Практическая работа № 9. Использование полиморфизма при программировании при реализации алгоритмов сортировок и поиска

Цель работы: освоение на практике методов сортировки с использованием приемов программирования на объектно-ориентированном языке Java.

Теоретические сведения

Сортировка — это процесс упорядочивания списка элементов (организация в определенном порядке) исходного списка элементов, который возможно организован в виде контейнера или храниться в виде коллекции.

Процесс сортировки основан на упорядочивании конкретных значений, например:

- сортировка списка результатов экзаменов баллов в порядке возрастания результата;
- сортировка списка людей в алфавитном порядке по фамилии.

Есть много алгоритмов для сортировки списка элементов, которые различаются по эффективности.

Алгоритм сортировки вставками.

Работа метода сортировки состоит из следующих шагов:

- выбрать любой элемент из списка элементов и вставить его в надлежащееместо в отсортированный подсписок;
- повторять предыдущий шаг, до тех пор, пока все элементы не будут вставлены.

Более детально:

- рассматриваем первый элемент списка как отсортированный подсписок (то есть первый элемент списка);
- вставим второй элемент в отсортированный подсписок, сдвигая первый элемент по мере необходимости, чтобы освободить место для вставки нового элемента;
- вставим третий элемент в отсортированный подсписок (из двух элементов), сдвигая элементы по мере необходимости;
- повторяем до тех пор, пока все значения не будут вставлены на свои соответствующие позиции.

Алгоритм быстрой сортировки (Quick Sort).

Состоит из последовательного выполнения двух шагов:

- массив A[1..n] разбивается на два непустых подмассивов по

отношению к"опорному элементу”;

- два подмассива сортируются рекурсивно посредством Quick Sort.

Алгоритм сортировка слиянием (Merge Sort).

Состоит из последовательного выполнения трех шагов:

- разделить массив A[1..n] на 2 равные части;
- провести сортировку слиянием двух подмассивов (рекурсивно);
- объединить (соединить) два отсортированных подмассива.

Использование интерфейса Comparable в Джава программах для сортировки объектов

Техника программирования сортировок в Джава отличается от написания алгоритмов на процедурных языках программирования. При написании кода большим преимуществом является использование основного принципа ООП – полиморфизма. Напомним, что класс, который реализует интерфейс Comparable определяет метод compareTo(), чтобы определить относительный порядок своих объектов.

Таким образом мы можем использовать полиморфизм, чтобы разработать обобщенную сортировку для любого набора Comparable объектов.

При разработке класса, реализующего метод сортировки, нужно помнить, что метод принимает в качестве параметра массив объектов типа Comparable или фактически полиморфных ссылок.

Таким образом, один метод, может быть, использован для сортировки любых объектов, например: People (людей), Books (книг), или любой каких-либо других объектов.

Методу сортировки все-равно, что именно он будет сортировать, ему только необходимо иметь возможность вызвать метод compareTo().

Это обеспечивается использованием в качестве типа формального параметра интерфейса Comparable или интерфейсной ссылки.

Кроме того, таким образом каждый класс “для себя” решает, что означает для одного объекта, быть меньше, чем другой.

Листинг 9.1 – Пример решения задачи.

```
public class Sorting {
public static void selectionSort (Comparable[] list){
int min;
Comparable temp;
for (int index = 0; index < list.length-1; index++){
min = index;
for (int scan = index+1; scan < list.length; scan++)
```

```
 if (list[scan].compareTo(list[min]) < 0){
  min = scan;
  temp = list[min];
  list[min] = list[index];
  list[index] = temp;
    }
  }
}
```

Задания на практическую работу № 9

1. Написать тестовый класс, который создает массив класса Student и сортирует массив iDNumber и сортирует его вставками.

2. Напишите класс SortingStudentsByGPA который реализует интерфейс Comparator таким образом, чтобы сортировать список студентов по их итоговым баллам в порядке убывания с использованием алгоритма быстрой сортировки.

3. Напишите программу, которая объединяет два списка данных о студентах в один отсортированный списках с использованием алгоритма сортировки слиянием.

4. Напишите свою собственную реализацию интерфейса Comparable