1. **Перечисления. Структуры**

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

фамилия, имя;

номер телефона;

дата рождения.

Написать программу, выполняющую следующие действия:

ввод с клавиатуры данных в массив, состоящий из восьми элементов типа NOTE (записи должны быть размещены по алфавиту);

вывод на экран информации о людях, чьи дни рождения приходятся на месяц, значение которого введено с клавиатуры (если таких нет, вывести соответствующее сообщение).

Листинг программы:

namespace Task\_1

{

// Структура

public struct Note : IComparable<Note>

{

public string fNameSName;

public string telehpone;

public DateOnly berthday;

public int CompareTo(Note other)

{

throw new NotImplementedException();

}

public override string ToString()

{

return ($"{fNameSName}\t{telehpone}\t{berthday}");

}

}

class Program

{

public static void Main(string[] args)

{

Console.Write("Введите кол-во записей: ");

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

Console.WriteLine("Введите данные");

Note[] note = GetSortNote(InputUserData(size));

PrintNotes(note);

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

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

PrintNotes(GetUserByMonthBirth(note, month));

}

public static Note[] InputUserData(int size)

{

Note[] note = new Note[size];

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

{

Console.Write("Фамилия и Имя: ");

note[i].fNameSName = Console.ReadLine();

Console.Write("Номер телефона: ");

note[i].telehpone = Console.ReadLine();

Console.Write("Дату рождения: ");

note[i].berthday = DateOnly.Parse(Console.ReadLine());

}

return note;

}

public static void PrintNotes(IEnumerable<Note> notes)

{

if (notes.Count() == 0)

{

Console.WriteLine("Нет такого человека");

}

else

{

foreach (var note in notes)

{

Console.WriteLine(note.ToString());

}

}

}

public static Note[] GetSortNote(Note[] note)

{

Array.Sort(note, (note1, note2) => note1.fNameSName.CompareTo(note2.fNameSName));

return note;

}

public static List<Note> GetUserByMonthBirth(Note[] note, int month)

{

List<Note> noteByMonth = new List<Note>();

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

{

if (month == note[i].berthday.Month)

{

noteByMonth.Add(note[i]);

}

}

return noteByMonth;

}

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 8  Микитчук Вячек +345 (22 07 00)  Коваль Петр +436 (13.05.08)  Анапов Иван +324 (9 9 9)  Арвбов Вася +242 (15/07/200)  Степко Степь +324 (1 10 2008)  Ганжа Гарик +555 (28.08.1998)  Кук Вера +434 (01 01 01)  Супчик Света +998 (09 3 15)  5 | Анапов Иван +324 (9 9 9)  Арвбов Вася +242 (15/07/200)  Ганжа Гарик +555 (28.08.1998)  Коваль Петр +436 (13.05.08)  Кук Вера +434 (01 01 01)  Микитчук Вячек +345 (22 07 00)  Степко Степь +324 (1 10 2008)  Супчик Света +998 (09 3 15)  Коваль Петр +436 (13.05.08) |

Анализ результатов:

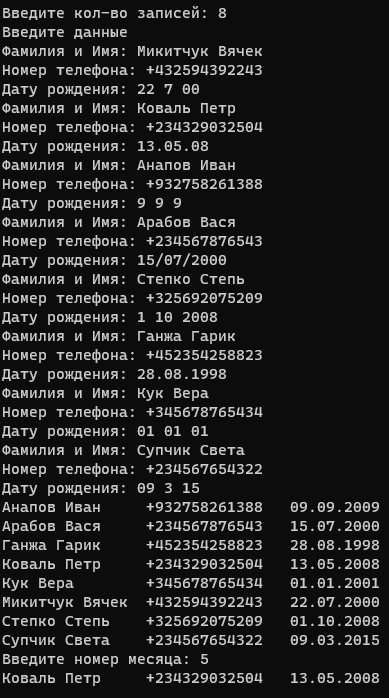


Рисунок 1.1 – Результат работы программы

Источник: собственная разработка

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

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

Листинг программы:

namespace Task\_2

{

public enum Post

{

Analyst = 164,

Teacher = 180,

Gardener = 150,

Loader = 154,

Agronomist = 170,

Programmer = 160,

Builder = 100

}

public class Program

{

public static void Main()

{

foreach (string post in Enum.GetNames(typeof(Post)))

{

Console.Write($"Введите, cколько времени проработал за этот месяц {post}: ");

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

Enum.TryParse<Post>(post, true, out Post postEnum);

Console.WriteLine($"{post} премия: {GetBonusLoc(postEnum, hours)}");

Console.WriteLine();

}

}

public static string GetBonusLoc(Post worker, int hours)

{

Accauntant acc = new Accauntant();

if (acc.AskForBonus(worker, hours))

{

return "Положена премия";

}

else

{

return "Не положена премия";

}

}

}

public class Accauntant

{

public Accauntant()

{

}

public bool AskForBonus(Post worker, int hours)

{

if ((int)worker < hours)

{

return true;

}

else

{

return false;

}

}

}

}

Таблица 2.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 202 100 205 155 170 196 181 |  |

Анализ результатов:

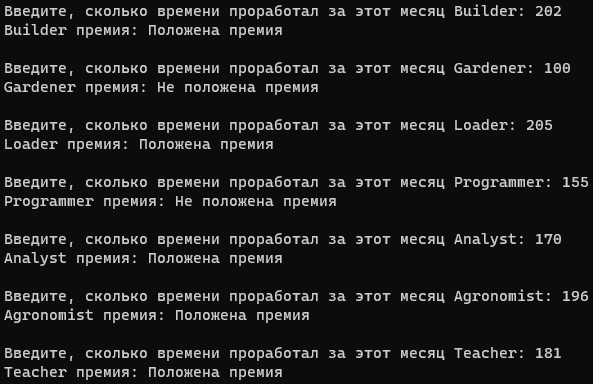


Рисунок 2.1 – Результат работы программы

Источник: собственная разработка