Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/12 Интеллектуальный анализ больших данных в системах поддержки принятия решений**

О Т Ч Е Т

по лабораторной работе № 3

Название: Классы, наследование, полиморфизм

Дисциплина: Языки программирования для работы с большими данными

Студент	ИУ6-23М		Д.В. Пешков
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			П.В. Степанов
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2024

Цель работы

Целью лабораторной работы является изучение наследования и полиморфизма в классах на языке Kotlin.

Задание

Вариант 1

10. Определить класс Булева матрица (BoolMatrix) размерности (n x m). Класс должен содержать несколько конструкторов. Реализовать методы для логического сложения (дизъюнкции), умножения и инверсии матриц. Реализовать методы для подсчета числа единиц в матрице и упорядочения строк в лексикографическом порядке.

1. Определить класс Вектор размерности n. Реализовать методы сложения, вычитания, умножения, инкремента, декремента, индексирования. Определить массив из m объектов. Каждую из пар векторов передать в методы, возвращающие их скалярное произведение и длины. Вычислить и вывести углы между векторами.

Вариант 2

Создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль.

10. Train: Пункт назначения, Номер поезда, Время отправления, Число мест (общих, купе, плацкарт, люкс). Создать массив объектов. Вывести: a) список поездов, следующих до заданного пункта назначения; b) список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа; c) список поездов, отправляющихся до заданного пункта назначения и имеющих общие места.

1. Student: id, Фамилия, Имя, Отчество, Дата рождения, Адрес, Телефон, Факультет, Курс, Группа. Создать массив объектов. Вывести: a) список студентов заданного факультета; b) списки студентов для каждого

факультета и курса; c) список студентов, родившихся после заданного года; d) список учебной группы.

Вариант 3

Создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString().

10. Создать объект класса Год, используя классы Месяц, День. Методы: задать дату, вывести на консоль день недели по заданной дате, рассчитать количество дней, месяцев в заданном временном промежутке.

11. Создать объект класса Сутки, используя классы Час, Минута. Методы: вывести на консоль текущее время, рассчитать время суток (утро, день, вечер, ночь).

Вариант 4

Построить модель программной системы.

10. Система Железнодорожная касса. Пассажир делает Заявку на станцию назначения, время и дату поездки. Система регистрирует Заявку и осуществляет поиск подходящего Поезда. Пассажир делает выбор Поезда и получает Счет на оплату. Администратор вводит номера Поездов, промежуточные и конечные станции, цены.

11. Система Факультатив. Преподаватель объявляет запись на Курс. Студент записывается на Курс, обучается и по окончании Преподаватель выставляет Оценку, которая сохраняется в Архиве. Студентов, Преподавателей и Курсов при обучении может быть несколько.

Ход работы

Составлены программы для выполнения всех требуемых задач. Каждая из задач была покрыта набором unit-тестов JUnit, был настроен CI для проверки прохождения тестов на каждое изменение в коде.

Фрагмент программного кода приведен в листинге 1.

Листинг 1 — Фрагмент задания 1 из варианта 1

```
package org.lab3

/**
 *  10 Определить класс Булева матрица (BoolMatrix) размерности (n x m). Класс должен содержать несколько конструкторов. Реализовать методы для логического сложения (дизъюнкции), умножения и инверсии матриц. Реализовать методы для подсчета числа единиц в матрице и упорядочения строк в лексикографическом порядке
 */
class BoolMatrix(private val n: Int, private val m: Int, private val matrix: Array<BooleanArray>) {

    constructor(n: Int, m: Int) : this(n, m, Array(n) { BooleanArray(m) })

    fun disjunction(other: BoolMatrix): BoolMatrix {
        require(n == other.n && m == other.m) { "Matrices must have the same dimensions for disjunction operation" }
        val result = Array(n) { BooleanArray(m) }
        for (i in 0 until n) {
            for (j in 0 until m) {
                result[i][j] = matrix[i][j] || other.matrix[i][j]
            }
        }
        return BoolMatrix(n, m, result)
    }

    fun multiplication(other: BoolMatrix): BoolMatrix {
        require(m == other.n) { "Number of columns in the first matrix must be equal to the number of rows in the second matrix for multiplication" }
        val result = Array(n) { BooleanArray(other.m) }
        for (i in 0 until n) {
            for (j in 0 until other.m) {
                for (k in 0 until m) {
                    result[i][j] = result[i][j] || (matrix[i][k] && other.matrix[k][j])
                }
            }
        }
```

```
        return BoolMatrix(n, other.m, result)
    }

    fun inversion(): BoolMatrix {
        val result = Array(n) { BooleanArray(m) }
        for (i in 0 until n) {
            for (j in 0 until m) {
                result[i][j] = !matrix[i][j]
            }
        }
        return BoolMatrix(n, m, result)
    }

    fun countTrue(): Int {
        return matrix.sumOf { row -> row.count { it } }
    }

    fun returnMatrix(): Array<BooleanArray> {
        return matrix
    }

    fun sortRowsLexicographically(): BoolMatrix {
        val result = matrix.sortedBy { it.joinToString("")
}.toTypedArray()
        return BoolMatrix(n, m, result)
    }
}

fun main() {
    println("Hello World!")
}
```

Полные программные коды программ доступны в репозитории: https://github.com/DPeshkoff/PLfBD.

Вывод

В ходе выполнения данной лабораторной работы были получены требуемые компетенции. Было выполнено ознакомление с наследованием и полиморфизмом в языке Kotlin.