Задачи к лекциям 26-28

1. Задача на цепочку обязанностей номер 1

Пишем тот же класс треугольника, к нему пишем цепочку с 1 методом в интерфейсе – рассчитать площадь. Первое звено – если треугольник равносторонний то высчитываем через формулу (a^2 * Math.sqrt(3))/ 4. Если нет, то отдаем другому звену. Второе звено для равнобедренного треугольника, сначала находим высоту через теорему Пифагора, после уже берем произведение половины стороны и высоты (0.5 * a * h) . Далее звено для прямоугольного треугольника – там просто половина произведения катетов. Ну и последнее звено общая формула которая описана в одной из лекций про треугольник. Найдите сами.

Дайте доступ к сторонам треугольника через геттеры для этой задачи чтобы не хранить методы определения типа треугольника. Или же напишите метод определения типа треугольника в классе треугольника который вернет enum и используйте его, а поля класса сделайте приватными. Решайте сами.

Сначала напишите классическим способом через сеттер следующего звена, после перепишите на метод с интерфейсом и private final first , second.

2. Задача на цепочку обязанностей номер 2

Давайте напишем цепочку из валидаторов для пароля. Первое звено должно проверить что входящая строка (String) не является пустой. Если проходим валидацию отдаем следующему звену на проверку, если же строка была пустой, то кидаем исключение с сообщением типа – пустой пароль. Далее добавляем еще звенья – валидатор минимальной длины – в конструктор получаем число для минимального количества и проверяем что пароль не меньше этой длины. Если меньше то бросаем исключение. Если удовлетворяет этому правилу, то отдаем ответственность следующему звену. Пишем еще одно звено для определения что среди символов есть как минимум одна цифра. Кстати, подумайте какие звенья похожи и как можно применить наследование.

3. Задача на цепочку обязанностей номер 3

Имитируем игру. Звенья цепи будут уровни. Игра типа квест. У каждого звена/уровня будет свой вопрос или несколько вопросов с какими-то верными ответами или же любой ответ будет допустим, но из списка предложенных. Все ответы нужно хранить в каком-то классе, можно в списке/массиве. Самое интересное будет то, что следующая цепочка будет проверять повторно ответ предыдущей цепи и исходя из этого задавать нужный вопрос. Как в сценарии. Каждый уровень зависит от итога предыдущего и сценарий будет свой.

Для примера – уровень первый. Вы оказались на перепутье, на камне надпись “налево пойдешь смерть найдешь, направо пойдешь – ногу потеряешь, прямо пойдешь – мудрость найдешь”. И нужно выбрать направление. Если выбрали налево, то там какой-то сценарий типа – вы встретили торговца, у него есть товары типа такой и такой, ваши действия и так далее. Важно чтобы на каждом шагу была вероятность выбрать смертельный исход, если не смертельный то идем по сценарию на следующий уровень.

4. Сложная задача с лямбдой и фабрикой

Самостоятельно изучить класс Timer, написать класс, который каждые N (рандом от 1 до 10) секунд добавляет в фабрику новостей новую с рандом заголовком и текстом (можете использовать шаблон). Написать колбек который наружу отдает событие когда в массиве новостей набирается 4 штуки и их обрабатывает редактор и в консоль пишет типа газету. Нумеровать выпуск газеты и при наступлении 10-ого выпуска прекратить.

5. Еще одна сложная задача на лямбду и фабрику

Написать класс грузчика у которого есть boolean isBusy флаг занятости. Он становится занятым если есть что разгрузить. Ему из фабрики каждые N секунд (от 2 до 8) приходит фура с товарами которые надо разгрузить. У фуры есть обьем, int значение количества товаров внутри. Имеем ввиду что нужно менять флаг занятости когда грузчик приступил к разгрузке фуры и менять флаг обратно когда он закончит разгрузку. Имейте ввиду что наш грузчик работает с мощностью 10 товаров в секунду. Значит нужно найти время за которое грузчик разгрузит фуру и с помощью класса Timer поменять ему флаг. Грузчик подписывается на новости о приезжающей фуре, но если в момент когда приехала новая фура он занят, то ничего не происходит. А когда он станет свободным, то сам должен уведомить фабрику фур об этом и получить доступ к первой же фуре которая ожидает. (просто список фур первый элемент если добавляем в конец).

6. Задача на список номер 1

Через консоль пишем строки. Нужно класть в список значений по длине строки. т.е. если первая строка от юзера была длиной в 3 символа, то при втором вводе в 4 строки нужно положить после первого, а если 2 символа, то положить перед первым. В конечном итоге у вас будет список отсортированных строк по длине. Можете использовать вспомогательный список из чисел где хранить длину строк. Прервать добавление строк на 10.

7. Сложная задача номер 3

Написать имитатор разносчика кофе. Добавить ему 3 наблюдателя. Когда он предлагает купить кофе все наблюдатели смотрят в свой кошелек и в зависимости от этого покупают или не покупают кофе. Назначить цену для кофе и добавить работнику 3 наблюдателя с разным кешем в кармане. Через разные рандомные промежутки времени предлагать кофе (от 2 секунд до 7). Каждому человеку хранить счетчик выпитых кофе, даже если у тебя много денег, то больше 2 раз не пить кофе. Кофе пьется 5 секунд. Пока оно пьется человек не может заказать еще один кофе.

8. Улучшем список

Написать класс SafeList который имплементирует интерфейс List и делает все вызовы методов безопасными. Так же не даем класть в список null и дубликаты элементов. При попытке получить элемент с индексом больше чем размер списка отдаем null. Т.е. стараемся сделать так, чтобы при работе с этим классом не возникло ни одной ошибки.

9. Еще несколько задач на списки

Написать класс который принимает аргументом список чисел. Добавить ему методы на нахождение минимума и максимума, метод сортировки по возрастанию и убыванию. Имейте ввиду что поле класса не должно меняться. Т.е. результат всех методов новый список.

Так же добавьте метод который находит в списке элемент и еще один который отдаст новый список состоящий из элементов больше переданного аргумента. Еще один метод на сумму всех членов списка. И еще один метод который заменит все нули в списке на Integer.MAX_VALUE.

10. Задача на компаратор

Написать класс с 2 числовыми полями. Создать список из этих элементов и отсортировать компоратором где пропишем логику – сначала должны идти те элементы в которых оба числа позитивны, после те в которых оба числа негативны и в конце уже все остальные.