МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «Белгородский государственный
национальный исследовательский университет»

Лабораторная работа №3
дисциплина «Современные методы, среды и языки программирования»
по теме «Создание модульного приложения в Java на основе объектно-ориентированного подхода»

Выполнил: студент группы 12002135 Макаров Д.С.
Проверил:

Белгород 2021

Лабораторная работа №3

«Создание модульного приложения в Java на основе объектно-ориентированного подхода»

Цель работы:Научиться выполнять декомпозицию решения типовых задач по разработке сложных систем, создавать библиотеки (классы, пакеты) и модули на языке Java для реализации декомпозированных решений.

Ход работы

Разработать модульное приложение, позволяющее вести учет ключевых объектов, имеющих некоторый вид связи (агрегация или композиция) с двумя другими объектами, находящимися друг с другом в такой же взаимосвязи, для одной из приведенных в конце предметных областей.

Реализовать в программе возможность вывода на экран следующей информации:

- список ключевых объектов, связанных с одним из представителей первого объекта;
- список ключевых объектов, связанных со каждым представителем второго объекта;
- список ключевых объектов, связанных с одним из представителей второго объекта.

Выбранная тема: информация о товарах на складе.

Исходный код программы см. в приложении.

Пример вывода программы:

```
Список продуктов:
 - holy-glitter
 - rough-sunset
 - dawn-silence
 - solitary-forest
 - cool-star
 - throbbing-fire
 - bitter-haze
 - lingering-cherry
 - frosty-sky
 - holy-river
 - summer-water
Список складов:
 - holy-glitter
 - rough-sunset
 - dawn-silence
 - solitary-forest
 - cool-star
 - throbbing-fire
Полный список товаров по складам:
склад holy-glitter
        секция 1
                holy-glitter x27
                cool-star x84
                throbbing-fire x91
                bitter-haze x93
                summer-water x40
        секция 2
                summer-water x75
                dawn-silence x8
                holy-river x29
склад rough-sunset
склад dawn-silence
склад solitary-forest
        секция 1
                rough-sunset x35
                dawn-silence x28
```

```
                cool-star x95
                lingering-cherry x52
                holy-river x76
        секция 2
                rough-sunset x26
                cool-star x23
                bitter-haze x1
                lingering-cherry x12
                holy-glitter x72
склад cool-star
        секция 1
                rough-sunset x42
                dawn-silence x24
                solitary-forest x59
                summer-water x55
        секция 2
                throbbing-fire x66
                bitter-haze x84
                lingering-cherry x76
                summer-water x57
                holy-glitter x32
                rough-sunset x1
                dawn-silence x9
                solitary-forest x54
склад throbbing-fire
        секция 1
                lingering-cherry x87
                rough-sunset x91
                dawn-silence x47
                cool-star x62
        секция 2
                dawn-silence x68
                throbbing-fire x87
                bitter-haze x93
                holy-river x54
                summer-water x91
        секция 3
                holy-glitter x3
                dawn-silence x9
                cool-star x9
                throbbing-fire x18
                lingering-cherry x77
                frosty-sky x73
```

Приложение

Содержимое файла main.go

```
package main

// Задание Информация о товарах на складе

import (
        "lab3/pkg/product"
        "lab3/pkg/warehouse"
)

func main() {
        productList := product.CreateMockProducts(10)
        warehouseList := warehouse.CreateMockWarehouse(5, productList)
        product.PrintProducts(productList)
        warehouse.PrintWarehouses(warehouseList)
        warehouse.PrettyPrintWarehouses(warehouseList)
}
```

Содержимое файла product.go

```
package product

import (
        "fmt"
        "time"

        "github.com/goombaio/namegenerator"
)

type Product struct {
        Name string
}

func (p Product) Print(prefix string) {
        fmt.Printf("%s%s", p.Name, prefix)
}

func CreateMockProducts(count int) []Product {
        resultNameSlice := make([]Product, 0, count)

        seed := time.Now().UTC().UnixNano()
        nameGenerator := namegenerator.NewNameGenerator(seed)

        for i := 0; i <= count; i++ {
                name := nameGenerator.Generate()
                resultNameSlice = append(resultNameSlice, Product{Name: name})
        }
        return resultNameSlice
}

func PrintProducts(productList []Product) {
        fmt.Println("Список продуктов:")
        for _, p := range productList {
                p.Print(" - ")
        }
}
```

Содержимое файла warehouse.go

```
package warehouse

import (
        "fmt"
        "lab3/pkg/product"
        "math/rand"
        "time"

        "github.com/goombaio/namegenerator"
)

// Задание Информация о товарах на складе

type Warehouse struct {
        name     string
```

```
        sections []WarehouseSection
}

func (w Warehouse) Print() {
        fmt.Printf("склад %s\n", w.name)
        for s_index, s := range w.sections {
                fmt.Printf("\tсекция %d\n", s_index+1)
                s.Print("\t\t")
        }
}

type WarehouseSection struct {
        products map[product.Product]int
}

func (s WarehouseSection) Print(prefix string) {
        for product, count := range s.products {
                fmt.Printf("%s%s x%d\n", prefix, product.Name, count)
        }
}

func CreateMockWarehouse(count int, productList []product.Product) []Warehouse {
        resultNameSlice := make([]Warehouse, 0, count)

        seed := time.Now().UTC().UnixNano()
        rand.Seed(time.Now().UTC().UnixNano())
        nameGenerator := namegenerator.NewNameGenerator(seed)

        for i := 0; i <= count; i++ {
                name := nameGenerator.Generate()
                warehouseSections := CreateMockWarehouseSection(rand.Intn(5), productList)
                resultNameSlice = append(resultNameSlice, Warehouse{name, warehouseSections})
        }
        return resultNameSlice
}

func CreateMockWarehouseSection(count int, productList []product.Product) []WarehouseSection {
        resultSlice := make([]WarehouseSection, 0, count)

        for i := 1; i < count; i++ {
                section := make(map[product.Product]int)
                for _, p := range productList {
                        if rand.Intn(2)-1 == 0 {
                                section[p] = rand.Intn(100)
                        }
                }
                resultSlice = append(resultSlice, WarehouseSection{section})
        }

        return resultSlice
}

func PrintWarehouses(warehouseList []Warehouse) {
        fmt.Println("Список складов:")
        for _, w := range warehouseList {
                fmt.Printf(" - %s\n", w.name)
        }
}

func PrettyPrintWarehouses(warehouseList []Warehouse) {
        fmt.Println("Полный список товаров по складам:")
        for _, w := range warehouseList {
                w.Print()
        }
}
```