|  |
| --- |
| , RD Dep. |
| **Практическое задание**  **JAVA.SE.06 Generic and Collections** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **REVISION HISTORY** | | | | | |
| **Ver.** | **Description of Change** | **Author** | **Date** | **Approved** | |
| **Name** | **Effective Date** |
| <1.0> | Первая версия | Игорь Блинов | <04.08.2011> |  |  |
| <2.0> | Вторая версия: задания изменены согласно обновленному содержанию модуля | Ольга Смолякова | <12.11.2014> |  |  |

**Задание 1. Классы коллекций**

Изучите классы реализации коллекций и заполните следующую таблицу

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Ordering | Random Access | Key-Value Pairs | Allows Duplicates | Allows Null Values | Thread Safe | Blocking Operations |
| ArrayList | Yes | Yes | No | Yes | Yes | No | No |
| HashMap | No | Yes | Yes | No | Yes | No | No |
| Vector | Yes | Yes | No | Yes | Yes | Yes | No |
| Hashtable | No | Yes | Yes | No | No | Yes | No |
| HashSet | No | Yes | No | No | Yes | No | No |
| TreeSet | Yes | Yes | No | No | No | No | No |
| LinkedList | Yes | No | No | Yes | Yes | No | No |
| ArrayDeque | Yes | Yes | No | Yes | No | No | No |
| Stack | Yes | No | No | Yes | Yes | Yes | No |
| TreeMap | Yes | Yes | Yes | No | No | No | No |
| WeakHashMap | No | Yes | Yes | No | Yes | No | No |
| Arrays | Yes | Yes | No | Yes | Yes | No | No |
| Properties | No | Yes | Yes | No | No | Yes | No |
| CopyOnWriteArrayList | Yes | Yes | No | Yes | Yes | Yes | No |
| ConcurrentHashMap | No | Yes | Yes | No | No | Yes | No |
| ConcurrentSkipListMap | Yes | Yes | Yes | No | No | Yes | No |
| ConcurrentSkipListSet | Yes | No | No | No | No | Yes | No |
| CopyOnWriteArraySet | Yes | Yes | No | No | Yes | Yes | No |
| ConcurrentLinkedQueue | Yes | No | No | Yes | No | Yes | No |
| ConcurrentLinkedDeque | Yes | No | No | Yes | No | Yes | No |
| ArrayBlockingQueue | Yes | No | No | Yes | No | Yes | Yes |
| LinkedBlockingQueue | Yes | No | No | Yes | No | Yes | Yes |
| LinkedTransferQueue | Yes | No | No | Yes | No | Yes | Yes |
| PriorityBlockingQueue | Yes | No | No | Yes | No | Yes | No |
| LinkedBlockingDeque | Yes | No | No | Yes | No | Yes | Yes |
| SynchronousQueue | Yes | No | No | Yes | No | Yes | No |
| DelayQueue | Yes | No | No | Yes | No | Yes | No |

**Задание 2. Использование Map**

Создать “универсальный” класс, позволяющий получить значение из любого properties-файла. Физическое чтение файла должно происходить только один раз. Результаты чтения храните в коллекции типа Map. Ответьте на вопрос: как ведет себя map-коллекция если в нее добавить элемент с ключом, который уже присутствует?

**Задание 3. Ссылки на коллекции**

Определена иерархия классов

**class** MedicalStaff{}

**class** Doctor **extends** MedicalStaff{}

**class** Nurse **extends** MedicalStaff{}

**class** HeadDoctor **extends** Doctor{}

Укажите корректные и некорректные операторы. Дайте ответу пояснение.

|  |  |  |
| --- | --- | --- |
|  | correct | not correct |
| Doctor doctor1 = **new** Doctor(); | + |  |
| Doctor doctor2 = **new** MedicalStaff(); |  | + |
| Doctor doctor3 = **new** HeadDoctor(); | + |  |
| Object object1 = **new** HeadDoctor(); | + |  |
| HeadDoctor doctor5 = **new** Object(); |  | + |
| Doctor doctor6 = **new** Nurse(); |  | + |
| Nurse nurse = **new** Doctor(); |  | + |
| Object object2 = **new** Nurse(); | + |  |
|  |  |  |
| List<Doctor> list1= **new** ArrayList<Doctor>(); | + |  |
| List<MedicalStaff> list2 = **new** ArrayList<Doctor>(); |  | + |
| List<Doctor> list3 = **new** ArrayList<MedicalStaff>(); |  | + |
| List<Object> list4 = **new** ArrayList<Doctor>(); |  | + |
| List<Object> list5 = **new** ArrayList<Object>(); | + |  |

**Задание 4. Применение коллекций**

Заполните таблицу.

|  |  |  |
| --- | --- | --- |
|  | Основная функциональность | Примеры типичного использования |
| Set | Реализация интерфейса Set представляет собой неупорядоченную коллекцию, которая не может содержать дублирующие данные | Множества часто используются для проверки принадлежности, чтобы проверить, принадлежит ли объект заданному множеству, поэтому на практике обычно выбирается реализация HashSet,оптимизированная для быстрого поиска. |
| List | List - упорядоченная коллекция иногда называемая списком или последовательностью. Список может содержать повторяющиеся элементы. | Когда необходимы следующие свойства коллекции:  1) Список может включать одинаковые элементы  2) Элементы в списке хранятся в том порядке, в котором они туда помещались. Самопроизвольных перемещений элементов в нем не происходит.  3) Можно получить доступ к любому элементу по его порядковому номеру/индексу внутри списка |
| Queue | Queue - коллекция, предназначенная для хранения элементов в порядке, нужном для их обработки. В дополнение к базовым операциям интерфейса Collection, очередь предоставляет дополнительные операции вставки, получения и контроля | Когда необходимо использовать свойство коллекции - возможность управления порядком элементов (FIFO и LIFO) |
| Map | Map позволяет искать объекты по ключу. Объект, ассоциированный с ключом, называется значением. И ключи, и значения являются объектами. Ключи могут быть уникальными, а значения могут дублироваться | Множество представляет собой набор данных, в котором можно быстро найти существующий элемент. Однако для этого нужно иметь точную копию требуемого элемента. Этот вид поиска не очень распространен, поскольку обычно известна лишь некоторая информация (ключ), по которой можно найти соответствующий элемент. Специально для этого предназначена структура данных,поддерживающая отображение, которую называют также картой. |