Лабораторная работа 7

Написать программу, решающую задачу, и оформить отчет.

1. Необходимо реализовать обобщенный класс АТД «Список» на двух массивах, первый - массив элементов списка, второй – целочисленный массив позиций (необходимые операции (открытые методы) перечислены ниже).
2. Реализовать класс позиции в списке (см. ниже).
3. Реализовать класс исключений для методов списка.
4. Реализовать отдельный класс элемента списка (объект), который реализует интерфейс Comparable<T> (по вариантам).
5. Реализовать класс с обобщенным статическим методом (методами) для типов реализующий интерфейс Comparable<T> (по вариантам). В main() (в этом же классе) проверить метод (методы) для классов собственного типа (см. 2), String и Integer.

Абстрактный тип данных (АТД) «Список»

Список – последовательность элементов определенного типа (все одного):
$a_1, a_2, \ldots, a_n$,
где n >= 0, n – длина списка.
Если n >= 1, то a_1 – первый элемент списка, a_n - последний элемент списка.
n = 0 – список пустой.
Важное свойство списка – его элементы можно линейно упорядочить, в соответствии с их **позицией** в списке. Элемент a_i предшествует элементу a_{i+1} для i = 1, 2, …, n-1, a_i следует за a_{i-1} для i = 2, 3, …, n. Элемент a_i имеет позицию i. За последним элементом списка располагается позиция после последнего.

L – список, x – объект (тип совпадает с типом значений списка), p – позиция в списке.

Операции (указаны все и явные, и неявные параметры):

End(L) – возвращает позицию после последнего.

Insert(x, p, L)
$a_1, a_2, \ldots, a_n$
вставим x в позицию p
$a_1, a_2, \ldots, a_{p-1}, x, a_p, \ldots, a_n$
вставим x в позицию End(L)
$a_1, a_2, \ldots, a_n, x$
Если позиции p в списке L нет, то результат неопределен (ничего не делать).

Locate(x, L) – возвращает позицию в списке L объекта x. Если объекта в списке нет, то возвращается позиция End(L). Если несколько значений, совпадает со значением x, то возвращается первая позиция от начала.

Retrieve(p, L) – возвращается объект списка L в позиции p. Результат неопределен, если p=End(L) или в L нет позиции p (выбросить исключение).

Delete(p, L) – удалить элемент списка L в позиции p.
$a_1, a_2, \ldots, a_n$
после удаления
$a_1, a_2, \ldots, a_{p-1}, a_{p+1}, \ldots, a_n$
Результат неопределен, если в списке L нет позиции p или p=End(L) (ничего не делать).

Next(p, L) – возвращает следующую за p позицию в списке L. Если p – последняя позиция в списке L, то Next(p, L) = End(L). Результат неопределен, если p нет в списке или p=End(L) (выбросить исключение)

Previous(p, L) – возвращает предыдущую перед p позицию в списке L. Результат неопределен, если p = 1, p = End(L) или позиции p нет в списке L (выбросить исключение).

Makenull(L) – делает список пустым.

First(L) – возвращает 1-ую позицию в списке L. Если список пустой, то возвращается End(L).

Printlist(L) – вывод списка на печать в порядке расположения элементов в списке.

Класс позиции

Номер элемента массива. Для удобства использования объекта доступ к переменной стоит оставить по умолчанию.

Класс элемента списка

Класс должен реализовывать интерфейс Comparable <T>. Для этого надо реализовать единственный метод:

public int compareTo(T obj);

Использование:

a.compareTo(b);

Возвращаемый результат - целое число:

меньше 0, если a<b, 0, если a==b и больше 0 a>b.

Необходимо переопределить метод toString() для отображения объекта (элемента списка) на экран.

Лабораторная работа состоит из двух этапов: проектирование и реализация. Пока не принят этап проектирования, реализация не рассматривается!

Проект представляется в виде текстового файла, в котором описываются все классы и их данные. Кроме того, на естественном языке перечисляются все действия для всех конструкторов, операций и вспомогательных методов.

Общие требования:

1. Не использовать коллекции и класс Object.
2. Качество кода. Грамотное проектирование. Функциональная прочность вспомогательных методов. Оптимизация по времени и по памяти.
3. Не использовать доступ по умолчанию (кроме класса позиции). Указывать доступ для всего, классов, данных, методов.
4. Не использовать рекурсию.
5. Для всех классов перед каждым методом (кроме main()),в комментариях должно быть записано: какую задачу решает метод, какие параметры ему передаются, что возвращается в результате.
6. Для всех объявленных в методах переменных, включая метод main(), в комментариях необходимо указать их назначение.
7. Все важные для понимания методов моменты должны сопровождаться комментариями.

Отчет по лабораторной работе

Содержание отчета:

- Условие задачи.
- Описание алгоритма (проект).
- Листинг программы (проект + код).
- Набор тестов программы (в main()).

Отчет по лабораторной работе представляется в виде набора текстовых файлов (*.java). Защита отчета проходит в форме доклада студента по выполненной работе и ответов на вопросы преподавателя.

В случае если оформление отчета и поведение студента во время защиты соответствуют указанным требованиям, студент получает максимальное количество баллов.

Основаниями для снижения количества баллов в диапазоне от max до min являются:

- реализован неэффективный алгоритм по памяти и/или количеству операций;
- не проведена оптимизация повторов, ветвлений;
- полное или частичное отсутствие комментариев.

Отчет не может быть принят и подлежит доработке в случае:

- неправильной работы программы для всех или некоторых входных данных;
- отсутствия необходимых разделов;
- неполного выполнения задания по лабораторной работе.