# ЛР1 Обобщённое программирование в Java

Ниже представлены 15 разнообразных заданий, каждое из которых фокусируется на реализации иерархии интерфейсов и классов с использованием механизма Generics в Java. Каждое задание ориентировано на определенную доменную область, что позволяет применить знания о Generics в различных контекстах.

Распределение вариантов: вариант – ваш номер в списке группы. Если в группе людей больше, чем вариантов, то список начинается сначала (например, 16ый человек в списке берет 1ый вариант, 17 берет 2 и тд.)

## Содержание отчета

1. Правильно оформленный титульный лист.
2. Текст вашего варианта.
3. Краткое описание состава программы: класс <название> делает это, предназначен для того-то
4. UML-диаграмма классов
5. Скриншот работы программы
6. Листинг разработанной программы. Для сокращения места можно удалить комментарии, можно использовать Consolas 10 – выглядит лучше и занимает меньше места.

## Общие рекомендации по реализации заданий

Для каждого из приведенных заданий рекомендуется придерживаться следующих шагов:

### Определение Интерфейсов:

* Создайте базовый обобщенный интерфейс Repository<T>, включающий основные CRUD-операции (создание, чтение, обновление, удаление).
* Создайте расширяющие интерфейсы, такие как SortableRepository<T>, добавляющие методы для сортировки элементов.
* При необходимости добавьте интерфейсы, расширяющие предыдущие, например, SearchableRepository<T>, включающие методы для поиска.

### Определение Сущностей:

* Создайте классы, представляющие различные типы сущностей в выбранной доменной области. Убедитесь, что они соответствуют требованиям интерфейсов (например, могут реализовывать дополнительные интерфейсы, если требуется).

### Реализация Репозиториев:

* Для каждой сущности создайте класс репозитория, который реализует соответствующие интерфейсы. Например, UserRepository implements SearchableRepository<User>.
* Реализуйте методы интерфейсов, обеспечивая функциональность для работы с данными (например, сортировку, поиск).

### Демонстрация Использования:

* Создайте основной класс (например, Main), где продемонстрируйте создание экземпляров репозиториев, добавление, удаление, обновление и поиск элементов.
* Покажите работу методов сортировки и поиска, выводя результаты на консоль или другой интерфейс.

### Дополнительные Возможности:

* При необходимости добавьте обработку исключений, валидацию данных или другие механизмы для повышения надежности системы.
* Реализуйте интерфейсы Comparator или Comparable для объектов, если требуется специализированная логика сортировки.

# Варианты

### Система Управления Библиотекой

Разработайте систему для управления библиотечным фондом, включающую различные типы материалов (книги, журналы, электронные книги). Необходимо создать обобщенные интерфейсы и классы для выполнения операций CRUD, поиска и сортировки.

**Требования**

**Интерфейсы:**

* Repository<T> – базовый интерфейс для операций CRUD.
* SortableRepository<T> extends Repository<T> – добавляет возможность сортировки.
* SearchableRepository<T> extends SortableRepository<T> – добавляет возможность поиска по критериям.

**Сущности:**

* Book
* Magazine
* EBook

**Реализационные классы репозиториев:**

* BookRepository implements SearchableRepository<Book>
* MagazineRepository implements SearchableRepository<Magazine>
* EBookRepository implements SearchableRepository<EBook>

1. **Система Управления Магазином Электроники**

Создайте систему для управления ассортиментом электронных товаров, включая смартфоны, ноутбуки и планшеты. Система должна поддерживать операции добавления, удаления, обновления и поиска товаров, а также их сортировку по различным критериям.

**Требования**

**Интерфейсы:**

* Repository<T>
* SortableRepository<T> extends Repository<T>
* FilterableRepository<T> extends SortableRepository<T>

**Сущности:**

* Smartphone
* Laptop
* Tablet

**Реализационные классы репозиториев:**

* SmartphoneRepository implements FilterableRepository<Smartphone>
* LaptopRepository implements FilterableRepository<Laptop>
* TabletRepository implements FilterableRepository<Tablet>

1. **Система Управления Онлайн Курсами**

Разработайте платформу для управления онлайн курсами, включающую различные типы курсов (видео-курсы, текстовые курсы, интерактивные курсы). Система должна позволять создавать, обновлять, удалять и искать курсы, а также сортировать их по рейтингу и популярности.

**Требования**

**Интерфейсы:**

* CourseRepository<T>
* SortableCourseRepository<T> extends CourseRepository<T>
* SearchableCourseRepository<T> extends SortableCourseRepository<T>

**Сущности:**

* VideoCourse
* TextCourse
* InteractiveCourse

**Реализационные классы репозиториев:**

* VideoCourseRepository implements SearchableCourseRepository<VideoCourse>
* TextCourseRepository implements SearchableCourseRepository<TextCourse>
* InteractiveCourseRepository implements SearchableCourseRepository<InteractiveCourse>

1. **Система Управления Медицинскими Записями**

Создайте систему для управления медицинскими записями пациентов, включая различные типы документов (анализы, выписки, рецепты). Система должна поддерживать создание, чтение, обновление и удаление записей, а также их сортировку и поиск по различным параметрам.

**Требования**

**Интерфейсы:**

* MedicalRepository<T>
* SortableMedicalRepository<T> extends MedicalRepository<T>
* SearchableMedicalRepository<T> extends SortableMedicalRepository<T>

**Сущности:**

* LabReport
* DischargeSummary
* Prescription

**Реализационные классы репозиториев:**

* LabReportRepository implements SearchableMedicalRepository<LabReport>
* DischargeSummaryRepository implements SearchableMedicalRepository<DischargeSummary>
* PrescriptionRepository implements SearchableMedicalRepository<Prescription>

1. **Система Управления Рестораном**

Разработайте систему для управления меню ресторана, включая различные категории блюд (закуски, основные блюда, десерты). Система должна позволять добавлять, удалять, обновлять и искать блюда, а также сортировать их по цене и популярности.

**Требования**

**Интерфейсы:**

* MenuRepository<T>
* SortableMenuRepository<T> extends MenuRepository<T>
* SearchableMenuRepository<T> extends SortableMenuRepository<T>

**Сущности:**

* Appetizer
* MainCourse
* Dessert

**Реализационные классы репозиториев:**

* AppetizerRepository implements SearchableMenuRepository<Appetizer>
* MainCourseRepository implements SearchableMenuRepository<MainCourse>
* DessertRepository implements SearchableMenuRepository<Dessert>

1. **Система Управления Проектами в IT Компании**

Создайте систему для управления проектами внутри IT компании, включающую различные типы проектов (веб-разработка, мобильные приложения, облачные решения). Система должна поддерживать создание, обновление, удаление и поиск проектов, а также их сортировку по срокам и статусу.

**Требования**

**Интерфейсы:**

* ProjectRepository<T>
* SortableProjectRepository<T> extends ProjectRepository<T>
* SearchableProjectRepository<T> extends SortableProjectRepository<T>

**Сущности:**

* WebDevelopmentProject
* MobileAppProject
* CloudSolutionProject

**Реализационные классы репозиториев:**

* WebDevelopmentProjectRepository implements SearchableProjectRepository<WebDevelopmentProject>
* MobileAppProjectRepository implements SearchableProjectRepository<MobileAppProject>
* CloudSolutionProjectRepository implements SearchableProjectRepository<CloudSolutionProject>

1. **Система Управления Студенческими Записями**

Разработайте систему для управления студенческими записями, включающую различные типы документов (зачетки, дипломы, курсовые работы). Система должна позволять добавлять, удалять, обновлять и искать записи, а также сортировать их по дате и оценкам.

**Требования**

**Интерфейсы:**

* StudentRecordRepository<T>
* SortableStudentRecordRepository<T> extends StudentRecordRepository<T>
* SearchableStudentRecordRepository<T> extends SortableStudentRecordRepository<T>

**Сущности:**

* GradeBook
* Thesis
* CourseWork

**Реализационные классы репозиториев:**

* GradeBookRepository implements SearchableStudentRecordRepository<GradeBook>
* ThesisRepository implements SearchableStudentRecordRepository<Thesis>
* CourseWorkRepository implements SearchableStudentRecordRepository<CourseWork>

1. **Система Управления Финансовыми Инструментами**

Создайте систему для управления различными финансовыми инструментами, включая акции, облигации и инвестиционные фонды. Система должна поддерживать операции добавления, удаления, обновления и поиска инструментов, а также их сортировку по стоимости и доходности.

**Требования**

**Интерфейсы:**

* FinancialInstrumentRepository<T>
* SortableFinancialInstrumentRepository<T> extends FinancialInstrumentRepository<T>
* SearchableFinancialInstrumentRepository<T> extends SortableFinancialInstrumentRepository<T>

**Сущности:**

* Stock
* Bond
* InvestmentFund

**Реализационные классы репозиториев:**

* StockRepository implements SearchableFinancialInstrumentRepository<Stock>
* BondRepository implements SearchableFinancialInstrumentRepository<Bond>
* InvestmentFundRepository implements SearchableFinancialInstrumentRepository<InvestmentFund>

1. **Система Управления Событиями в Спортивном Клубе**

Разработайте систему для управления спортивными событиями, включая различные виды спорта (футбол, баскетбол, теннис). Система должна позволять создавать, обновлять, удалять и искать события, а также сортировать их по дате и популярности.

**Требования**

**Интерфейсы:**

* EventRepository<T>
* SortableEventRepository<T> extends EventRepository<T>
* SearchableEventRepository<T> extends SortableEventRepository<T>

**Сущности:**

* FootballMatch
* BasketballGame
* TennisTournament

**Реализационные классы репозиториев:**

* FootballMatchRepository implements SearchableEventRepository<FootballMatch>
* BasketballGameRepository implements SearchableEventRepository<BasketballGame>
* TennisTournamentRepository implements SearchableEventRepository<TennisTournament>

1. **Система Управления Электронными Документами**

Создайте систему для управления электронными документами, включая различные типы документов (PDF, DOCX, HTML). Система должна поддерживать операции добавления, удаления, обновления и поиска документов, а также их сортировку по дате создания и размеру.

**Требования**

**Интерфейсы:**

* DocumentRepository<T>
* SortableDocumentRepository<T> extends DocumentRepository<T>
* SearchableDocumentRepository<T> extends SortableDocumentRepository<T>

**Сущности:**

* PDFDocument
* WordDocument
* HTMLDocument

**Реализационные классы репозиториев:**

* PDFDocumentRepository implements SearchableDocumentRepository<PDFDocument>
* WordDocumentRepository implements SearchableDocumentRepository<WordDocument>
* HTMLDocumentRepository implements SearchableDocumentRepository<HTMLDocument>

1. **Система Управления Электронными Документами**

Разработайте систему для управления различными типами транспортных средств, включая автомобили, мотоциклы и велосипеды. Система должна позволять добавлять, удалять, обновлять и искать транспортные средства, а также сортировать их по цене и мощности.

**Требования**

**Интерфейсы:**

* VehicleRepository<T>
* SortableVehicleRepository<T> extends VehicleRepository<T>
* SearchableVehicleRepository<T> extends SortableVehicleRepository<T>

**Сущности:**

* Car
* Motorcycle
* Bicycle

**Реализационные классы репозиториев:**

* CarRepository implements SearchableVehicleRepository<Car>
* MotorcycleRepository implements SearchableVehicleRepository<Motorcycle>
* BicycleRepository implements SearchableVehicleRepository<Bicycle>

1. **Система Управления Обучающимися в Танцевальной Школе**

Создайте систему для управления записями обучающихся в танцевальной школе, включая различные стили танцев (балет, хип-хоп, сальса). Система должна поддерживать операции добавления, удаления, обновления и поиска учеников, а также их сортировку по уровню подготовки и возрасту.

**Требования**

**Интерфейсы:**

* StudentRepository<T>
* SortableStudentRepository<T> extends StudentRepository<T>
* SearchableStudentRepository<T> extends SortableStudentRepository<T>

**Сущности:**

* BalletStudent
* HipHopStudent
* SalsaStudent

**Реализационные классы репозиториев:**

* BalletStudentRepository implements SearchableStudentRepository<BalletStudent>
* HipHopStudentRepository implements SearchableStudentRepository<HipHopStudent>
* SalsaStudentRepository implements SearchableStudentRepository<SalsaStudent>

1. **Система Управления Запасами на Складском Помещении**

Разработайте систему для управления запасами на складе, включая различные категории товаров (электроника, одежда, продукты питания). Система должна позволять добавлять, удалять, обновлять и искать товары, а также сортировать их по количеству и дате поступления.

**Требования**

**Интерфейсы:**

* InventoryRepository<T>
* SortableInventoryRepository<T> extends InventoryRepository<T>
* SearchableInventoryRepository<T> extends SortableInventoryRepository<T>

**Сущности:**

* ElectronicsItem
* ClothingItem
* FoodItem

**Реализационные классы репозиториев:**

* ElectronicsItemRepository implements SearchableInventoryRepository<ElectronicsItem>
* ClothingItemRepository implements SearchableInventoryRepository<ClothingItem>
* FoodItemRepository implements SearchableInventoryRepository<FoodItem>

1. **Система Управления Контентом для Веб-Сайта**

Создайте систему для управления контентом веб-сайта, включая различные типы контента (статьи, блоги, новости). Система должна поддерживать операции добавления, удаления, обновления и поиска контента, а также их сортировку по дате публикации и популярности.

**Требования**

**Интерфейсы:**

* ContentRepository<T>
* SortableContentRepository<T> extends ContentRepository<T>
* SearchableContentRepository<T> extends SortableContentRepository<T>

**Сущности:**

* Article
* BlogPost
* NewsItem

**Реализационные классы репозиториев:**

* ArticleRepository implements SearchableContentRepository<Article>
* BlogPostRepository implements SearchableContentRepository<BlogPost>
* NewsItemRepository implements SearchableContentRepository<NewsItem>

1. **Система Управления Мероприятиями в Концертном Зале**

Разработайте систему для управления мероприятиями в концертном зале, включая различные типы мероприятий (концерты, театральные постановки, лекции). Система должна позволять добавлять, удалять, обновлять и искать мероприятия, а также их сортировку по дате и жанру.

**Требования**

**Интерфейсы:**

* EventRepository<T>
* SortableEventRepository<T> extends EventRepository<T>
* SearchableEventRepository<T> extends SortableEventRepository<T>

**Сущности:**

* Concert
* TheaterPlay
* Lecture

**Реализационные классы репозиториев:**

* ConcertRepository implements SearchableEventRepository<Concert>
* TheaterPlayRepository implements SearchableEventRepository<TheaterPlay>
* LectureRepository implements SearchableEventRepository<Lecture>