Лабораторная №1: класс Object — метод toString() и контракт equals-hashCode, generics и Iterable, правило PECS, клонирование

1. Создание классов для использования вместе с коллекциями

Реализуйте класс для хранения телефонного номера. Телефонный номер состоит из двух частей — кода региона и местного номера. Обе части должны состоять только из цифр.

Экземпляры класса должны корректно представляться в строковом виде — код региона должен быть заключён в скобки, а цифры местного номера разделены дефисом на группы по два знака, первая группа состоит из двух или трёх знаков. Например: (981)023-45-67, (7890)65-43-21.

Экземпляры класса должны корректно использоваться вместе с множествами и ассоциативными массивами — множества не могут содержать два одинаковых номера, а ассоциативные массивы могут иметь только одно значение для одного и того же номера. Одинаковыми считаются номера, у которых совпадают код региона и местный номер. Проиллюстрируйте использование класса вместе с множествами и ассоциативными массивами.

2. Generic односвязного списка, его использование в «for-each»

Доработайте класс односвязного списка из соответствующей задачи 110-го курса, чтобы он позволял:

- сохранять и извлекать значения заданного типа;
- перебирать при помощи оператора «for-each» всё содержимое списка;
- выполнять заданное действие для каждого значения списка;
- агрегировать значения (см. ниже).

Агрегирование значений

Агрегирование значений — это операция, преобразующая (сводящая) набор значений к какому-то итоговому значению. В качестве примера можно рассмотреть суммирование чисел коллекции, поиск минимального/максимального числа, объединение строк коллекции и т. п.

Агрегирование выполняется следующим образом:

1. Задаётся исходное значение агрегата.
2. Для каждого значения из списка по очереди вызывается заданная методу агрегирования функция, которой передаётся это значение и текущее значение агрегата (для первого значения списка исходное значение агрегата), на основании чего функция должна вычислить следующее значение агрегата.
3. Результат последнего вызова функции и есть агрегированное значение. Если список пуст, то исходное значение агрегата будет итоговым.

Например, при суммировании чисел списка исходное значение равно нулю, а на каждом шаге агрегирования метод будет складывать накопленную сумму с очередным числом из списка.

При конкатенации (сложении) строк списка исходным значением будет пустая строка, а на каждом шаге агрегирования метод будет добавлять к итоговой строке очередную строку списка.

Тип результата не обязательно должен совпадать с типом значений коллекции. Так при конкатенации строк исходным значением может служить не пустая строка, а экземпляр класса `StringBuilder`. На каждом шаге очередная строка будет добавляться к `StringBuilder` ‘у, результатом агрегирования также будет `StringBuilder`.

3. Извлечение данных из параметризованной коллекции

Реализуйте в классе `Person` из соответствующей задачи 110-го курса статический метод, печатающий данные обо всех персонах, содержащихся в заданном наборе, реализующим `Iterable<Person>`. Что произойдёт, если передать в метод коллекцию параметризованную классом, дочерним к `Person`? Исправьте метод, чтобы он мог использоваться с коллекциями, параметризованными дочерними типами. Проиллюстрируйте использование метода.

4. Заполнение параметризованной коллекции

Реализуйте в классе `RegularStudent` из соответствующей задачи 110-го курса статический метод, заполняющий заданную коллекцию типа `Collection<RegularStudent>` данными о некоторых студентах (например, данными, приведёнными в примере в задаче 110-го курса).

Сделайте так, чтобы метод мог корректно использоваться как с коллекциями типа `Collection<RegularStudent>`, так и с коллекциями, параметризованными другими типами. Подумайте, какими типами может быть параметризована такая коллекция. Проиллюстрируйте использование метода.

*Доработайте класс односвязного списка, чтобы методы «выполнять заданное действие» и «агрегировать значения» соответствовали правилам PECS при использовании вспомогательных типов.

5. Generic двусвязного списка, его использование в «for-each»

Доработайте класс двусвязного списка из соответствующей задачи 110-го курса, чтобы он позволял:

- сохранять и извлекать значения заданного типа;
- перебирать при помощи оператора «for-each»:
  - всё содержимое списка;
  - всё содержимое списка в обратном порядке от последнего элемента к первому;
- выполнять заданное действие для каждого значения списка в прямом/обратном порядке (два метода);
- агрегировать значения (см. выше в задаче 2) в прямом/обратном порядке (два метода);
- *клонировать список (последующие операции с исходным списком или его копией не должны влиять друг на друга).

*При реализации заданий «выполнять заданное действие» и «агрегировать значения» учитывайте правила PECS при использовании вспомогательных типов.

6. *Использование коллекции логических значений в конструкции «for-each», клонирование коллекции

Доработайте классы коллекций логических значений из соответствующей практики 110-го курса, чтобы они позволяли:

- перебирать при помощи оператора «for-each» все установленные значения, то есть те индексы, элементы коллекции в которых равны true;
- создавать полную копию коллекции (клонировать коллекцию).