**Лабораторная работа №1 — Комплексные числа**

**Задача:** написать класс complex, позволяющий пользователю

работать с комплексными числами.

**Указания по выполнению:**

- лабораторная работа состоит из трех файлов:

*complex.java* — реализация методов класса complex;

*test-complex.java* — тестирование и проверка возможностей

класса;

- класс complex должен содержать поля для хранения

вещественной и мнимой части комплексного числа;

- класс complex должен содержать следующие методы:

конструктор по-умолчанию,

конструктор с 2 параметрами,

вычисление модуля числа,

вычисление аргумента числа,

арифметические действия(+,-).

**Лабораторная работа №2 — Связный список**

**Задача:** реализовать на основе односвязного списка контейнер

myList, который бы позволял хранить произвольный тип данных.

**Указания по выполнению:**

- лабораторная работа состоит из двух файлов:

*myList.java* — описание класса list, прототипы методов,

реализация методов;

*test-list.java* — тестирование и проверка возможностей класса;

- класс mylist должен содержать следующие поля:

ссылку на первый элемент списка;

ссылку на последний элемент списка;

число элементов в списке;

- класс myList должен содержать следующие методы:

конструктор;

конструктор копирования;

добавление элемента в начало списка, в конец списка;

удаление элемента из начала списка;

удаление элемента по значению;

получение числа элементов в списке;

очистка списка (удаление всех элементов);

- при невозможности извлечения или удаления элемента должны

генерироваться исключения.

**Лабораторная работа №3 — Динамический массив**

**Задача:** реализовать контейнер myVector — динамический массив

для хранения произвольного типа данных.

**Указания по выполнению:**

- лабораторная работа состоит из двух файлов:

*myVector.java* — описание класса vector, прототипы методов,

реализация методов;

*test-vector.java* — тестирование и проверка возможностей

класса;

- класс myVector должен быть шаблонным;

- класс myVector должен содержать следующие поля:

ссылку на массив;

текущее число элементов;

максимальное число элементов;

- класс myVector должен содержать следующие методы:

конструктор;

конструктор копирования;

добавление элемента в конец;

удаление последнего элемента;

удаление элемента по индексу;

вставка элемента по индексу;

изменение размера массива;

очистка массива (удаление всех элементов);

получение текущего и максимального размера контейнера;

- пользователь должен иметь возможность задать начальное

число элементов в массиве с помощью конструктора;

- память под массив должна выделяться с запасом. Можно

придерживаться простейшей стратегии: макс.число элементов =

2\*(число элементов) + 10;

- если число элементов за счет добавления новых превышает

максимальное, необходимо заново выделить память, скопировать

существующие элементы

- при невозможности удаления элемента, доступа к элементу и

т. п. должны генерироваться исключения;

**Лабораторная работа №4 — Двоичное дерево поиска**

**Задача:** реализовать класс binary\_tree, который бы обеспечивал

логарифмическую сложность поиска и добавления нового

элемента.

**Указания по выполнению:**

- лабораторная работа состоит из двух файлов:

*binary\_tree.java* — описание класса binary\_tree, прототипы

методов, реализация методов;

*test-binary\_tree.java* — тестирование и проверка возможностей

класса;

- создать вспомогательную шаблонную структуру, содержащую

поле для хранения значения произвольного типа и 2 поля для

хранения ссылок на левого и правого потомков;

- класс binary\_tree должен быть шаблонным;

- класс binary\_tree должен содержать следующие поля:

ссылку на корневой элемент;

число элементов в дереве;

- класс binary\_tree должен содержать следующие методы:

конструктор;

конструктор копирования;

добавление элемента;

поиск элемента;

удаление всех элементов;

необходимые интерфейсы (в зависимости от реализации);

**Лабораторная работа №5 — Простейшие сортировки**

**Задача:** реализовать три алгоритма сортировок: пузырьковую,

вставками, выбором.

**Указания по выполнению:**

- лабораторная работа состоит из двух файлов:

*simple-sort.java* — прототипы функций, реализация функций;

*test-simple-sort.java* — тестирование функций;

- список требований к функциям (достаточно реализовать три):

1) сортировка массива целых чисел заданного размера

2) сортировка любой подпоследовательности массива

3) сортировка массивов с любыми типами данных

4) сортировка контейнера по выбору

**Лабораторная работа №6 — Эффективные сортировки**

**Задача:** реализовать три алгоритма сортировок: быструю,

пирамидальную, слиянием.

**Указания по выполнению:**

- лабораторная работа состоит из двух файлов:

*effective-sort.java* — прототипы функций, реализация функций;

*test-effective-sort.java* — тестирование функций;

- список требований к функциям (достаточно реализовать три):

1) сортировка массива целых чисел заданного размера

2) сортировка любой подпоследовательности массива

3) сортировка массивов с любыми типами данных

4) сортировка контейнера по выбору

**Лабораторная работа №7 — Поразрядные сортировки**

**Задача:** реализовать три алгоритма сортировок: MSD, LSD,

подсчетом.

**Указания по выполнению:**

- лабораторная работа состоит из двух файлов:

*lin-sort.java* — прототипы функций, реализация функций;

*test-lin-sort.java* — тестирование функций;

- список требований к функциям (достаточно реализовать два):

1) сортировка массива целых чисел заданного размера

2) сортировка любой подпоследовательности массива

3) поддержка целых чисел разной длины

**Лабораторная работа №8 — Итератор контейнера**

**Задача:** реализовать вложенный класс myIterator для контейнера myList из ЛР№2.

**Указания по выполнению:**

- лабораторная работа состоит из двух файлов:

*myList.java* — описание класса myList, описание класса myIterator,

прототипы методов, реализация методов;

*test-list.java* — тестирование и проверка возможностей класса;

- класс myIterator должен содержать следующие поля:

ссылку на элемент списка;

- класс myIterator должен содержать следующие методы:

конструктор по-умолчанию

- класс myList следует дополнить методами, возвращающими

итератор на начало/конец списка;

**Дополнительные задания:**

- протестировать сортировки из ЛР№5 на получившемся

контейнере;