1. **«СТРУКТУРЫ, ПЕРЕЧИСЛЕНИЯ.»**

Задание 1. Описать структуру с именем TRAIN, содержащую следующие поля:

 название пункта назначения;

 номер поезда;

 время отправления.

Написать программу, выполняющую следующие действия:

 ввод с клавиатуры данных в массив, состоящий из восьми

элементов типа TRAIN (записи должны быть упорядочены по номерам

поездов);

 вывод на экран информации о поезде, номер которого введен с

клавиатуры (если таких поездов нет, вывести соответствующее сообщение).

Листинг программы:

using System;

struct TRAIN

{

public string destination;

public int trainNumber;

public DateTime departureTime;

}

class Program

{

static void Main(string[] args)

{

TRAIN[] trains = new TRAIN[8];

Console.WriteLine("Введите данные о 8 поездах:");

for (int i = 0; i < 8; i++)

{

Console.Write($"Поезд #{i + 1}\nНазвание пункта назначения: ");

trains[i].destination = Console.ReadLine();

Console.Write("Номер поезда: ");

trains[i].trainNumber = int.Parse(Console.ReadLine());

Console.Write("Время отправления (в формате ДД.ММ.ГГГГ ЧЧ:ММ): ");

trains[i].departureTime = DateTime.ParseExact(Console.ReadLine(), "dd.MM.yyyy HH:mm", null);

}

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

Console.Write("Введите номер поезда: ");

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

int index = Array.FindIndex(trains, t => t.trainNumber == number);

if (index == -1)

{

Console.WriteLine($"Поезд #{number} не найден.");

}

else

{

Console.WriteLine($"Информация о поезде #{number}:\n" +

$"Пункт назначения: {trains[index].destination}\n" +

$"Время отправления: {trains[index].departureTime}");

}

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Поезд #1  Название пункта назначения: Москва  Номер поезда: 324  Время отправления: 05.06.2023 21:15  Поезд #2  Название пункта назначения: Питер  Номер поезда: 546  Время отправления: 21.06.2023 23:56  Поезд #3  Название пункта назначения: Минск  Номер поезда: 910  Время отправления : 16.07.2023 09:15  Поезд #4  Название пункта назначения: Гродно  Номер поезда: 610  Время отправления: 15.04.2023 20:01  Поезд #5  Название пункта назначения: Киев  Номер поезда: 450  Время отправления: 21.03.2023 02:03  Поезд #6  Название пункта назначения: Вильнюс  Номер поезда: 1002  Время отправления: 25.10.2023 21:17  Поезд #7  Название пункта назначения: Белосток  Номер поезда: 230  Время отправления: 30.12.2023 01:01  Поезд #8  Название пункта назначения: Варшава  Номер поезда: 760  Время отправления: 12.12.2023 09:56 | Введите номер поезда: 1002  Информация о поезде #1002:  Пункт назначения: Вильнюс  Время отправления: 25.10.2023 21:17:00 |

Анализ результатов:

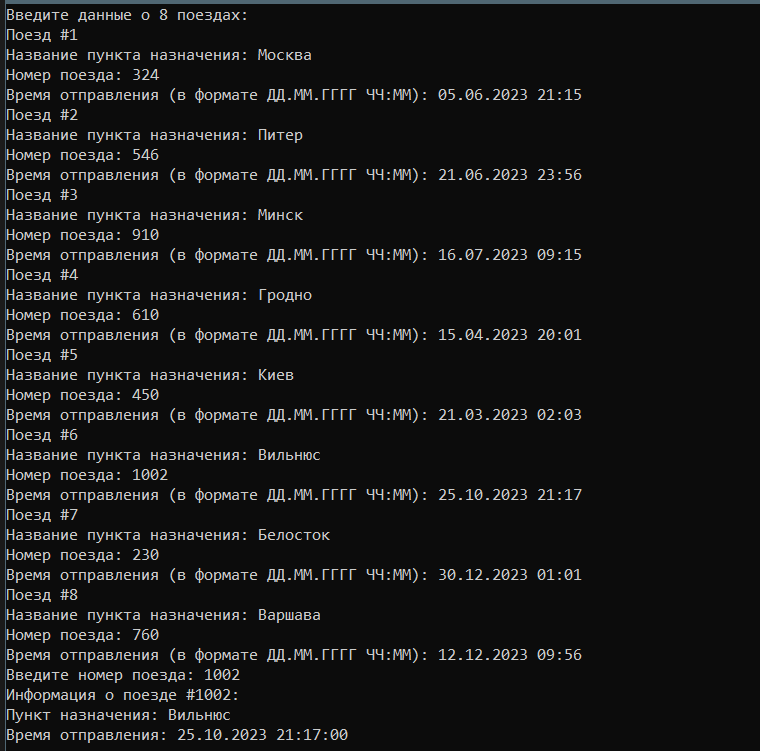


Рисунок 1.1 – Результат работы программы

Источник: собственная разработка.

Задание 2. 1. Создайте перечисление, в котором будут содержаться должности сотрудников как имена констант. Присвойте каждой константе значение, задающее количество часов, которые должен отработать сотрудник за месяц.

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

Листинг программы:

using System;

namespace EmployeeBonus

{

enum Post

{

Manager = 160,

Programmer = 180,

Developer = 180,

Designer = 200

}

class Accountant

{

public bool AskForBonus(Post worker, int hours)

{

if (hours > (int)worker)

{

return true;

}

else

{

return false;

}

}

}

class Program

{

static void Main(string[] args)

{

Accountant accountant = new Accountant();

Post manager = Post.Manager;

Console.Write("Введите количество отработанных часов менеджера: ");

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

bool managerBonus = accountant.AskForBonus(manager, managerHours);

Console.WriteLine($"Менеджер отработал {managerHours} часов{(managerBonus ? ", ему положена премия" : ", ему не положена премия")}");

Post programmer = Post.Programmer;

Console.Write("Введите количество отработанных часов программиста: ");

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

bool programmerBonus = accountant.AskForBonus(programmer, programmerHours);

Console.WriteLine($"Программист отработал {programmerHours} часов{(programmerBonus ? ", ему положена премия" : ", ему не положена премия")}");

Post designer = Post.Designer;

Console.Write("Введите количество отработанных часов дизайнера: ");

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

bool designerBonus = accountant.AskForBonus(designer, designerHours);

Console.WriteLine($"Дизайнер отработал {designerHours} часов{(designerBonus ? ", ему положена премия" : ", ему не положена премия")}");

Post developer = Post.Manager;

Console.Write("Введите количество отработанных часов разработчика: ");

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

bool developerBonus = accountant.AskForBonus(developer, developerHours);

Console.WriteLine($"Разработчик отработал {developerHours} часов{(developerBonus ? ", ему положена премия" : ", ему не положена премия")}");

Console.ReadKey();

}

}

}

Таблица 1.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Количество отработанных часов менеджера: 165  Количество отработанных часов программиста: 168  Количество отработанных часов дизайнера: 190  Количество отработанных часов разработчика: 200 | Менеджер отработал 165 часов, ему положена премия  Программист отработал 168 часов, ему не положена премия  Дизайнер отработал 190 часов, ему не положена премия  Разработчик отработал 200 часов, ему положена премия |

Анализ результатов:

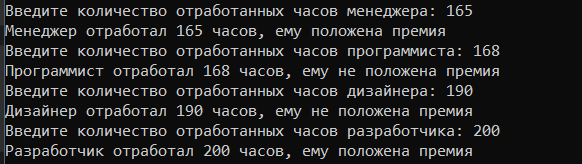


Рисунок 1.2 – Результат работы программы

Источник: собственная разработка.