Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Кафедра инженерной психологии и эргономики

­­­­­

Современные языки программирования

Отчет по лабораторной работе №1

«Использование языка программирования Kotlin»

Выполнил: Касперец Е.А.

Студент группы 310902

Преподаватель: Усенко Ф. В.

Минск 2024

Цель: Выполнить разработку приложения с использованием языка программирования Kotlin.

Задание.Вариант 12.

Для классов кофе и сахар сделать родительский класс сыпучий предмет. Добавить наследующий от него класс Чай. Добавить класс молоко. Обеспечить класс Чайник методом Нагреть. Классу студент добавить метод Сделать кофе с молоком() и Сделать чай().

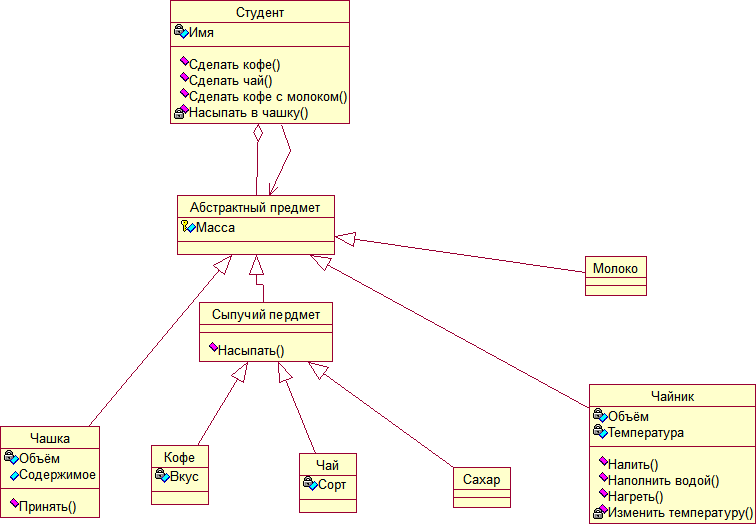


Рисунок 24 – Диаграмма классов

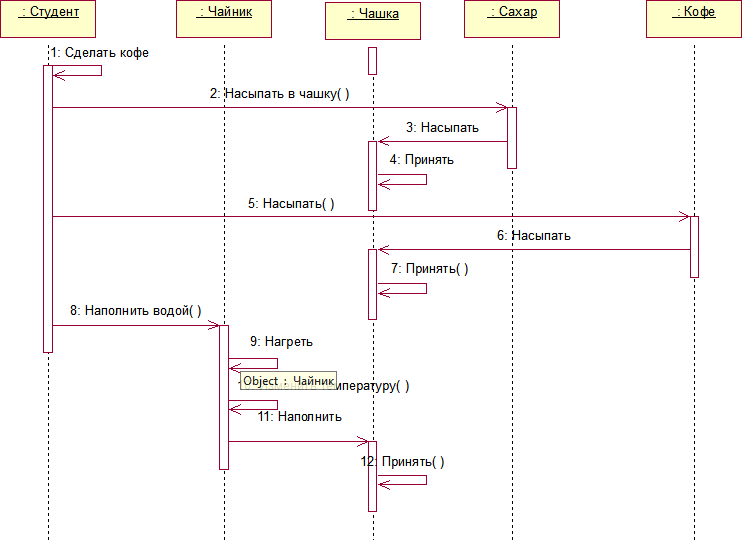


Рисунок 25 – Диаграмма последовательности

Листинг кода:

Файл Main.kt:

import student.\*

fun main(){

val our\_student = Student()

while (true) {

println("1 Сделать чай")

println("2 Сделать кофе")

println("3 Сделать кофе с молоком")

println("4 Выйти")

var choice: Int;

try {

choice = readln().toInt()

} catch (e: Exception) {

println("Неверный ввод")

continue;

}

when (choice) {

1 -> {

our\_student.create\_tea()

}

2 -> {

our\_student.create\_coffee()

}

3 ->{

our\_student.create\_coffee\_with\_milk()

}

4 -> {

break;

}

}

}

}

Файл Predmet.kt:

package predmet

abstract class Predmet{

abstract var mass:Int

}

abstract class loose\_predmet:Predmet(){

fun sprinkle(){

println("Введите количество ложек")

var mass = readLine()

}

}

class tea:loose\_predmet(){

override var mass = 100

val sorts = arrayOf("Черный", "Зеленый", "Улун")

}

class coffee:loose\_predmet(){

override var mass = 100

val flavors = arrayOf("Арабика", "Робуста", "Либерика ")

}

abstract class sugar:loose\_predmet()

abstract class milk:Predmet()

class teapot:Predmet(){

override var mass = 500

var volume = 1000

var temp:Int = 0

var volume\_puter = 0

fun temperature\_up(){

println("Чайник наполнен")

change\_temperature()

var loader = 0.0;

var index = 1.0

while(true){

if(loader < 100.0) {

loader = index / temp \* 100.0 / volume\_puter \* 1000.0

index += 1.0

Thread.sleep(1000)

val rounded\_loader = String.format("%.1f", loader)

println("Чайник нагрелся на: " + rounded\_loader + "%")

}

if(loader >= 100.0){

break

}

}

println("Чайник нагрелся")

}

fun put\_water(){

println("Введите количество воды");

var count\_of\_water:Int

try {

count\_of\_water = readln().toInt()

volume\_puter = count\_of\_water

if(count\_of\_water > volume){

println("Вода выливается из чайника, налито допустимое значение 1000 мл");

count\_of\_water = volume;

volume\_puter = 1000

}

else if(count\_of\_water<200){

println("Воды маловато, долейте еще")

while(true){

var temp = count\_of\_water

count\_of\_water = readln().toInt()

count\_of\_water += temp;

if(count\_of\_water> 200){

if(count\_of\_water > volume){

println("Вода выливается из чайника");

count\_of\_water = volume;

volume\_puter = 1000

}

volume\_puter = count\_of\_water

println("Отлично, продолжаем")

break

}

else{

println("ДОЛЕЙТЕ ЕЩЕ!!!")

continue

}

}

}

else {

volume = count\_of\_water

}

} catch (e: Exception) {

println("Неверный ввод")

put\_water()

}

}

fun change\_temperature(){

println("Введите желаемую температуру")

while (true) {

try {

var temperature = readln().toInt()

temp = temperature

if (temperature in 50..100) {

temp = temperature

break

} else {

println("Неверный ввод. Температура должна быть в диапазоне от 50 до 100.")

}

} catch (e: NumberFormatException) {

println("Неверный ввод. Введите целое число.")

}

}

}

}

class pot:Predmet(){

override var mass = 300

val volume = 300

var content = "Вода"

fun accept(index:String){

var order = mutableListOf<String>()

order.add(index)

println("Чашка приняла")

}

}

Файл Student.kt:

package student

import predmet.\*

abstract class student\_actions:loose\_predmet() {

abstract fun create\_tea();

abstract fun create\_coffee();

abstract fun create\_coffee\_with\_milk();

}

val myTeapot = teapot()

val myPot = pot()

class Student:student\_actions(){

override var mass: Int = 0;

override fun create\_coffee(){

println("Сахар:")

super.sprinkle()

println("Выберите кофе")

println("1 Арабика")

println("2 Робуста")

println("3 Либерика")

val myCoffee = coffee()

var chosen: Int? = null

try {

chosen = readln().toIntOrNull()

if (chosen != null && (chosen < 0 || chosen > myCoffee.flavors.size)) {

throw Exception("Индекс вне допустимого диапазона.")

}

} catch (e: Exception) {

println("Неверный ввод:")

create\_coffee()

}

if (chosen != null) {

myPot.accept(myCoffee.flavors[chosen-1])

} else {

create\_coffee()

}

println("Кофе:")

super.sprinkle()

myTeapot.put\_water()

myTeapot.temperature\_up()

myPot.accept("Вода")

println("Ваш кофе готов")

}

override fun create\_tea(){

println("Сахар:")

super.sprinkle()

println("Выберите чай")

println("1 Черный")

println("2 Зеленый")

println("3 Улун")

val myTea = tea()

var chosen: Int? = null

try {

chosen = readln().toIntOrNull()

if (chosen != null && (chosen < 0 || chosen > myTea.sorts.size)) {

throw Exception("Индекс вне допустимого диапазона.")

}

} catch (e: Exception) {

println("Неверный ввод:")

create\_tea()

}

if (chosen != null) {

myPot.accept(myTea.sorts[chosen-1])

} else {

create\_tea()

}

println("Чай:")

super.sprinkle()

myTeapot.put\_water()

myTeapot.temperature\_up()

myPot.accept("Вода")

println("Ваш чай готов")

}

override fun create\_coffee\_with\_milk(){

println("Сахар:")

super.sprinkle()

println("Выберите кофе")

println("1 Арабика")

println("2 Робуста")

println("3 Либерика")

val myCoffee = coffee()

var chosen: Int? = null

try {

chosen = readln().toIntOrNull()

if (chosen != null && (chosen < 0 || chosen > myCoffee.flavors.size)) {

throw Exception("Индекс вне допустимого диапазона.")

}

} catch (e: Exception) {

println("Неверный ввод:")

create\_coffee\_with\_milk()

}

if (chosen != null) {

myPot.accept(myCoffee.flavors[chosen-1])

} else {

create\_coffee\_with\_milk()

}

println("Кофе:")

super.sprinkle()

myTeapot.put\_water()

myTeapot.temperature\_up()

myPot.accept("Вода")

println("Молоко:")

super.sprinkle()

myPot.accept("Молоко")

println("Ваш кофе с молоком готов")

}

}

Результат работы программы представлены на рисунках 1 – 8.

# 

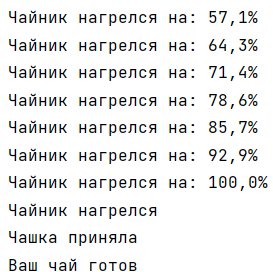


Рисунок 1 – Результат выполнения программы

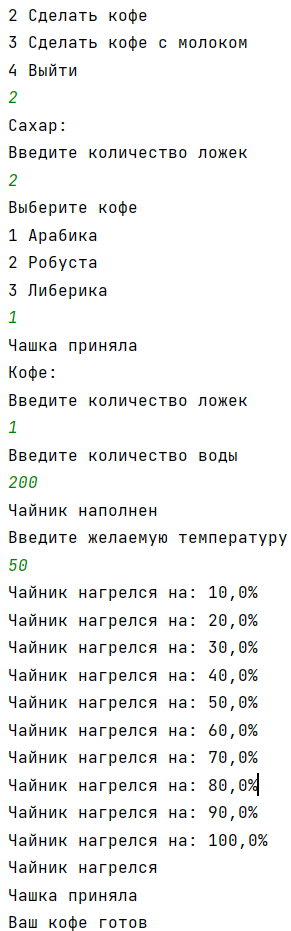


Рисунок 2 – Результат выполнения программы

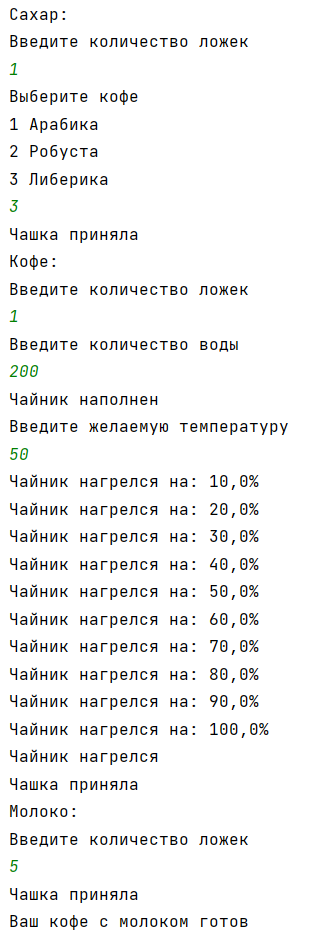


Рисунок 3 – Результат выполнения программы

Вывод: в ходе выполнения лабораторной работы номер 1 по языку программирования kotlin мы познакомились с основами языка и научились на практике применять основные его конструкции. Изучение UML диаграмм для корректного понимая задания лабораторной работы.