**Введение**

В современном мире, насыщенном технологиями и высокотехнологичными решениями, роль алгоритмов становится неотъемлемой частью нашей повседневной жизни. Алгоритмы, как последовательности инструкций, программ и вычислительных процессов, проникают в самые разнообразные сферы нашего существования, облегчая и улучшая множество аспектов нашей повседневной рутины. От простых задач вроде рекомендации контента на платформах потокового видео до сложных систем управления в медицине и науке - алгоритмы не только влияют на нашу жизнь, но и играют ключевую роль в формировании современного облика нашего общества. В данной работе мы рассмотрим пример, как алгоритмы оказывают сильное влияние на различные аспекты нашего быта, обеспечивая нам более эффективные, интеллектуальные и комфортные условия жизни.

**Электронная «Записная книга»**

Записная книга представляет собой программу, которая предоставляет данные о большом количестве людей, а именно: фамилия, имя, отчество, адрес, дата рождения, а также их номера их сотовых телефонов.

Для хранения данных о людях используется список объектов, который создается при запуске программы. В приложении предоставляет возможность поиска людей по номеру телефона или по другим параметрам, которые можно комбинировать между собой.

**Разработка кода**

Для начала был создан класс, представляющий человека, а данные для объектов класса было решено хранить непосредственно прямо в коде в виде списка.

Далее был разработан прототип программы, где всё управление было только на ручном вводе, а выбор действий осуществлялся вводом номера пункта меню, была введена проверка на корректность введенного номера, ФИО и года рождения. Когда работа с данными была налажена, было создано базовое оформление, выбраны цвета текста и добавлен выбор действия с помощью стрелок и клавиши enter. Затем выбор параметра поиска был изменен на способ из главного меню, выбор месяца, а также даты был также переведен на управление клавишами.

**Описание структуры кода**

Программа состоит из 4 классов, разбитых на 4 файла. Главным классом является Notebook, в нём расположена функция Main:

*static* void Main() {

        CreatePersons(persons);  *// Люди для примера*

        PersonComparer cmp = new PersonComparer();

        persons.Sort(cmp);

        while (true) {

            Console.SetWindowSize(92, 45);

            Console.CursorSize = 100;

            Console.ForegroundColor = ConsoleColor.DarkCyan;

            Display.PrintHeader();

            Console.ForegroundColor = ConsoleColor.DarkGreen;

            if (statusScr == 0) {

                Menu();

            } else if ((int)statusScr == 1) {

                FindByNumber();

            } else if ((int)statusScr == 2) {

                ChooseParam();

            } else if ((int)statusScr == 3) {

                PrintByParam();

            } else if ((int)statusScr == 4) {

                FindByParam();

            }

            Console.Clear();

        }

    }

В функции реализован основной цикл, который отвечает за запуск «экранов» и очистку консоли.

В области переменных располагаются 2 enum: screenStatuses – хранит значения для выбора «экрана», printParams – хранит параметры вывода на экран. Переменные statusScr – выбор следующего выведенного экрана, printByParam – выбор выводимого на экран параметра поиска, findParam – универсальная переменная для хранения вводимых параметров, error – значение ошибки. Два списка: persons

static void Menu – является «экраном» отвечающим за вывод и активацию дальнейших действий, а именно поиск человека в записной книге по номеру его сотового телефона, или по другим различным параметрам. После выбора действия меняет переменную statusScr на соответствующее значение.

static void FindByNumber – «экран» отвечает за ввод номера телефона и проверяет его на наличие посторонних символов, в случае обнаружения таковых, не меняет значение переменной statusScr, вносит в переменную error значение ошибки и после чего «экран» ввода перезапускается, выводится ошибка и запрашивается повторный ввод номера. После успешного ввода номера statusScr принимает значение, отвечающее за вывод найденных людей на экран, в переменную findParam сохраняется введённый номер, а переменная printByParam, отвечающая за то, какой именной был введен параметр, принимает статус, присущий номеру телефона.

static void ChooseParam – «экран» отвечает за выбор параметра, по которому будет произведен поиск, меняет значение printByParam на соответствующее выбору, затем меняет statusScr на ввод параметра.

static void FindByParam – «экран» отвечает за ввод выбранного параметра, если это имя, фамилия или отчество, то проверяет на наличие цифр, если это год, то проверяет что он состоит только из цифр и входит в допустимые значения, если это месяц или день, то выбор происходит с помощью клавиш. Если в вводе была обнаружена ошибка, то значение error меняется, и перезапускается экран ввод с выводом ошибки. В случае успешного ввода statusScr принимает значение, отвечающее за вывод найденных людей на экран, в переменную findParam сохраняется введённый параметр.

static void PrintByParam – «экран» отвечает за вывод людей, подходящих под заданный параметр. Для начала она вносит в отдельный список подходящих людей, в затем выводит их на экран, если список оказывается пустым, выводится сообщение «Ничего не найдено». На этом же экране предоставляется возможность добавить значение параметра и применить его к уже выведенным, изменить значение или выйти в меню (если поиск был по номеру, пункт добавления параметра отсутствует).

static int Choose – отвечает за работу выбора чего-либо с помощью клавиш.

Следующий класс – public Person, отвечает за хранение данных об одном человеке.

Его поля:

* private string Surname
* private string Name
* private string Patronymic
* private string Number
* private string Address
* private string Year
* private string Month
* private string Date

В нём находятся 3 конструктора класса, метод public void printData, отвечающая за вывод данных, 8 методов с модификатором public, которые возвращают bool, сравнивая передаваемое им значение параметра с хранящимся в объекте, в случае совпадения возвращает true. 3 метода: public int getYear, public string getMonth, public int getDate – которые возвращают год, месяц, день рождения человека, хранящегося в объекте.

Следующий класс – public PersonComparer: IComparer<Person>, который обеспечивает сортировку объектов класса Person по дате рождения.

Последний класс Display – несет сугубо косметическую необходимость.