Теоретический материал для выполнения ИЗ №5

Гутников С.Е.

Условие ИЗ №5

1) Создать класс (классы), указанный в задании. По возможности использовать assert и исключения для обработки ошибочных ситуаций.
2) В отдельном файле разработать тестовое приложение, использующее класс (классы), указанный в задании. Провести тестирование всех методов и конструкторов с выводом данных и результатов

Условие ИЗ №5

В каждом варианте задания есть следующее условие:

Класс должен реализовать:

-интерфейс Comparable и Comparator с возможностью выбора одного из полей для сравнения

-интерфейс Iterable - индексатор по всем полям объекта

-метод для сохранения значений всех полей в строке текста (переопределить toString())

-конструктор или метод для инициализации объекта из строки текста, соответствующий реализации метода toString()

Условие ИЗ №5

Обратите внимание, что в последнем пункте условия приведено требование совместимости конструктора с единственным строковым аргументом и метода toString() класса, т.е. результат toString() можно подать в конструктор и воссоздать объект.

Интерфейсы

Интерфейс Iterable объявлен в java.lang:

```
public interface Iterable<T> {
  Iterator<T> iterator();
}
```

Интерфейсы

Интерфейс Iterator объявлен в java.util.

Методы интерфейса Iterator<E>:

boolean hasNext() – проверяет наличие следующего элемента, а в случае его отсутствия (завершения коллекции) возвращает false. Итератор при этом остается неизменным.

Интерфейсы

E next() – возвращает объект, на который указывает итератор, и передвигает текущий указатель на следующий, предоставляя доступ к следующему элементу. Если следующий элемент коллекции отсутствует, то метод next() может возбудить исключение NoSuchElementException;

void remove() – удаляет объект, возвращенный последним вызовом метода next(), метод необязательный, если не реализован – возбуждает исключение UnsupportedOperationException.

Цикл forEach

```
for( T element : Iterable<T> ) {
   /*
    здесь можно использовать element
          обычно в правой части выражения
   */
}
```

Интерфейсы

Интерфейс Comparable<E> объявлен в java.lang:

```
interface Comparable<E> {
    int compareTo(E other);
}
```

Этот интерфейс предоставляет т.н. «естественный порядок сортировки».

Интерфейсы

int compareTo(E other)

- сравнивает this объект с other и возвращает:

-1 если this < other,

0 – если они равны и

1 если this > other.

Интерфейсы

Интерфейс Comparator<E> объявлен в java.util:

```
interface Comparator<E> {
   int compare(E obj, E obj1);
}
```

Содержит объявление единственного метода.

Интерфейсы

int compare(E obj, E obj1)

- сравнивает obj объект с obj1 и возвращает:

-1 если obj < obj1,

0 – если они равны и

1 если obj > obj1.

Примеры

Самостоятельно изучите примеры из папок:

example_1 и example_2.

Пример в папке example_1 показывает реализацию интерфейсов Comparable<>, Iterable<>, Iterator<> и использование их для сортировки и перебора полей объектов.

Пример в папке example_2 полностью соответствует условию ИЗ 5, полное условие варианта смотрите в CONTACTS.TXT. Результаты работы программы в CONTACTS.OUT.TXT.