите время отправления: 14.02.2022 0:00:00  Введите id: 22  Введите место назначения: Seul  Введите время отправления: 14.02.2022 0:00:00  Введите id чтобы найти поезд: 14 | Id: 14; Destination: Tokyo; Date: 14.02.2022 0:00:00 |

Анализ результатов:

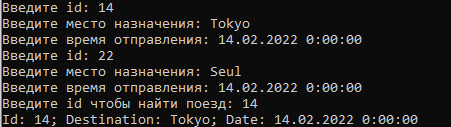


Рисунок 8.1 – Результат работы программы

Источник: собственная разработка

Задание 2. Создайте класс Accauntant с методом bool AskForBonus (Postmworker, int hours), отражающее давать или нет сотруднику премию. Если сотрудник отработал больше положеных часов в месяц, то ему положена премия.

Листинг программы:

class Program

{

//Перечиление работкников; числа - сколько часов должны отработать работники

enum Employs

{

Manager = 150,

Admin = 270,

Director = 290

};

//Статический класс Accauntant

static class Accauntant

{

//Метод для проверки даваьт ли работнику премию

//emp - работник; workedHours - отработанные работником часы

//возвращает true или false

public static bool AskForBonus(Employs emp, int workedHours)

{

return (int)emp < workedHours;

}

}

static void Main(string[] args)

{

//Словарь отработанных часов работниками

//ключ - работник

//value - сколько часов он отработал

var empHours = new Dictionary<Employs, int>

{

[Employs.Manager] = 100,

[Employs.Admin] = 310,

[Employs.Director] = 90

};

//Перебираем словарь и выводим на экран выдавать примию или нет.

foreach (var pair in empHours)

{

Console.WriteLine($"{ pair.Key }: { Accauntant.AskForBonus(pair.Key, pair.Value) }");

}

}

}

Таблица 8.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Manager = 150,  Admin = 270,  Director = 290 | Manager: False  Admin: True  Director: False |

Анализ результатов:



Рисунок 8.2 – Результат работы программы

Источник: собственная разработка