Программа Разработка ПО МИФИ
Курс Алгоритмы и структуры данных
Группа М24-535
Баклыков Никита

ДОМАШНЕЕ ЗАДАНИЕ

Модуль 3. Сортировки и поиск

ЗАДАНИЕ

Реализовать все методы binarySearch, которые имеются для классов Collections и Arrays в Java, используя документацию, доступную по ссылкам: Arrays и Collections соответственно.

РЕШЕНИЕ

static int binarySearch(byte[] a, byte key)	выполнено
static int binarySearch(byte[] a, int fromIndex, int toIndex, byte key)	выполнено
static int binarySearch(char[] a, char key)	выполнено
static int binarySearch(char[] a, int fromIndex, int toIndex, char key)	выполнено
static int binarySearch(double[] a, double key)	выполнено
static int binarySearch(double[] a, int fromIndex, int toIndex, double key)	выполнено
static int binarySearch(float[] a, float key)	выполнено
static int binarySearch(float[] a, int fromIndex, int toIndex, float key)	выполнено
static int binarySearch(int[] a, int key)	выполнено
static int binarySearch(int[] a, int fromIndex, int toIndex, int key)	выполнено
static int binarySearch(long[] a, long key)	выполнено
static int binarySearch(long[] a, int fromIndex, int toIndex, long key)	выполнено
static int binarySearch(short[] a, short key)	выполнено
static int binarySearch(short[] a, int fromIndex, int toIndex, short key)	выполнено
static int binarySearch(T[] a, T key, Comparator c)	выполнено
static int binarySearch(T[] a, int fromIndex, int toIndex, T key, Comparator c)	выполнено

Решение данной задачи основано на применении подхода перегрузки в Java, которая реализует концепцию статического полиморфизма, также известную как раннее связывание (early binding). Это означает, что компилятор определяет, какой метод вызвать, основываясь на типе и количестве аргументов, переданных методу.
При реализации методов для класса Arrays и всех методов класса Collections применяется класс Comparator, интерфейс Comparable и механизм дженериков.
Объект Comparator в Java используется для задания нестандартного (или вторичного) порядка сортировки объектов. В отличие от интерфейса Comparable, который определяет естественный порядок объектов через метод compareTo, Comparator позволяет определить различные способы сравнения объектов без изменения их классов.
Дженерики — это механизм, который позволяет создавать классы, интерфейсы и методы, способные работать с различными типами данных, сохраняя строгую типизацию. Дженерики предоставляют средства для параметризации типов, что позволяет избежать ошибок времени выполнения, связанных с неправильным приведением типов, и улучшает читаемость и повторное использование кода.

Методы бинарного поиска реализованы в классах MyArrays и MyCollections.

Для решения задачи использовался список todo:

Класс Arrays содержит реализации следующих методов

2 КлассCollections

содержит реализации следующих методов

static int binarySearch(List> list, T key)	выполнено
static int binarySearch(List list, T key, Comparator c)	выполнено

Проверим работу разработанных методов на разных примерах.

```
// Проверка бинарного поиска для массива byte byte[]
        byteArray = {1, 3, 5, 7, 9, 11};
Поиск в byte array элемента 5: 2
Поиск в byte array элемента 8: -5

// Проверка бинарного поиска для части массива byte
Поиск в byte array (fromIndex=1, toIndex=4) элемента 3: 1
Поиск в byte array (fromIndex=1, toIndex=4) элемента 8: -5

// Проверка бинарного поиска для массива char
        char[] charArray = {'a', 'c', 'e', 'g', 'i', 'k'};
Поиск в char array элемента 'e': 2
Поиск в char array элемента 'h': -5
// Проверка бинарного поиска для части массива char
Поиск в char array (fromIndex=1, toIndex=4) элемента 'g': 3
Поиск в char array (fromIndex=1, toIndex=4) элемента 'h': -5

// Проверка бинарного поиска для массива int int[]
        intArray = {10, 20, 30, 40, 50};
Поиск в int array элемента 20: 1
Поиск в int array элемента 60: -6
// Проверка бинарного поиска для части массива int
Поиск в int array (fromIndex=1, toIndex=4) элемента 30: 2
Поиск в int array (fromIndex=1, toIndex=4) элемента 60: -5

// Проверка бинарного поиска для массива long long[]
        longArray = {100L, 200L, 300L, 400L, 500L};
Поиск в long array элемента 200: 1
Поиск в long array элемента 600: -6
// Проверка бинарного поиска для части массива long
Поиск в long array (fromIndex=1, toIndex=4) элемента 300: 2
Поиск в long array (fromIndex=1, toIndex=4) элемента 600: -5

// Проверка бинарного поиска для массива short short[]
        shortArray = {1, 2, 3, 4, 5};
Поиск в short array элемента 4: 3
Поиск в short array элемента 6: -6
// Проверка бинарного поиска для части массива short
Поиск в short array (fromIndex=1, toIndex=4) элемента 4: 3
Поиск в short array (fromIndex=1, toIndex=4) элемента 6: -5

// Проверка бинарного поиска для массива double double[] doubleArray = {1.1, 2.2, 3.3, 4.4,
        5.5};Поиск в double array элемента 4.4: 3
Поиск в double array элемента 6.6: -6
// Проверка бинарного поиска для части массива double
Поиск в double array (fromIndex=1, toIndex=4) элемента 3.3: 2
Поиск в double array (fromIndex=1, toIndex=4) элемента 6.6: -5
```

```
// Проверка бинарного поиска для массива float
        float[] floatArray = {0.5f, 1.5f, 2.5f, 3.5f, 4.5f};
Поиск в float array элемента 3.5: 3
Поиск в float array элемента 6.5: -6
// Проверка бинарного поиска для части массива float
Поиск в float array (fromIndex=1, toIndex=4) элемента 2.5: 2
Поиск в float array (fromIndex=1, toIndex=4) элемента 5.5: -5

// Проверка бинарного поиска для массива объектов с Comparator
        String[] stringArray = {"apple", "banana", "cherry", "date", "fig"};
Поиск в string array элемента 'cherry': 2
Поиск в string array элемента 'grape': -6
// Проверка бинарного поиска для части обобщенного массива с использованием Comparator
Поиск в string array (fromIndex=1, toIndex=4) элемента 'date': 3 Поиск в
string array (fromIndex=1, toIndex=4) элемента 'grape': -5

Тестирование binarySearch с естественным порядком:
// Создаем тестовый список для проверки методов бинарного поиска list = {0,
        10, 20, 30, 40, 50, 60, 70, 80, 90}
Индекс элемента 30: 3
Индекс элемента 35 (не найден): -5

Тестирование binarySearch с пользовательским компаратором:
Индекс элемента 30 в обратном порядке: 6
Индекс элемента 35 в обратном порядке (не найден): -7
```