

## СЦЕНАРИЙ ВИДЕО: "КОНСОЛЬНАЯ ПРОГРАММА 'ТЕЛЕФОННАЯ КНИГА' НА JAVA С НУЛЯ"

Этот сценарий основан на структуре, предоставленной в исходном документе, и адаптирован под проект "Телефонная книга" на Java с использованием ООП и работы с файлами (FileWriter/FileReader).

---

### ЧАСТЬ 1: ВВЕДЕНИЕ (3 минуты)

Тайминг	Содержание и Акценты
0:00–3:00	"Привет, друзья! Сегодня мы создадим с нуля классическое консольное приложение 'Телефонная книга' на Java — идеальный проект для начинающих, который поможет вам разобраться в основах <b>Объектно-Ориентированного Программирования (ООП)</b> и важнейшей теме — <b>работе с файлами!</b> "
	"Эта программа — отличный способ прокачать навыки Java, научиться работать с коллекциями (ArrayList), логическими операторами, и что самое главное, <b>обеспечить сохранение данных</b> на диске, чтобы они не пропадали после закрытия программы."
	<b>Наша Телефонная книга будет уметь:</b>
	* Создавать и хранить записи контактов (Имя, Телефон).
	* Использовать <b>ООП</b> , создав класс Contact для каждой записи.
	* <b>Сохранять</b> данные в текстовый файл (.txt) с помощью FileWriter.
	* <b>Загружать</b> данные из файла при старте программы с помощью FileReader.
	* Принимать выбор пользователя через консольный ввод.
	* Работать в цикле для многократного выполнения команд (меню).
	"Весь код мы напишем в <b>структурированном виде</b> , разбив его на классы, чтобы вы поняли, как работают реальные приложения!"

---

### ЧАСТЬ 2: ТЕХНОЛОГИИ И ИНСТРУМЕНТЫ (3 минуты)

Тайминг	Содержание и Акценты
3:00–6:00	"Мы используем только стандартные, нативные возможности Java. Всё максимально просто и применимо на любом проекте."
	<b>Что применим:</b>
	* <b>ООП:</b> Создание класса Contact с полями и методами (инкапсуляция).
	* <b>Коллекции:</b> ArrayList или List<Contact> для хранения контактов в оперативной памяти.
	* <b>Файловый I/O:</b> Классы <b>FileWriter</b> и <b>FileReader</b> (обернутые в PrintWriter и BufferedReader для удобной работы со строками).
	* <b>Scanner:</b> Для взаимодействия с пользователем и получения команд меню.
	* <b>Циклы while:</b> Для основного цикла программы (отображение меню).
	* <b>Условные операторы if-else / switch:</b> Для обработки команд меню.
	* <b>Обработка исключений (try-catch):</b> Для корректной работы с файлами (IOException, FileNotFoundException).
	"Мы создадим компактный, но <b>полноценный</b> менеджер данных, демонстрирующий все ключевые концепции ООП и файлового ввода/вывода на начальном уровне!"

### ЧАСТЬ 3: ЧЕМУ ВЫ НАУЧИТЕСЬ (3 минуты)

Тайминг	Содержание и Акценты
6:00–9:00	"После просмотра видео вы получите не только работающую программу, но и глубокое понимание фундаментальных концепций, которые нужны каждому Junior Java-разработчику."
	<b>Вы поймёте:</b>
	* Как применять <b>ООП</b> на практике, создавая классы-модели.
	* Как использовать <b>коллекции</b> (ArrayList) для динамического хранения данных.
	* Как настроить <b>персистентность</b> данных, используя FileWriter для сохранения.

Тайминг	Содержание и Акценты
	* Как реализовать <b>загрузку</b> данных при старте программы, используя FileReader.
	* Как использовать конструкцию <b>try-with-resources</b> для безопасной работы с файлами.
	* Как строить циклический <b>интерфейс меню</b> и обрабатывать команды.
	"Это идеальный стартовый проект для практики <b>алгоритмического мышления</b> и освоения <b>файлового I/O</b> в Java!"

#### ЧАСТЬ 4: ФУНКЦИОНАЛ ПРОГРАММЫ (3 минуты)

Тайминг	Содержание и Акценты
<b>9:00–12:00</b>	"Мы создадим полноценную систему управления контактами с удобным меню и надежным сохранением данных!"
	<b>Что будет у нашей программы:</b>
	* <b>Объектно-ориентированная модель:</b> Отдельный класс Contact для структурирования информации.
	* <b>Интуитивное меню:</b> Простой выбор команд (Добавить, Показать, Выйти).
	* <b>Автоматическая загрузка:</b> Книга загружается из файла phonebook.txt при запуске.
	* <b>Надежное сохранение:</b> Данные записываются в файл при выходе из программы.
	* <b>Прозрачный формат файла:</b> Использование простого разделителя (например, ;) для чтения и записи.
	* <b>Защита от сбоев:</b> Обработка ошибок ввода-вывода (например, если файл не найден или поврежден).
	"Мы создадим систему, где все ваши контакты будут <b>сохранены</b> и доступны при следующем запуске — это ключевой шаг в создании реальных приложений!"

#### ЧАСТЬ 5: НАПИСАНИЕ КОДА (15 минут)

Тайминг	Содержание и Акценты
12:00–27:00	"Всё максимально понятно и структурировано. Мы не просто пишем код, мы разбираем его <b>логику</b> и <b>алгоритмы</b> ."
	<b>Блок 1: Класс Contact (Модель ООП)</b>
	* Создаём класс Contact с приватными полями name и phoneNumber.
	* Добавляем конструктор и публичные геттеры (принцип инкапсуляции).
	* Пишем метод toFileString() для форматирования контакта в строку с разделителем.
	<b>Блок 2: Класс PhoneBookManager (Коллекция и Логика)</b>
	* Инициализируем List<Contact> contacts для хранения данных.
	* Реализуем метод addContact(Contact contact) для добавления в список.
	* Реализуем метод viewAllContacts() для вывода списка контактов.
	<b>Блок 3: Файловый I/O (Сохранение)</b>
	* Реализуем метод saveContactsToFile() с использованием <b>FileWriter</b> и <b>PrintWriter</b> .
	* Используем конструкцию <b>try-with-resources</b> для автоматического закрытия ресурсов.
	* В цикле записываем каждый контакт, используя toFileString().
	<b>Блок 4: Файловый I/O (Загрузка)</b>
	* Реализуем метод loadContactsFromFile() с использованием <b>FileReader</b> и <b>BufferedReader</b> .
	* Читаем файл <b>построчно</b> в цикле while.
	* Используем метод <b>split()</b> для парсинга строки по разделителю и восстановления объекта Contact.
	<b>Блок 5: Класс Main (Меню и Управление)</b>
	* Инициализируем Scanner и PhoneBookManager.
	* Организуем главный цикл программы (while (running)).

Тайминг	Содержание и Акценты
	* Реализуем логику меню (switch) для вызова методов менеджера.
	* <b>АКЦЕНТ:</b> Обеспечиваем вызов <b>saveContactsToFile()</b> непосредственно перед выходом из программы.
	"Каждый блок мы будем писать и сразу же тестировать, чтобы вы видели результат каждого этапа и понимали, как данные перемещаются между памятью и диском!"

## ЧАСТЬ 6: ЗАВЕРШЕНИЕ (3 минуты)

Тайминг	Содержание и Акценты
<b>27:00–30:00</b>	"Подводим итоги: Мы научились создавать интерактивные консольные приложения на Java, работать с ООП, коллекциями и, главное, с <b>файловым вводом/выводом</b> ."
	* Реализовали полноценный, функциональный менеджер контактов с персистентностью данных.
	* На практике разобрали работу с Scanner, ArrayList, FileWriter и FileReader.
	* Создали надежную систему, использующую <b>try-with-resources</b> для безопасности I/O операций.
	* Построили четкую структуру с разделением обязанностей между классами (Contact, Manager, Main).
	"Этот проект — отличная основа для дальнейшего развития! Вы можете добавить функции поиска, удаления, редактирования контактов, а также сохранения статистики в формат JSON или XML!"
	"Если тебе понравился проект, поставь лайк и подпишись, впереди будет ещё больше практических проектов на Java!"
	"С вами был (твое имя), до встречи в следующем видео, где мы создадим новый полезный инструмент на Java!"

## КЛЮЧЕВЫЕ АКЦЕНТЫ СЦЕНАРИЯ:

Категория	Акцент
<b>Демонстрация базовых концепций:</b>	<b>ООП:</b> Создание модели Contact и использование инкапсуляции.
	<b>Файловый I/O:</b> FileWriter / FileReader для сохранения/загрузки данных.
	<b>Безопасность:</b> Использование try-with-resources для закрытия потоков.
	<b>Логические операции и Циклы:</b> Управление меню и циклом жизни программы.
<b>Практическая польза:</b>	<b>Реальная работающая программа</b> с дружелюбным интерфейсом.
	<b>Персистентность данных:</b> Данные сохраняются между сессиями.
	Код, который можно сразу использовать и расширять (навыки для резюме).
<b>Образовательная ценность:</b>	Понимание работы <b>память vs. диск</b> (коллекция vs. файл).
	Навыки построения логики меню и валидации данных.
	Принципы организации крупных проектов с разделением на классы.