МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІНСТИТУТ КОМП'ЮТЕРНИХ СИСТЕМ
КАФЕДРА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

ЛАБОРОТОРНА РОБОТА №2

з дисципліни «ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ»

на тему "Початок роботи з класами та об'єктами. Інкапсуляція. Конструктор."

Студента 2 курсу групи АД-181
Гежа Н. І.
Перевіряв
доцент Рудніченко Н. Д.

Одеса - 2019

ЗМІСТ

Введення

Ціллю даної лабораторної роботи є ознайомлення з основами написання класів, освоєння інкапсуляції класів, використання конструктора, навчитися використовувати модифікатори доступу.

Теоретична частина

Об'єктозорієнтоване програмування полягає у уяві програм як совокупність об’єктів які взаємодіють між собою, кожен з яких є екземпляром деякого класу.

Об’єкт є структурою яка складається з даних об’єкту та його методів (функцій). Кожен об’єкт описується класом, у якому зазначено його структуру. Данні об’єктів описують стан об’єкта, а методи є операціями які він може виконувати. Коли об’єкт А викликає метод об’єкту В, говориться що об’єкт А посилає повідомлення до об’єкту В, а відповіддю об’єкта В буде повернене значення.

Конструктор класу є методом що викликається при створенні класу, у ньому можливо виконати ініціалізацію даних, також можна конструктору передавати аргументи.

Інкапсуляція є одним із основоположних принципів ООП, вона дозволяє ізолювати стан та методи класу від зовнішнього коду. Це дозволяє розділити клас на інтерфейс та реалізацію.

- Інтерфейс — у інтерфейсі зібрано тільки усе що стосується взаємодії об’єкта з зовнішнім кодом, та він скриває деталі реалізації до яких доступ зовнішньому коду не потрібен
- Реалізація — реалізацією є внутрішні данні та методи, які забезпечують роботу класу, але не мають необхідності бути видними ззовні

У мові Java є 4 типа інкапсуляції: private (доступ тільки з самого класу), package-private (доступ тільки коду з пакету класу), protected (доступ класам пакету та підкласам), public (доступ усім).

Завдання №1

1. Создайте класс «Товар» (Item).

А) Класс «Товар» должен содержать следующие поля: наименование, цена.

Б) Класс «Товар» должен содержать конструктор, который принимает два параметра: наименование и начальную цену товара.

В) Класс «Товар» должен иметь следующие публичные методы: поднятие цены на определенный процент (значение процента типа float передается как аргумент метода); снижение цены на определенный процент (значение процента типа float передается как аргумент метода).

Г) В классе должна быть реализована проверка цены на отрицательное значение. Если в конструкторе передается отрицательное значение цены, либо в результате снижения цены на >100% цена становится отрицательной, она должна быть принудительно установлена в 0.

2. Создайте класс «Корзина» (Cart).

А) «Корзина» должна реализовывать структуру данных стек, в котором содержатся объекты класса «товар». Будем считать, что мы добавляем всегда по 1 единице товара. Стек «внутри» реализуется обычным массивом объектов Item. Класс должен содержать проверки, связанные с работой стека – переполнение стека, попытка извлечь элемент из пустого стека и т.д.

Б) «Корзина» должна содержать конструктор с 1 параметром – максимальным количеством элементов в стеке. В этом конструкторе происходила инициализация массива, который реализует стек.

В) «Корзина» должна содержать следующие публичные методы: добавление товара, удаление товара, подсчет суммы цен товаров в корзине (пройтись по элементам массива и сложить все значения цен), повышение и понижение цен всех товаров в стеке (два отдельных метода, значение цены передается как параметр метода. Необходимо пройтись по всем элементам массива и передать соответствующее сообщение объектам.

3. В методе Main необходимо создать объект класса «Корзина» с некоторым максимальным количеством элементов в стеке.

А) Заполнить корзину объектами класса Item (5-6 объектов будет достаточно);

Б) Вывести сумму цен товаров внутри корзины;

В) Поднять цены в корзине на 15%, вывести измененную сумму цен в консоль.

Г) Снизить цены в корзине на 30%, вывести измененную сумму цен в консоль.

Код програми у файлі Item.java:

```
package AD181.Gezha;

class Item {
  private String name;
  private float price;

  Item(String initialName, float initialPrice) {
    this.name = initialName;
    this.price = initialPrice;
    this.checkPrice();
  }

  void raisePrice(float percentile) {
    float coefficient = 1 + percentile / 100;
    this.alterPrice(coefficient);
  }

  void lowerPrice(float percentile) {
    float coefficient = 1 - percentile / 100;
    this.alterPrice(coefficient);
  }

  float getPrice() {
    return this.price;
  }

  String getName() {
    return this.name;
  }

  private void alterPrice(float coefficient) {
    this.price = this.price * coefficient;
    this.checkPrice();
  }

  // resets price to 0 if it is < 0
  private void checkPrice() {
    if (this.price < 0) {
      this.price = 0.0f;
    }
  }
}
```

Код програми у файлі Cart.java:

```
package AD181.Gezha;

import java.util.Stack;

class Cart {
  private Stack<Item> stack = new Stack<Item>();
  private int itemLimit;

  Cart(int maxItemsAmount) {
    this.itemLimit = maxItemsAmount;
  }

  boolean addItem(String name, float price) {
    if (this.stack.size() >= this.itemLimit) {
      return false;
    }
    this.stack.push(new Item(name, price));
    return true;
  }

  void removeItem() {
    if(this.stack.size() == 0) {
      return;
    }
    this.stack.pop();
  }

  int getItemLimit() {
    return this.itemLimit;
  }

  float getTotalPrice() {
    float totalPrice = 0.0f;
    for (Item item : this.stack) {
      totalPrice += item.getPrice();
    }
    return totalPrice;
  }

  void raisePrices(float percentile) {
    for (Item item : this.stack) {
      item.raisePrice(percentile);
    }
  }

  void lowerPrices(float percentile) {
    for (Item item : this.stack) {
      item.lowerPrice(percentile);
    }
  }
}
```

Код у файлі Main.java:

```
package AD181.Gezha;

import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    Cart cart = Main.createCart();
    Main.fillCart(cart);

    System.out.println("Total price: " + cart.getTotalPrice());
    cart.raisePrices(15);
    System.out.println("Total price after raising: " + cart.getTotalPrice());
    cart.lowerPrices(30);
    System.out.println("Total price after lowering: " + cart.getTotalPrice());
  }

  private static Cart createCart() {
    // requests item amount until an integer is given
    System.out.print("Input maximum items amount for the cart: ");
    Scanner console = new Scanner(System.in);
    while (!console.hasNextInt())
      console.nextLine();
    int itemAmount = console.nextInt();

    return new Cart(itemAmount);
  }

  private static void fillCart(Cart cart) {
    int itemLimit = cart.getItemLimit();
    System.out.println("Input " + itemLimit + " items");

    // fills the cart with entered items until it's full
    for (int i = 0; i < itemLimit; ++i) {
      Scanner console = new Scanner(System.in);
      System.out.println("Item " + (i + 1) + " / " + itemLimit);

      // requests name until a string is given
      System.out.print("Input new item name: ");
      String itemName = new String("");
      while (itemName.isEmpty())
        itemName = console.nextLine();

      // requests price until a float is given
      System.out.print("Input new item price: ");
      while (!console.hasNextFloat())
        console.nextLine();
      float itemPrice = console.nextFloat();

      // adds the item to cart, messages user on fail
      boolean operationSuccessful = cart.addItem(itemName, itemPrice);
      if (!operationSuccessful) {
        System.out.println("Failed to add new item (cart is full)");
      }
    }
  }
}
```

Результати виконання програми:

1. Була перевірена програма при вводі очікуваних значень (Рис. 1). Програма обробила їх правильно, ціни були змінені коректно.
2. Була перевірена програма при вводі поганих значень ціни та імені (Рис. 2). У першому товарі була введена негативна ціна, але програма змінила її на 0, оскільки видно що вона не була зарахована у загальній ціні. У другому товарі спочатку було введено нічого замість імені, а потім 2 строки замість ціни, програма обробила цей ввід чекаючи подальших коректних значень.

```
Input maximum items amount for the cart: 5
Input 5 items
Item 1 / 5
Input new item name: Licensed WinRar
Input new item price: 8
Item 2 / 5
Input new item name: Melon
Input new item price: 1
Item 3 / 5
Input new item name: Laptop
Input new item price: 1000
Item 4 / 5
Input new item name: Bottle of wine
Input new item price: 100
Item 5 / 5
Input new item name: Chips
Input new item price: 10
Total price: 1119.0
Total price after raising: 1286.85
Total price after lowering: 900.795
```

Рис. 1: Вивід 1

```
Input maximum items amount for the cart: 4
Input 4 items
Item 1 / 4
Input new item name: Licensed WinRar
Input new item price: -123
Item 2 / 4
Input new item name:
Melon
Input new item price: not a float
still not a float
1
Item 3 / 4
Input new item name: Laptop
Input new item price: 1000
Item 4 / 4
Input new item name: Bottle of wine
Input new item price: 100
Total price: 1101.0
Total price after raising: 1266.15
Total price after lowering: 886.305
```

Рис. 2: Вивід 2

Висновок

Були вивчені основи написання класів, інкапсуляції членів класу з використанням модифікаторів доступу, та використання конструктору.

Посилання на github репозиторій: https://github.com/onpu-ad181ng/gezha-oop-lab2

Список літератури

1. Приклади з опису лабораторної работи
2. Форум Stackoverflow