**Задание № 5**

**Тема: Коллекции**

**Условие:**

1. Создайте ветку task-5 и переключитесь на нее из MASTER
2. Создайте проект task-5, в которой будет содержаться 2 подмодуля подмодуля: library, test.
3. В модуле library вам необходимо реализовать:
4. Iterator
5. Интерфейс
6. Реализация (-ии)
7. List
8. Интерфейс
9. Реализация листа как массив (аналог ArrayList) (назовите ArrayList)
10. Реализация листа как связанного списка (аналог LinkedList) (назовите LinkedList)
11. Stack
12. Интерфейс
13. Реализация как массив (назовите ArrayStack)
14. Реализация как связанный список (назовите ListStack)
15. Queue
16. Интерфейс
17. Реализация как массив (назовите ArrayQueue)
18. Реализация как связанный список (назовите ListQueue)
19. Entity – объект, содержащий пару <key, value>
20. Интерфейс
21. Реализацию
22. Map
23. Интерфейс
24. Реализация, где корзина реализована на основе массива (назовите ArrayMap)
25. Реализация, где корзина реализована на основе связанного списка (назовите ListMap)

Ваши коллекции должны иметь следующий функционал (вспомните **YAGNI**):

1. Iterator
   1. Итерироваться по List, Stack, Queue (метод getNext), возвращая объект коллекции
   2. Возвращать true, если имеется следующий элемент в коллекции, false – если последний (метод hasNext)
   3. Удалять элемент, на который ссылается итератор (remove)
   4. Вставлять новый объект перед и после объекта, на который ссылается итератор (addBefore, addAfter) (коллекции могут использовать сортировку, подумайте об этом)
2. List (+ реализации)
   1. Уметь работать с любым типом (кроме примитивного) (можно вставить null)
   2. Возвращать Iterator (метод getIterator) (так должны делать несколько коллекций, подумайте, как это оформить)
   3. Устанавливать компаратор (компаратор можно передать только в конструкторе)
   4. Автоматически сортировать объекты при вставке (add) используя компаратор, если компаратора нет – каким-либо образом. Возвращать индекс вставленного элемента
   5. Устанавливать максимальное количество элементов (метод setMaxSize), если не установлено – неограниченно количество
   6. Если установить MaxSize после того, как объект заполнен, причем, количество элементов в нем больше этого числа – обрезать все не влезающие элементы.
   7. Вставлять элементы в коллекцию, как по одному, так и массивом / коллекцией того же типа! (add, addAll) (используйте перегрузку методов). Из методов ..All не возвращать ничего
   8. Обновлять элемент по индексу (set). Возвращать старый элемент! Сортировать при обновлении!
   9. Удалять объекты (remove). Возвращая старый объект!
   10. Полностью очищать коллекцию (clear)
   11. Искать по элементу, возвращая номер позиции, или -1, если такого элемента нет(find) (не забываем – ваша коллекция отсортирована)
   12. Получать элементы по индексу (get)
   13. Отдавать массив элементов (toArray)
   14. Возвращать количество элементов (size)
   15. Удалять null-элементы (trim)
   16. Фильтровать сравнивая с коллекцией / массивом такого же типа, оставляя отличия/одинаковые элементы. (filterMatches, filterDifference)

Таким образом интерфейс List должен иметь как минимум методы:

List {

Iterator getIterator()

int add(элемент)

void addAll(коллекция / массив)

элемент set(позиция, элемент)

элемент remove(индекс)

void clear()

позиция find(Элемент)

элемент get(Позиция)

массив toArray(массив)

размер size()

void trim()

void filterMatches(Массив / Коллекция)

void filterDifference(Массив / Коллекция)

void setMaxSize(число)

void getMaxSize(число)

}

1. Stack (+ реализация)
   1. Уметь работать с любым типом (кроме примитивного) (нельзя вставить null)
   2. Возвращать Iterator (метод getIterator) (так должны делать несколько коллекций, подумайте, как это оформить)
   3. Возвращать true/false, если стэк пустой/не пустой (isEmpty)
   4. Получить верхний объект, без его удаления (peek)
   5. Получить верхний объект с удалением (pop)
   6. Добавить элемент в стэк (push). Сделать невозможным добавления уже имеющегося объекта (все объекты - уникальны)
   7. Вставка массива / коллекции того же типа в стэк (pushAll), не вставлять повторяющиеся элементы!
   8. Вернуть индекс объекта, если такой имеется в стэке, -1 – если нет (search)
   9. Очистить стэк (clear) Вернуть количество очищенных элементов.
   10. Вернуть размер стэка (size)
   11. Сортировать стэк по компаратору (sort)
   12. Итератор должен идти сверху вниз.

Таким образом интерфейс Stack должен иметь как минимум методы:

Stack {

Iterator getIterator()

Boolean isEmpty()

элемент peek()

элемент pop()

void push(Элемент)

void pushAll(коллекция / массив)

позиция search(элемент)

количество clear()

количество size()

void sort(компаратор)

}

1. Queue (+ реализация)
   1. Уметь работать с любым типом (кроме примитивного) (нельзя вставить null) (могут быть повторяющиеся объекты)
   2. Возвращать Iterator (метод getIterator) (так должны делать несколько коллекций, подумайте, как это оформить)
   3. Возвращать true/false, если очередь пустая/не пустая (isEmpty)
   4. Получить нижний объект, без его удаления (peek), если очередь пуста – выбросить исключение
   5. Получить нижний объект с удалением (poll)
   6. Получить верхний элемент без удаления (pull)
   7. Получить верхний элемент с удалением (remove)
   8. Вставить объект/массив/коллекцию в очередь (push, pushAll)
   9. Вернуть индекс объекта, если такой имеется в очереди, -1 – если нет (search)
   10. Очистить очередь (clear) Вернуть количество очищенных элементов.
   11. Вернуть размер очереди (size)
   12. Итератор должен идти снизу вверх.

Таким образом интерфейс Queue должен иметь как минимум методы:

Queue {

Iterator getIterator()

Boolean isEmpty()

элемент peek()

элемент poll()

элемент pull()

элемент remove()

void push(Элемент)

void pushAll(коллекция / массив)

позиция search(элемент)

количество clear()

количество size()

}

1. Entity (+ реализация)
   1. Уметь работать с любым типом (кроме примитивного) (нельзя вставить null, как на место ключа, так и значения)
   2. Содержать ключ / значение
   3. Возвращать ключ / значение (getKey, getValue)
   4. Установить ключ/ значение можно только через конструктор
   5. При вставке null – exception
2. Map
   1. Уметь работать с любым типом (кроме примитивного) (нельзя вставить null) (не могут быть повторяющиеся объекты) + Entity
   2. Обе реализации должны основываться на hash-е ключа.
   3. Возвращать true/false, если map пустая/не пустая (isEmpty)
   4. Добавлять значения как по ключу/значению так и по Entity (set)
   5. Выбрасывать исключение, если попытаться вставить по ключу, который уже есть в map
   6. Удаление объекта (ключ / значение) по ключу / Entity (remove). Возвращать удаленную Entity
   7. Получать List всех ключей (getKeys)
   8. Получать List всех значений (getValues)
   9. Возвращать true / false, если в map содержится / не содержится value (contains)
   10. Получать значение / Entity по ключу (get / getEntity)
   11. Вернуть размер map (size)
   12. Очистить map (clear) Вернуть количество очищенных элементов.

Таким образом интерфейс Map должен иметь как минимум методы:

Map {

Boolean isEmpty()

void set(key, value / Entity)

Entity remove(key / Entity)

List getKeysl()

List getValues()

value get(Ключ)

Entity getEntity(Ключ)

Boolean contains(value)

количество clear()

количество size()

}

1. Использовать стандартные коллекции запрещено! Вы можете использовать только свои реализации!
2. Второй модуль (test) должен содержать всеразличные действия с вашими коллекциями:
   1. Продемонстрируйте работу каждого из методов на используя тип Integer (<String, Integer> для Map)
   2. Продемонстрируйте работу с объектом User у которого есть поля: String name, String e,ail, Integer age, Entity sex (male, female). (<String, User> для Map)
   3. Модуль должен подключать первый модуль как библиотеку.
3. Если есть необходимость, можете добавлять свои функции / функциональность, но, должно быть реализовано то, что описано в задании.
4. Вторая jar (модуля test) должна запускаться из коробки – если используете зависимости, они должны быть уже в jar.
5. Версия java для компиляции приложения – 1.8
6. Изменить README.md добавив в него строку “Задание #5: Привет мои коллекции – Date: ” + 2 июля 2019 04:07 (дата, месяц, год, час, минута) (естественно, тут ваша дата)
7. Закомитать ваш код в ветку task-5 и запушить на удаленный сервер.
8. Создать pull request