Вкладка 1

Отчет о лабораторной работе №4

Выполнил:
Петров Евгений Станиславович
Группа: 6204-010302D

Вкладка 2

1.Цель: Расширить возможности пакета для работы с функциями одной переменной добавив интерфейсы и классы для аналитически заданных функций, а также методы ввода и вывода табулированных функций.

Задачи:

1. Ввести базовый интерфейс `Function` и корректно инкапсулировать данные табулированных функций.

2. Реализовать аналитические функции: экспонента, логарифм, синус, косинус, тангенс, а также метафункции: сумма, произведение, композиция, сдвиг, масштабирование, возведение в степень.

3. Реализовать утилитарные классы `Functions` и `TabulatedFunctions` со статическими методами для комбинирования и табулирования функций, а также для текстового и бинарного ввода/вывода.

4. Обеспечить сериализацию табулированных функций с использованием интерфейсов `Serializable` и `Externalizable`.

5. Создать демонстрационную программу `Main`, проверяющую все сценарии задания, включая сравнение аналитических и табулированных функций и сохранение результатов в файлы.

6. Подготовить подробный отчёт по лабораторной работе с описанием архитектуры, теории, тестов и выводов.

---

2. Теоретические сведения

Функция одной переменной — отображение вида
$f: \mathbb{R} \to \mathbb{R}$,
характеризуемое областью определения, значениями и способом вычисления (аналитическим или табличным).

Табулированная функция представляется набором упорядоченных пар (x_i, y_i). Значения между соседними точками вычисляются методом линейной интерполяции. Требования к корректности:
– монотонность значений x,
– наличие не менее двух точек.

Интерфейсы и полиморфизм. Интерфейс `Function` обеспечивает единый способ взаимодействия с любыми функциями — как аналитическими, так и табулированными. Интерфейс `TabulatedFunction` расширяет его методами работы с узловыми точками. Это реализует принцип подстановки Лисков и принцип открытости/закрытости.

Аналитические функции. Классы `Exp`, `Log`, `Sin`, `Cos`, `Tan` реализованы на основе `Math`. Тригонометрические функции наследуются от общего базового класса `TrigonometricFunction`, что исключает дублирование кода.

Метафункции. Классы `Sum`, `Mult`, `Power`, `Scale`, `Shift`, `Composition` демонстрируют композицию объектов и паттерн «Декоратор»: они хранят ссылки на другие функции и

комбинируют их вычисления.

Табулирование. Метод `tabulate()` создаёт табличное представление функции на отрезке, равномерно распределяя точки. Значения вычисляются строго внутри области определения исходной функции.

Ввод и вывод.

- Бинарный формат (`DataInputStream`, `DataOutputStream`) — компактный и точный способ хранения чисел.
- Текстовый формат (`Writer`, `Reader`, `StreamTokenizer`) — удобен для анализа и редактирования человеком.
- Потоки не закрываются внутри методов, чтобы не нарушать управление ресурсами вызывающей стороны.

Сериализация.

- `ArrayTabulatedFunction` использует автоматическую сериализацию (`Serializable`).
- `LinkedListTabulatedFunction` реализует `Externalizable`, позволяя полностью контролировать формат записи списка.

---

3. Описание архитектуры программы

3.1. Структура проекта

```
src/
├── Main.java
└── functions/
    ├── Function.java
    ├── FunctionPoint.java
    ├── FunctionPointIndexOutOfBoundsException.java
    ├── InappropriateFunctionPointException.java
    ├── TabulatedFunction.java
    ├── ArrayTabulatedFunction.java
    ├── LinkedListTabulatedFunction.java
    ├── Functions.java
    ├── TabulatedFunctions.java
    ├── basic/
    │   ├── TrigonometricFunction.java
    │   ├── Sin.java
    │   ├── Cos.java
    │   ├── Tan.java
    │   ├── Exp.java
    │   └── Log.java
    └── meta/
        ├── Sum.java
        ├── Mult.java
        ├── Power.java
        ├── Scale.java
        ├── Shift.java
        └── Composition.java
```

3.2. Назначение ключевых классов

- **Function** — базовый интерфейс для всех функций.
- **FunctionPoint** — объект точки *(x, y)*, поддерживает клонирование и сериализацию.
- **ArrayTabulatedFunction** — хранение точек в массиве, высокая скорость доступа, реализует `Serializable`.
- **LinkedListTabulatedFunction** — двусвязный кольцевой список, реализует `Externalizable`.
- **Functions** — фабрика для создания метафункций.
- **TabulatedFunctions** — табулирование, ввод/вывод, сериализация.
- **Main** — демонстрация возможностей библиотеки.

3.3. Принципы проектирования

- **Инкапсуляция:** скрытие массива или списка точек; доступ только через методы.
- **Наследование:** базовые классы для групп функций.
- **Полиморфизм:** работа через интерфейсы, позволяющая подменять реализации.
- **Композиция:** метафункции строят новые функции на основе существующих.

---

4. Модульное тестирование

Проведено ручное тестирование по сценариям из `Main`.

Таблица тестов

№	Тест	Вход	Ожидаемый результат
1	Сравнение аналитического sin(x) и табулированного	10 точек	Разница соответствует ошибке интерполяции
2	Сумма квадратов $\sin^2(x)+\cos^2(x)$	10 и 25 точек	Чем больше точек, тем ближе результат к 1
3	Текстовый ввод/вывод экспоненты	write/read	Значения совпадают
4	Бинарный ввод/вывод ln(x)	output/input	Совпадение значений
5	Serializable	ln(exp(x))	Полное совпадение
6	Externalizable	список ln(exp(x))	Полное совпадение

---

5. Воспроизводимость

- Требуемая версия JDK: **OpenJDK 17** или новее.
- Команда компиляции:

javac $(find src -name "*.java") -d out

- Команда запуска:

java -cp out Main

---

6. Результаты работы программы

Программа формирует следующие файлы:

- `data/exp_tabulated.txt` — текстовое представление exp(x).
- `data/ln_tabulated.bin` — бинарный файл с табулированным логарифмом.
- `data/ln_exp_serial.bin` — сериализованная табулированная функция.
- `data/ln_exp_external.bin` — сериализация списка точек через `Externalizable`.

Дополнительно на консоль выводятся:

- значения аналитических функций,
- сравнение аналитических и табулированных функций,
- демонстрация работы метафункций,
- подтверждение корректной сериализации.

Возможные улучшения:

- внедрение JUnit-тестов;
- вынесение магических чисел;
- больше параметризации в утилитных методах табулирования;
- обработка интерполяции вне области определения (например, через исключения).

---

8. Вывод

В ходе выполнения работы был разработан полный пакет для работы с функциями одной переменной, включающий:

- аналитические и табулированные реализации функций;
- метафункции для сочетания и преобразования функций;
- механизмы табулирования, сериализации, текстового и бинарного ввода/вывода;

демонстрационную программу.
User program running

=== Analytic sin(x) ===

x=0,00 -> 0,000000

x=0,10 -> 0,099833

x=0,20 -> 0,198669

x=0,30 -> 0,295520

x=0,40 -> 0,389418

x=0,50 -> 0,479426

x=0,60 -> 0,564642

x=0,70 -> 0,644218

x=0,80 -> 0,717356

x=0,90 -> 0,783327

x=1,00 -> 0,841471

x=1,10 -> 0,891207

x=1,20 -> 0,932039

x=1,30 -> 0,963558

x=1,40 -> 0,985450

x=1,50 -> 0,997495

x=1,60 -> 0,999574

x=1,70 -> 0,991665

x=1,80 -> 0,973848

x=1,90 -> 0,946300

x=2,00 -> 0,909297

x=2,10 -> 0,863209

x=2,20 -> 0,808496

x=2,30 -> 0,745705

x=2,40 -> 0,675463

x=2,50 -> 0,598472

x=2,60 -> 0,515501

x=2,70 -> 0,427380

x=2,80 -> 0,334988

x=2,90 -> 0,239249

x=3,00 -> 0,141120

x=3,10 -> 0,041581

=== Analytic cos(x) ===

x=0,00 -> 1,000000

x=0,10 -> 0,995004

x=0,20 -> 0,980067

x=0,30 -> 0,955336

x=0,40 -> 0,921061

x=0,50 -> 0,877583

x=0,60 -> 0,825336

x=0,70 -> 0,764842

x=0,80 -> 0,696707

x=0,90 -> 0,621610

x=1,00 -> 0,540302

x=1,10 -> 0,453596

x=1,20 -> 0,362358

x=1,30 -> 0,267499

x=1,40 -> 0,169967

x=1,50 -> 0,070737

x=1,60 -> -0,029200

x=1,70 -> -0,128844

x=1,80 -> -0,227202

x=1,90 -> -0,323290

x=2,00 -> -0,416147

x=2,10 -> -0,504846

x=2,20 -> -0,588501

x=2,30 -> -0,666276

x=2,40 -> -0,737394

x=2,50 -> -0,801144

x=2,60 -> -0,856889

x=2,70 -> -0,904072

x=2,80 -> -0,942222

x=2,90 -> -0,970958

x=3,00 -> -0,989992

x=3,10 -> -0,999135

=== sin(x): analytic vs tabulated (10 points) ===

x=0,00 -> analytic=0,000000; tabulated=0,000000

x=0,10 -> analytic=0,099833; tabulated=0,097982

x=0,20 -> analytic=0,198669; tabulated=0,195963

x=0,30 -> analytic=0,295520; tabulated=0,293945

x=0,40 -> analytic=0,389418; tabulated=0,385907

x=0,50 -> analytic=0,479426; tabulated=0,472070

x=0,60 -> analytic=0,564642; tabulated=0,558234

x=0,70 -> analytic=0,644218; tabulated=0,643982

x=0,80 -> analytic=0,717356; tabulated=0,707935

x=0,90 -> analytic=0,783327; tabulated=0,771888

x=1,00 -> analytic=0,841471; tabulated=0,835841

x=1,10 -> analytic=0,891207; tabulated=0,883993

x=1,20 -> analytic=0,932039; tabulated=0,918022

x=1,30 -> analytic=0,963558; tabulated=0,952051

x=1,40 -> analytic=0,985450; tabulated=0,984808

x=1,50 -> analytic=0,997495; tabulated=0,984808

x=1,60 -> analytic=0,999574; tabulated=0,984808

x=1,70 -> analytic=0,991665; tabulated=0,984808

x=1,80 -> analytic=0,973848; tabulated=0,966204

x=1,90 -> analytic=0,946300; tabulated=0,932175

x=2,00 -> analytic=0,909297; tabulated=0,898147

x=2,10 -> analytic=0,863209; tabulated=0,862441

x=2,20 -> analytic=0,808496; tabulated=0,798488

x=2,30 -> analytic=0,745705; tabulated=0,734535

x=2,40 -> analytic=0,675463; tabulated=0,670582

x=2,50 -> analytic=0,598472; tabulated=0,594072

x=2,60 -> analytic=0,515501; tabulated=0,507908

x=2,70 -> analytic=0,427380; tabulated=0,421745

x=2,80 -> analytic=0,334988; tabulated=0,334698

x=2,90 -> analytic=0,239249; tabulated=0,236716

x=3,00 -> analytic=0,141120; tabulated=0,138735

x=3,10 -> analytic=0,041581; tabulated=0,040753

=== cos(x): analytic vs tabulated (10 points) ===

x=0,00 -> analytic=1,000000; tabulated=1,000000

x=0,10 -> analytic=0,995004; tabulated=0,982723

x=0,20 -> analytic=0,980067; tabulated=0,965446

x=0,30 -> analytic=0,955336; tabulated=0,948170

x=0,40 -> analytic=0,921061; tabulated=0,914355

x=0,50 -> analytic=0,877583; tabulated=0,864608

x=0,60 -> analytic=0,825336; tabulated=0,814862

x=0,70 -> analytic=0,764842; tabulated=0,764620

x=0,80 -> analytic=0,696707; tabulated=0,688404

x=0,90 -> analytic=0,621610; tabulated=0,612188

x=1,00 -> analytic=0,540302; tabulated=0,535972

x=1,10 -> analytic=0,453596; tabulated=0,450633

x=1,20 -> analytic=0,362358; tabulated=0,357141

x=1,30 -> analytic=0,267499; tabulated=0,263648

x=1,40 -> analytic=0,169967; tabulated=0,169931

x=1,50 -> analytic=0,070737; tabulated=0,070437

x=1,60 -> analytic=-0,029200; tabulated=-0,029056

x=1,70 -> analytic=-0,128844; tabulated=-0,128549

x=1,80 -> analytic=-0,227202; tabulated=-0,224761

x=1,90 -> analytic=-0,323290; tabulated=-0,318254

x=2,00 -> analytic=-0,416147; tabulated=-0,411747

x=2,10 -> analytic=-0,504846; tabulated=-0,504272

x=2,20 -> analytic=-0,588501; tabulated=-0,580488

x=2,30 -> analytic=-0,666276; tabulated=-0,656704

x=2,40 -> analytic=-0,737394; tabulated=-0,732920

x=2,50 -> analytic=-0,801144; tabulated=-0,794171

x=2,60 -> analytic=-0,856889; tabulated=-0,843917

x=2,70 -> analytic=-0,904072; tabulated=-0,893664

x=2,80 -> analytic=-0,942222; tabulated=-0,940984

x=2,90 -> analytic=-0,970958; tabulated=-0,958261

x=3,00 -> analytic=-0,989992; tabulated=-0,975537

x=3,10 -> analytic=-0,999135; tabulated=-0,992814

=== sin^2(x) + cos^2(x) based on 10-point tabulation ===

x=0,00 -> 1,000000

x=0,10 -> 0,975345

x=0,20 -> 0,970488

x=0,30 -> 0,985429

x=0,40 -> 0,984968

x=0,50 -> 0,970398

x=0,60 -> 0,975624

x=0,70 -> 0,999358

x=0,80 -> 0,975073

x=0,90 -> 0,970586

x=1,00 -> 0,985897

x=1,10 -> 0,984515

x=1,20 -> 0,970314

x=1,30 -> 0,975910

x=1,40 -> 0,998723

x=1,50 -> 0,974808

x=1,60 -> 0,970691

x=1,70 -> 0,986371

x=1,80 -> 0,984068

x=1,90 -> 0,970237

x=2,00 -> 0,976203

x=2,10 -> 0,998094

x=2,20 -> 0,974549

x=2,30 -> 0,970802

x=2,40 -> 0,986852

x=2,50 -> 0,983628

x=2,60 -> 0,970167

x=2,70 -> 0,976503

x=2,80 -> 0,997473

x=2,90 -> 0,974298

x=3,00 -> 0,970920

x=3,10 -> 0,987341

=== sin^2(x) + cos^2(x) based on 25-point tabulation ===

x=0,00 -> 1,000000

x=0,10 -> 0,996914

x=0,20 -> 0,995736

x=0,30 -> 0,996464

x=0,40 -> 0,999099

x=0,50 -> 0,997471

x=0,60 -> 0,995842

x=0,70 -> 0,996120

x=0,80 -> 0,998304

x=0,90 -> 0,998135

x=1,00 -> 0,996055

x=1,10 -> 0,995882

x=1,20 -> 0,997616

x=1,30 -> 0,998905

x=1,40 -> 0,996374

x=1,50 -> 0,995751

x=1,60 -> 0,997034

x=1,70 -> 0,999781

x=1,80 -> 0,996800

x=1,90 -> 0,995726

x=2,00 -> 0,996559

x=2,10 -> 0,999299

x=2,20 -> 0,997333

x=2,30 -> 0,995808

x=2,40 -> 0,996190

x=2,50 -> 0,998479

x=2,60 -> 0,997971

x=2,70 -> 0,995996

x=2,80 -> 0,995928

x=2,90 -> 0,997766

x=3,00 -> 0,998716

x=3,10 -> 0,996291

=== exp(x) text I/O verification ===

x=0 -> expected=1,000000; actual=1,000000

x=1 -> expected=2,718282; actual=2,718282

x=2 -> expected=7,389056; actual=7,389056

x=3 -> expected=20,085537; actual=20,085537

x=4 -> expected=54,598150; actual=54,598150

x=5 -> expected=148,413159; actual=148,413159

x=6 -> expected=403,428793; actual=403,428793

x=7 -> expected=1096,633158; actual=1096,633158

x=8 -> expected=2980,957987; actual=2980,957987

x=9 -> expected=8103,083928; actual=8103,083928

x=10 -> expected=22026,465795; actual=22026,465795

=== ln(x) binary I/O verification (x=0 ������ �� ������������ double) ===

x=4,900e-324* -> expected=-744,440072; actual=-744,440072

x=1 -> expected=0,000000; actual=0,000000

x=2 -> expected=0,693147; actual=0,693147

x=3 -> expected=1,098612; actual=1,098612

x=4 -> expected=1,386294; actual=1,386294

x=5 -> expected=1,609438; actual=1,609438

x=6 -> expected=1,791759; actual=1,791759

x=7 -> expected=1,945910; actual=1,945910

x=8 -> expected=2,079442; actual=2,079442

x=9 -> expected=2,197225; actual=2,197225

x=10 -> expected=2,302585; actual=2,302585

=== ln(exp(x)) Serializable verification ===

x=0 -> expected=0,000000; actual=0,000000

x=1 -> expected=1,000000; actual=1,000000

x=2 -> expected=2,000000; actual=2,000000

x=3 -> expected=3,000000; actual=3,000000

x=4 -> expected=4,000000; actual=4,000000

x=5 -> expected=5,000000; actual=5,000000

x=6 -> expected=6,000000; actual=6,000000

x=7 -> expected=7,000000; actual=7,000000

x=8 -> expected=8,000000; actual=8,000000

x=9 -> expected=9,000000; actual=9,000000

x=10 -> expected=10,000000; actual=10,000000

=== ln(exp(x)) Externalizable verification ===

x=0 -> expected=0,000000; actual=0,000000

x=1 -> expected=1,000000; actual=1,000000

x=2 -> expected=2,000000; actual=2,000000

x=3 -> expected=3,000000; actual=3,000000

x=4 -> expected=4,000000; actual=4,000000

x=5 -> expected=5,000000; actual=5,000000

x=6 -> expected=6,000000; actual=6,000000

x=7 -> expected=7,000000; actual=7,000000

x=8 -> expected=8,000000; actual=8,000000

x=9 -> expected=9,000000; actual=9,000000

x=10 -> expected=10,000000; actual=10,000000

User program finished

-

Все требования лабораторной работы выполнены. Проект демонстрирует грамотное применение ООП, интерфейсов, полиморфизма, обработки ошибок и механизмов Java I/O.

---

9. Список источников

1. Readme.md лабораторной работы №4.
2. Assignment.md лабораторной работы №4.
3. Документация Oracle по пакетам `java.io` и `java.lang`.