Міністерство освіти і науки України  
НТУУ «КПІ ім. І. Сікорського»

Кафедра автоматизації проектування енергетичних процесів і систем

Лабораторна робота №3

«Java ООП. Наслідування та відношення між класами»

Виконав: студент II курсу ТЕФ, групи TI-92

Сопронюк Ю. А.

Перевірив: Касьянов А. С.

Дата виконання: 19.11.2020

Київ-2020

**Варіант №11**

ЗАВДАННЯ

Створити ієрархію класів відповідно до предметної області свого варіанту. Повинні бути наявними:

- Щонайменш 1 абстрактний клас.

- Щонайменш 2 класи-нащадки — від абстрактного (обов’язково хоча б один) або іншого класу.

- Щонайменш 1 інтерфейс Java і його впровадження в вашому класі.

- Асоціація класів (будь яка, на ваш вибір — звичайна асоціація, агрегація, композиція). Реалізувати вивід на консоль певної (довільної) інформації про об’єкти своїх класів у зовнішній програмі.

***Варіант завдання: Туристичний набір.***

Контрольні запитання:

1. Що таке наслідування в ООП?
2. Чи є у класів Java якийсь спільний батьківський клас?

**ВИКОНАНАННЯ РОБОТИ**

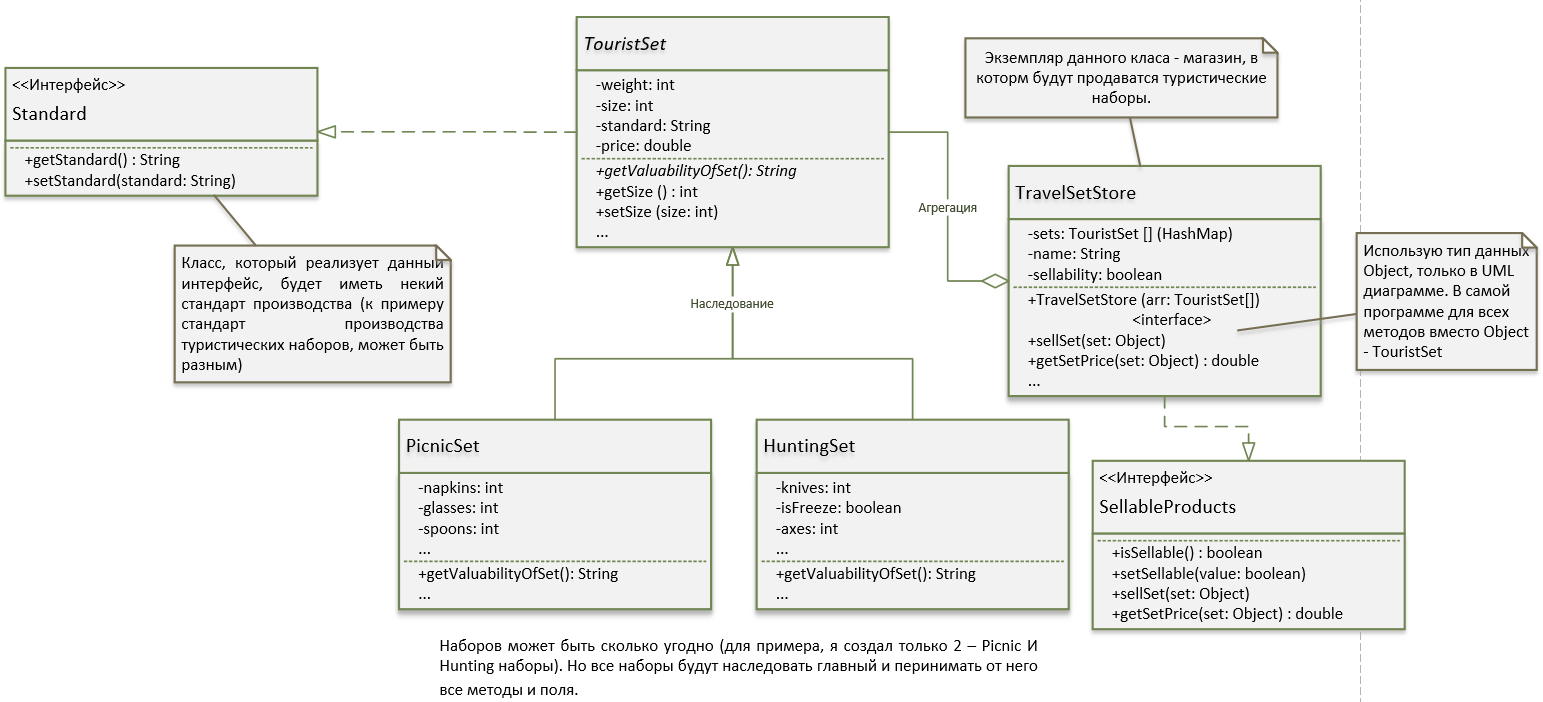
**Код програми**

**Усі необхідні файли проекту можна отримати за посиланням (ветка lab3):**

*https://github.com/steadyGuy/oop\_labs/tree/lab3*

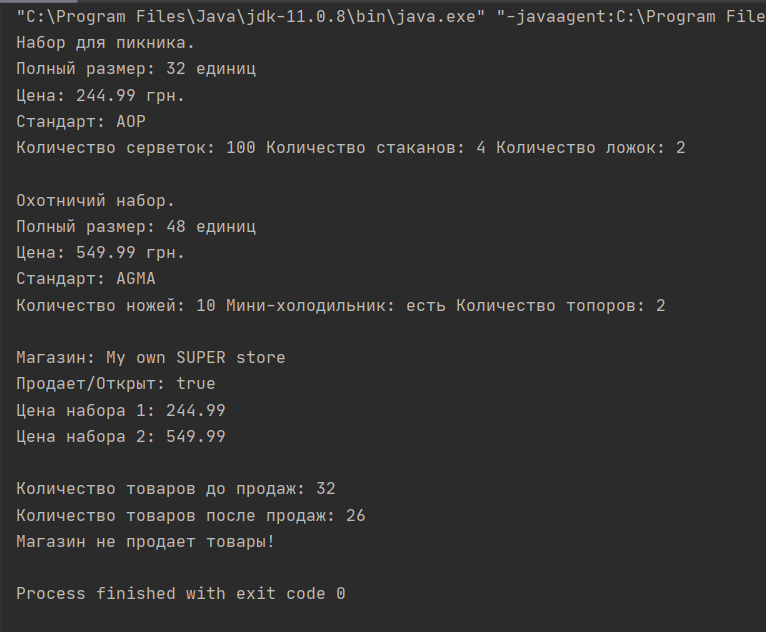
Крім основних файлів з розширенням *.java*, тут також знаходиться звіт у форматі *.docx*. *Main.java* – точка входу у програму.

**UML діаграма об’єктно орієнтованої системи**

****

Також, цю діаграму можна знайти в файлі *diagram.vsdx*

**Результат роботи програми**

**

*В консолі було виведено інформацію про два набори (які наслідують від основного набору), а також було продемонстровано роботу екземпляра класу* ***TravelSetStore****, для роботи з наборами в якості товарів магазину.*

**Контрольні запитання**

1) *Що таке наслідування в ООП?*

**Наслідування -** один з чотирьох найважливіших механізмів об'єктно-орієнтованого програмування (поряд з інкапсуаляцією, поліморфізмом та абстракцією), що дозволяє описати новий клас на основі вже існуючого (батьківського), при цьому властивості і функціональність батьківського класу запозичуються новим класом.

Іншими словами, клас, який є спадкоємцем (наследником) реалізує специфікацію вже існуючого класу (базового класу). Це дозволяє звертатися до об'єктів класу-спадкоємця так само, як до об'єктів базового класу.

2) *Чи є у класів Java якийсь спільний батьківський клас?*

Хоча ми можемо створити звичайний клас, який не є спадкоємцем, але фактично всі класи успадковуються від класу **Object**. Всі інші класи, навіть ті, які ми додаємо в свій проект, є неявно похідними від класу **Object**. Тому всі типи і класи можуть реалізувати ті методи, які визначені в класі **Object**.

**ВИСНОВОК**

На даній лабораторній роботі була створена програма мовою програмування Java, метою якої було створити набір класів, предметна область яких була пов’язана з *“туристичним набором”*. Під час написання класу було використано основні принципи ООП.

В основному було використано **наслідування** та **поліморфізм**. Під час написання програми, було створено абстрактний (який є базовим) клас *TouristSet*, від якого наслідуються інші класи (в моєму прикладі це *PicnicSet* і *HuntingSet*). При цьому усі властивості та функціональність батьківського класу були запозичені новими класами. Також програма реалізує інтерфейси (в моєму прикладі це *Standard* та *SellableProducts*), які реалізують відповідні їм класи. Можливість повторного використання коду – головна перевага, яку було отримано при використанні наслідування, адже деякі методи були повністю запозичені з базового класу, інші перевизначені разом із конструкторами. Це все значно зменшує розширення коду, адже тепер кожен новий клас буде наслідувати основну функціональність. Крім того, при створенні об’єктів, що наслідуються тепер можна використовувати лише один тип даних – назву батьківсього класу *TouristSet*, адже тут працює “восходящее преобразование” компілятором до базового типу.

При написанні програми було досліджено усі зв’язки, які можуть бути між різними класами. Існує всього 4 зв’язки між класами. В програмі було використано “асоціацію” між *TouristSet* та *TravelSetStore*, а саме її різновид – “*агрегацію*”. При агрегації реалізується слабкий зв'язок. У конструктор *TravelSetStore* передається посилання на вже наявну колекцію об'єктів *TouristSet*.

Всі відношення було зображено на діаграмі UML (включаючи наслідування, агрегацію та реалізацію).