МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

Факультет информационных технологий и программирования

**Лабораторная работа № 4**

Выполнил студент группы № M3307  
Бойцов Виталий

САНКТ-ПЕТЕРБУРГ  
2019

# Упражнение 4-1.Классы-коллекции.

Цель упражнения: Изучить преимущества использования классов-коллекций, предоставляемых в стандартной поставке JDK.

Описание упражнения: В этом упражнении вы реализуете класс ItemCatalog, хранящий список товаров, продаваемых интернет-магазином. Список в дальнейшем будет использоваться для демонстрации товаров покупателям на сайте магазина.

1. Добавьте в проект класс ItemCatalog, в котором будет реализована логика хранения списка товаров.
2. Включите в класс ItemCatalog поля, определенные следующим образом:   
   private HashMap<Integer,GenericItem> catalog =   
     new HashMap<Integer,GenericItem>();   
   private ArrayList<GenericItem> ALCatalog =   
     new ArrayList<GenericItem>();   
   *Примечание:* Обе эти коллекции будут хранить один и тот же список товаров. Коллекция HashMap более оптимальна для последующего поиска товаров в каталоге, а коллекция ArrayList нужна для сравнения с ней.
3. Реализуйте в классе ItemCatalog следующие методы:
   1. public void addItem(GenericItem item) : добавляет товар в каталог   
      public void addItem(GenericItem item) {   
        catalog.put(item.ID, item); // Добавляем товар в HashMap   
        ALCatalog.add(item); // Добавляем тот же товар в ArrayList   
      }
   2. public void printItems() : распечатывает товары из каталога на экране. Распечатку следует производить с использованием метода toString класса GenericItem.  
       public void printItems(){   
       for(GenericItem i : ALCatalog){   
        System.out.println(i);   
        }   
      }
   3. public GenericItem findItemByID(int id) : производит поиск в каталоге по переданному id товара. Поиск следует производит в коллекции catalog типа HashMap public GenericItem findItemByID(int id){   
      //Если нет такого ID, возвращаем пустое значение   
        if(!catalog.containsKey(id)) {   
        return null;   
        } else{   
        return catalog.get(id);   
        }  
      }
   4. public GenericItem findItemByIDAL(int id) : производит поиск в каталоге по переданному id товара. Поиск следует производит в коллекции ALCatalog типа ArrayList  
      public GenericItem findItemByIDAL(int id){   
        for(GenericItem i : ALCatalog){   
        if(i.ID==id) return i;   
        }   
        return null;   
      }
4. В методе main класса Main создайте новый экземпляр класса ItemCatalog. С помощью метода addItem добавьте в него несколько (порядка 10) товаров. С помощью приема из упр. 3-1 сравните скорость поиска по двум типам коллекций:   
   long begin = new Date().getTime();   
      
   for(int i=0; i<100000;i++)   
   cat.findItemByID(10);   
   long end = new Date().getTime();   
   System.out.println("In HashMap: "+(end-begin)); begin = new Date().getTime();   
   for(int i=0; i<100000;i++)   
     cat.findItemByIDAL(10);   
   end = new Date().getTime();   
   System.out.println("In ArrayList: "+(end-begin));
5. import javafx.util.Pair;  
   import ru.billing.exceptions.ItemAlreadyExistsException;  
     
   import java.util.ArrayList;  
   import java.util.HashMap;  
     
   public class ItemCatalog {  
    private HashMap<Integer, GenericItem> catalog =  
    new HashMap<Integer,GenericItem>();  
    private ArrayList<GenericItem> ALCatalog = new ArrayList<GenericItem>();  
     
    private HashMap<Pair<Category, Float>, ArrayList<GenericItem>> modified\_catalog =  
    new HashMap<Pair<Category, Float>, ArrayList<GenericItem>>();  
     
    public void addItem(GenericItem item) throws ItemAlreadyExistsException{  
    if (catalog.containsValue(item) || ALCatalog.contains(item))  
    throw new ItemAlreadyExistsException();  
    catalog.put(item.getID(), item);  
    ALCatalog.add(item);  
    ArrayList<GenericItem> temp = modified\_catalog.computeIfAbsent(new Pair<>(item.type, item.price), K->new ArrayList<>());  
    temp.add(item);  
    modified\_catalog.put(new Pair<>(item.type, item.price), temp);  
    }  
     
    public void printItems(){  
    for(GenericItem i : ALCatalog){  
    System.*out*.println(i);  
    }  
    }  
     
    public GenericItem findItemByID(int id){  
    if(!catalog.containsKey(id)) {  
    return null;  
    } else{  
    return catalog.get(id);  
    }  
    }  
     
    public GenericItem findItemByIDAL(int id){  
    for(GenericItem i : ALCatalog){  
    if(i.getID()==id) return i;  
    }  
    return null;  
    }  
     
    public ArrayList<GenericItem> findItemByParam(Float price, Category type){  
    if(!modified\_catalog.containsKey(new Pair<>(type, price))) {  
    return null;  
    } else {  
    return modified\_catalog.get(new Pair<>(type, price));  
    }  
    }  
     
   }

# Упражнение4-2. Абстрактные классы и интерфейсы.

Цель упражнения: Изучить полезные свойства абстрактных классов и интерфейсов.

Описание упражнения: В этом упражнении вы добавите в проект интерфейс CatalogLoader c методом load. Объект, реализующий метод load будет способен загружать список товаров в указанный каталог (ItemCatalog). В дальнейшем в нашем проекте появятся несколько классов реализующих метод load и заполняющих каталог из различных источников.

1. Добавьте в проект новый интерфейс CatalogLoader.
2. Опишите в интерфейсе CatalogLoader следующий метод:  
    public void load(ItemCatalog cat);
3. Добавьте в проект класс CatalogStubLoader, реализующий интерфейс CatalogLoader.
4. В методе load класса CatalogStubLoader реализуйте «ручной» способ загрузки каталога с помощью явно созданных программистом объектов:    
   GenericItem item1 = new GenericItem("Sony TV",23000,Category.GENERAL);   
   FoodItem item2 = new FoodItem("Bread",12,null,new Date(),(short)10);   
   cat.addItem(item1);   
   cat.addItem(item2);
5. В методе main класса Main создайте новый фрагмент кода, загружающий товары в каталог с помощью объекта-загрузчика:  
   CatalogLoader loader = new CatalogStubLoader();   
   loader.load(cat);
6. Запустите программу и проверьте корректность загрузки списка товаров.
7. package ru.billing.client;  
     
   import ru.billing.exceptions.CatalogLoadException;  
   import ru.billing.stocklist.ItemCatalog;  
     
   public interface CatalogLoader {  
    void load(ItemCatalog cat) throws CatalogLoadException;  
   }
8. package ru.billing.client;  
     
   import ru.billing.exceptions.CatalogLoadException;  
   import ru.billing.exceptions.ItemAlreadyExistsException;  
   import ru.billing.stocklist.Category;  
   import ru.billing.stocklist.FoodItem;  
   import ru.billing.stocklist.GenericItem;  
   import ru.billing.stocklist.ItemCatalog;  
     
   import java.util.Date;  
     
   public class CatalogStubLoader implements CatalogLoader {  
     
    @Override  
    public void load(ItemCatalog cat) throws CatalogLoadException {  
    GenericItem item1 = new GenericItem("Sony TV",23000, Category.*GENERAL*);  
    FoodItem item2 = new FoodItem("Bread",12,null,new Date(),(short)10);  
    try {  
    cat.addItem(item1);  
    cat.addItem(item2);  
    }  
    catch (ItemAlreadyExistsException e) {  
    e.printStackTrace();  
    throw new CatalogLoadException(e);  
    }  
     
     
    }  
   }

# Упражнение 4-3. Пакеты, модификаторы доступа и инкапсуляция.

Цель упражнения: Изучить на практике использование механизма пакетов и принципа инкапсуляции

Описание упражнения: В этом упражнении вы создадите пакеты в рамках проекта StockListProject и распределите классы и интерфейсы проекта по пакетам. Также вы создадите инкапсулированные версии классов проекта.

1. Инкапсулируйте классы GenericItem, FoodItem, TechnicalItem и ItemCatalog своего проекта. Для этого объявите все их поля как private, а для доступа к ним создайте соответствующие методы set… и get…  
   Это лучше сделать при помощи eclipse – Source/Generate Setters and Getters.
2. Создайте в проекте StockListProject пакеты со следующим содержанием:
   1. ru.billing.client
      1. Класс Main
      2. Интерфейс CatalogLoader
      3. Класс CatalogStubLoader
   2. ru.billing.exceptions
   3. ru.billing.stocklist
      1. Перечисление Category
      2. Класс GenericItem
      3. Класс FoodItem
      4. Класс TechnicalItem
      5. Класс ItemCatalog
   4. ru.lanit.warehouse
3. Исправьте ошибки, появившиеся в проекте из-за необходимости импорта классов из других пакетов.
4. Запустите программу, проверьте работоспособность проекта.

