Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина: Современные языки программирования

ЛАБОРАТОРНАЯ РАБОТА № 1

Вариант № 20

Выполнил:  Ратуцкий В.В.

гр. 310901

Проверил: Усенко Ф. В.

Минск 2024

***Тема занятия:*** «Использование языка программирования Kotlin»

***Цель:*** Выполнить разработку приложения с использованием языка программирования Kotlin

**Задание:** Преобразовать класс Бумага. Добавить в него поле содержимое, которое будет хранить написанное на бумаге. Добавить класс Ксерокс, который будет создавать копии с бумаги.

Диаграмма классов и диаграмма последовательности представлены на рисунках 1 и 2 соответственно.

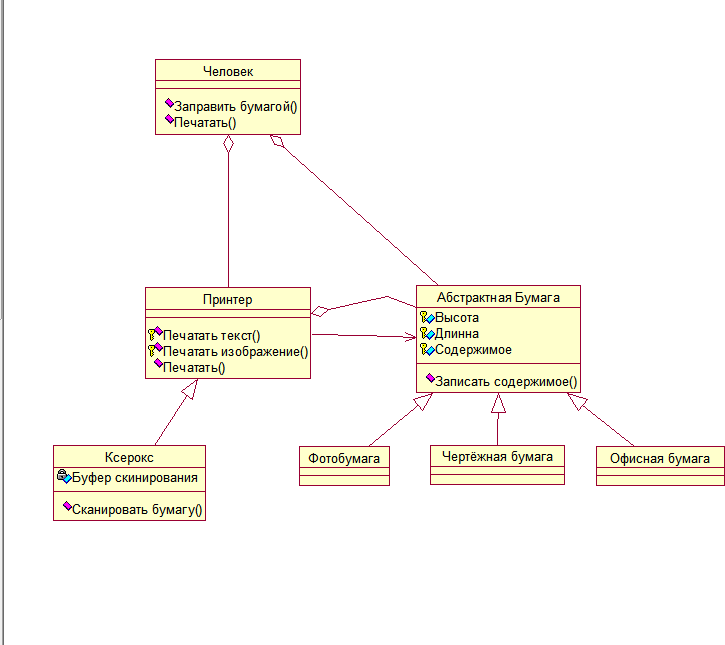


Рисунок 1 – Диаграмма классов

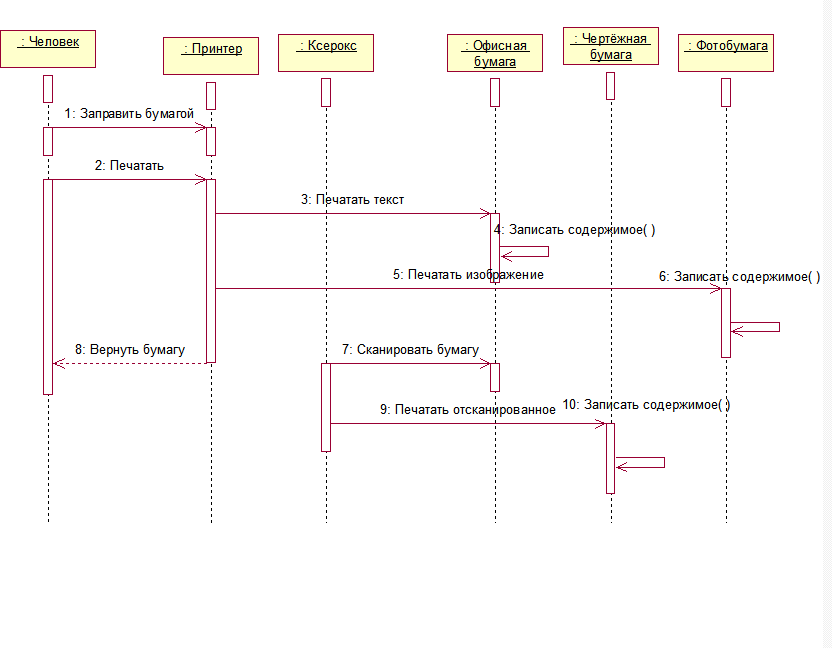


Рисунок 2 – Диаграмма последовательности

Структура программы представлена на рисунке 3.

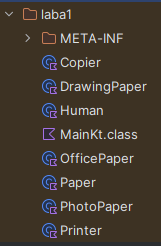


Рисунок 3

**Код программы:**

class Copier: Printer() {

private var buferScan: String = ""

fun scanOfficePaper(OfficePaper: Paper){

buferScan = "${OfficePaper.height} мм, ${OfficePaper.width} мм"

println("Бумага отсканирована")

println("Размер бумаги: $buferScan")

}

fun scanDrawingPaper(drawingPaper: DrawingPaper){

buferScan = "${drawingPaper.height} мм, ${drawingPaper.width} мм"

println("Бумага отсканирована")

println("Размер бумаги: $buferScan")

}

fun scanPhotoPaper(photoPaper: PhotoPaper){

buferScan = "${photoPaper.height} мм, ${photoPaper.width} мм"

println("Бумага отсканирована")

println("Размер бумаги: $buferScan")

}

}

public final class DrawingPaper public constructor(height: kotlin.Int, width: kotlin.Int) : Paper {

public open fun writeContent(): kotlin.Unit { /\* compiled code \*/ }

}

public final class Human public constructor() {

public final var paperAmount: kotlin.Int /\* compiled code \*/

public final fun loadPaper(): kotlin.Unit { /\* compiled code \*/ }

public final fun print(printer: Printer): kotlin.Unit { /\* compiled code \*/ }

}

public fun main(): kotlin.Unit { /\* compiled code \*/ }

public final class OfficePaper public constructor(height: kotlin.Int, width: kotlin.Int) : Paper {

public open fun writeContent(): kotlin.Unit { /\* compiled code \*/ }

}

public open class Paper public constructor(height: kotlin.Int, width: kotlin.Int) {

public final var height: kotlin.Int /\* compiled code \*/

public final var width: kotlin.Int /\* compiled code \*/

public open fun writeContent(): kotlin.Unit { /\* compiled code \*/ }

}

public final class PhotoPaper public constructor(height: kotlin.Int, width: kotlin.Int) : Paper {

public open fun writeContent(): kotlin.Unit { /\* compiled code \*/ }

}

public open class Printer public constructor() {

public final fun printText(textPaper: OfficePaper): kotlin.Unit { /\* compiled code \*/ }

public final fun printDrawing(drawingPaper: DrawingPaper): kotlin.Unit { /\* compiled code \*/ }

public final fun printImage(photoPaper: PhotoPaper): kotlin.Unit { /\* compiled code \*/ }

}

open class Paper( var height: Int, var width: Int) {

open fun writeContent(){

}

}

class OfficePaper(height: Int, width: Int): Paper(height, width){

override fun writeContent() {

println("Содержание текста загружено")

}

}

class DrawingPaper(height: Int, width: Int): Paper(height, width){

override fun writeContent(){

println("Содержание чертежа загружено")

}

}

class PhotoPaper(height: Int, width: Int): Paper(height, width){

override fun writeContent(){

println("Содержание изображения загружено")

}

}

class Human {

var paperAmount: Int = 0

fun loadPaper() {

paperAmount += 1

println("Количество бумаги: " + paperAmount)

}

fun print(printer: Printer){

if (paperAmount == 0){

println("В принтере нет бумаги")

println("1. Заправить бумагу\n2. Выключить принтер")

var b = readLine()?.toInt()

if (b == 1){

loadPaper()

} else if (b == 2){

return

} else {

println("Нет такого варианта ответа")

return

}

}

println("Вы хотите распечатать текст, изображение или чертеж?")

var a = readlnOrNull()

if (a == "Текст" || a == "текст"){

var textPaper = OfficePaper(210,297)

println("Размер бумаги: ${textPaper.height} мм, ${textPaper.width} мм")

paperAmount--

printer.printText(textPaper)

} else if (a == "Изображение" || a == "изображение"){

var photoPaper = PhotoPaper(210,297)

println("Размер бумаги: ${photoPaper.height} мм, ${photoPaper.width} мм")

paperAmount--

printer.printImage(photoPaper)

} else if (a == "Чертеж" || a == "чертеж" || a == "Чертёж" || a == "чертёж"){

var drawingPaper = DrawingPaper(210,297)

println("Размер бумаги: ${drawingPaper.height} мм, ${drawingPaper.width} мм")

paperAmount--

printer.printDrawing(drawingPaper)

} else {

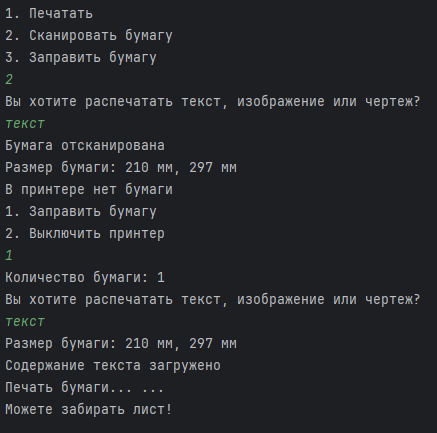
println("Нет такого варианта ответа. Выключение принтера...")

return

}

}

Работа программы представлена на рисунке 4.



Риснуок 4 – Работа программы