Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Дисциплина: современные языки программирования

**Лабораторная работа №1**

**«Использование языка программирования Kotlin»**

Вариант 15

Выполнил: Шабанович А.Б.

студент гр. 310102

Проверил: Усенко Ф.В.

Минск 2024

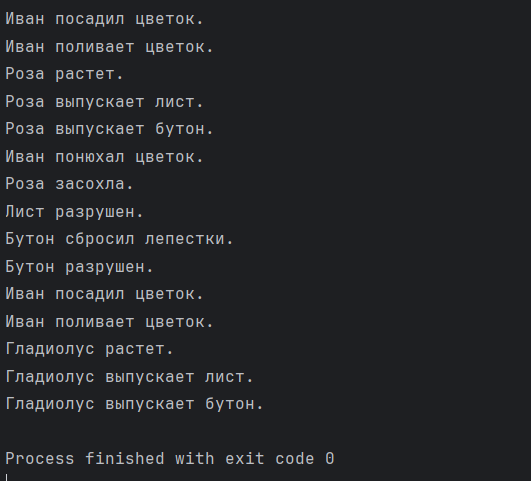
Цель: Выполнить разработку приложения с использованием языка программирования Kotlin.

Задание: Добавить класс Гладиолус. Добавить класс Лепесток, принадлежащий классу Бутон и класс Лист, принадлежащий классу Цветок. Обеспечить создание Объектов Лепесток одновременно с объектами Бутон. Реализовать метод Засохнуть() у класса Цветок.

Листинг кода:

package main  
  
// Абстрактный класс цветка  
abstract class AbstractFlower  
{  
 var height: Int = 0  
  
 abstract fun grow()  
 abstract fun releaseLeaf(): Leaf  
 abstract fun releaseBud(): Bud  
 abstract fun wither()  
}  
  
// Класс Роза  
class Rose : AbstractFlower()  
{  
 override fun grow()  
 {  
 *println*("Роза растет.")  
 height += 10  
 }  
  
 override fun releaseLeaf(): Leaf  
 {  
 *println*("Роза выпускает лист.")  
 return Leaf(5)  
 }  
  
 override fun releaseBud(): Bud  
 {  
 *println*("Роза выпускает бутон.")  
 return Bud("Розовый")  
 }  
  
 override fun wither()  
 {  
 *println*("Роза засохла.")  
 height = 0  
 }  
}  
  
// Класс Гладиолус  
class Gladiolus : AbstractFlower()  
{  
 override fun grow()  
 {  
 *println*("Гладиолус растет.")  
 height += 8  
 }  
  
 override fun releaseLeaf(): Leaf  
 {  
 *println*("Гладиолус выпускает лист.")  
 return Leaf(7)  
 }  
  
 override fun releaseBud(): Bud  
 {  
 *println*("Гладиолус выпускает бутон.")  
 return Bud("Красный")  
 }  
  
 override fun wither()  
 {  
 *println*("Гладиолус засох.")  
 height = 0  
 }  
}  
  
// Класс Лист  
class Leaf(val length: Int)  
{  
 fun destroy()  
 {  
 *println*("Лист разрушен.")  
 }  
}  
  
// Класс Бутон  
class Bud(val scent: String)  
{  
 val petal = Petal("Красный")  
  
 fun dropPetals()  
 {  
 *println*("Бутон сбросил лепестки.")  
 }  
  
 fun destroy()  
 {  
 *println*("Бутон разрушен.")  
 }  
}  
  
// Класс Лепесток  
class Petal(val color: String)  
{  
 fun destroy()  
 {  
 *println*("Лепесток разрушен.")  
 }  
}  
  
// Класс Человек  
class Human(val name: String)  
{  
 fun waterFlower(flower: AbstractFlower)  
 {  
 *println*("$name поливает цветок.")  
 flower.grow()  
 }  
  
 fun smellFlower(flower: AbstractFlower)  
 {  
 *println*("$name понюхал цветок.")  
 }  
  
 fun plantFlower(flower: AbstractFlower)  
 {  
 *println*("$name посадил цветок.")  
 }  
}  
  
// Основная функция  
fun main()  
{  
 val human = Human("Иван")  
 val rose = Rose()  
 val gladiolus = Gladiolus()  
  
 // Человек сажает цветок  
 human.plantFlower(rose)  
  
 // Человек поливает цветок  
 human.waterFlower(rose)  
  
 // Цветок выпускает лист и бутон  
 val leaf = rose.releaseLeaf()  
 val bud = rose.releaseBud()  
  
 // Человек нюхает цветок  
 human.smellFlower(rose)  
  
 // Засыхание  
 rose.wither()  
  
 // Уничтожение листа и бутона  
 leaf.destroy()  
 bud.dropPetals()  
 bud.destroy()  
  
 // Сажаем и поливаем Гладиолус  
 human.plantFlower(gladiolus)  
 human.waterFlower(gladiolus)  
 gladiolus.releaseLeaf()  
 gladiolus.releaseBud()  
}

Результат выполнения кода:



Вывод: разработана программа на языке программирования Kotlin с помощью двух диаграмм (диаграмма классов и диаграмма последовательности). В результате разработки программы на языке Kotlin была успешно реализована функциональность, позволяющая человеку посадить, полить и понюхать цветок, а цветку выпустить лист, выпустить бутон и др. Была проделана работа с функциями, классами, полями и методами.