|  |
| --- |
| EPAM Systems, 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 | No | Yes | No | Yes | Yes | No | No |
| LinkedList | No | Yes | No | Yes | Yes | No | No |
| Vector | No | Yes | No | Yes | Yes | Yes | Yes |
| Stack | No | No | No | Yes | Yes | No | Yes |
| HashSet | No | No | Yes | No | Yes | No | No |
| LinkedHashSet | Yes | No | Yes | No | Yes | No | No |
| TreeSet | Yes | No | Yes | No | Yes | No | 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?

Старый элемент заменяется новым.

##### **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 | Нет повторяющихся элементов | HashSet, TreeSet, LinkedHashSet |
| List | Упорядоченная коллекция | ArrayList, LinkedList, Stack, Vector |
| Queue | Размещение элемента перед его обработкой | LinkedList, ArrayDequeue |
| Map | Хранение пар ключ-значение | HashMap, HashTable, TreeMap, LinkedHashMap |