Simple Tasks

Завдання: реалізувати одну програму на вибір.

3.1. Напишіть консольний додаток, використовуючи архітектурний шаблон MVC, який:

- описує інтерфейс **Drawable** з методом побудови фігури *draw()*;
- описує абстрактний клас **Shape**, який реалізує інтерфейс **Drawable** і містить поле ***shapeColor*** типу **String** для кольору фігури і конструктор для його ініціалізації, абстрактний метод обчислення площі фігури *calcArea()* і перевизначений метод *toString()*;
- описує класи Rectangle, Triangle, Circle, які успадковуються від класу Shape і реалізують метод calcArea (), а також перевизначають метод toString ();
- створює набір даних типу **Shape** (масив розмірністю не менш 10 елементів);
- обробляє масив:
    - відображає набір даних;
    - обчислює сумарну площу всіх фігур набору даних;
    - обчислює сумарну площу фігур заданого виду;
    - впорядковує набір даних щодо збільшення площі фігур, використовуючи об'єкт інтерфейсу **Comparator**;
    - впорядковує набір даних за кольором фігур, використовуючи об'єкт інтерфейсу **Comparator**.

Значення для ініціалізації об'єктів вибираються з заздалегідь підготовлених даних (обраних випадковим чином або по порядку проходження).

3.2. Напишіть консольний додаток, використовуючи архітектурний шаблон MVC, який:

- описує тип даних згідно таблиці;
- створює набір даних описаного типу (масив розмірністю не менше 10 елементів);
- використовує роботу з меню для обробки створеного масиву даних згідно з таблицею;
- використовує інтерфейс **Comparator** для впорядкування елементів масиву даних згідно з таблицею.

ВИМОГИ

1. Значення для ініціалізації об'єктів вибираються з заздалегідь підготовлених даних (обраних випадковим чином або по порядку проходження).
2. Дані для обробки масиву (пошук) вибираються з заздалегідь підготовлених даних випадковим чином.
3. Якщо в результаті обробки масиву даних не знайдено, то необхідно вивести відповідне повідомлення.
4. Для перевірки результату роботи потрібно вивести вихідний масив після створення.

Таблиця

Клас	*Обробка масиву даних*
Книга: Назва, Автор, Видавництво, Рік видання, Кількість сторінок, Ціна	1. Отримати список книг зазначеного автора; 2. Отримати список книг, які видані зазначеним видавництвом; 3. Отримати список книг, виданих пізніше вказаного року. 4. Відсортувати книги за видавництвами

Middle Tasks

****при виконанні завдань 3.3 чи 3.4 не треба буде виконувати 5.2***

Завдання: реалізувати одну програму на вибір.

3.3. Написати програму, яка моделює роботу вбудованого процесора турнікету лижного підйомника щодо перевірки доступу по ski-pass.

Турнікет контролює вхід лижників на підйомник по ski-pass. Ski-pass бувають наступних видів:

1. На робочі дні:
    a. Без обліку кількості підйомів: на півдня (з 9 до 13 або з 13 до 17), на день, два дні, 5 днів.
    b. По кількості підйомів: на 10 підйомів, на 20 підйомів, на 50 підйомів, на 100 підйомів.
2. На вихідні дні:

a. Без обліку кількості поїздок: на півдня (з 9 до 13 або з 13 до 17), на день, два дні.
b. По кількості підйомів: на 10 підйомів, на 20 підйомів, на 50 підйомів, на 100 підйомів.

3. Абонемент на сезон.

Турнікет повинен бути зв'язаний з системою, в який ведеться реєстр виданих карток. В цій системі можливо:

1. випустити ski-pass;
2. заблокувати ski-pass через порушення правил підйому.

Дані щодо картки зберігаються на самій картці, а саме: унікальний ідентифікатор, тип картки, термін дії, кількість поїздок тощо.

Турнікет зчитує дані з картки та виконує її перевірку. Якщо дані не вдалося зчитати, чи картка прострочена, чи заблокована, чи на ній не залишилося кредитів для поїздок, то прохід заборонено. Інакше з картки знімається одна поїздка (якщо для картки передбачається облік підйомів) і прохід дозволяється.

Турнікет здійснює облік дозволів та відмов проходу. При цьому турнікет вміє видавати по запиту 1) сумарні дані та 2) дані розбиті по типах ski-pass.

3.4. Написати програму, яка моделює роботу вбудованого процесору турнікету швидкісного трамваю.

Турнікет контролює вхід пасажирів на перон. Для проходу на перон можуть використовуватися проїзні картки, які поділяються на:

a. Учнівські, Студентські, Звичайні.
b. По терміну дії: на місяць, на 10 днів.
c. По кількості поїздок: на 5, на 10.
d. Накопичувальні картки (можуть бути тільки «звичайного» типу – не можуть бути учнівські та студентські) – можуть поповнюватися на певну суму, з якої при кожній поїздці знімається вартість проїзду. Такі картки не мають обмежень по терміну дії.

Турнікет повинен бути зв’язаний з системою, в який ведеться реєстр виданих карток. В цій системі можливо:

1. випустити проїзну картку.

Дані щодо картки зберігаються на самій картці, а саме: унікальний ідентифікатор, тип картки, термін дії, кількість поїздок тощо.

Турнікет зчитує дані з картки та виконує її перевірку. Якщо дані не вдалося зчитати, чи картка прострочена, чи на ній не залишилося кредитів для поїздок, то прохід заборонено. Інакше з картки знімається одна і прохід дозволяється.

Турнікет здійснює облік дозволів та відмов проходу. При цьому турнікет вміє видавати по запиту 1) сумарні дані та 2) дані розбиті по типах проїзних квитків.