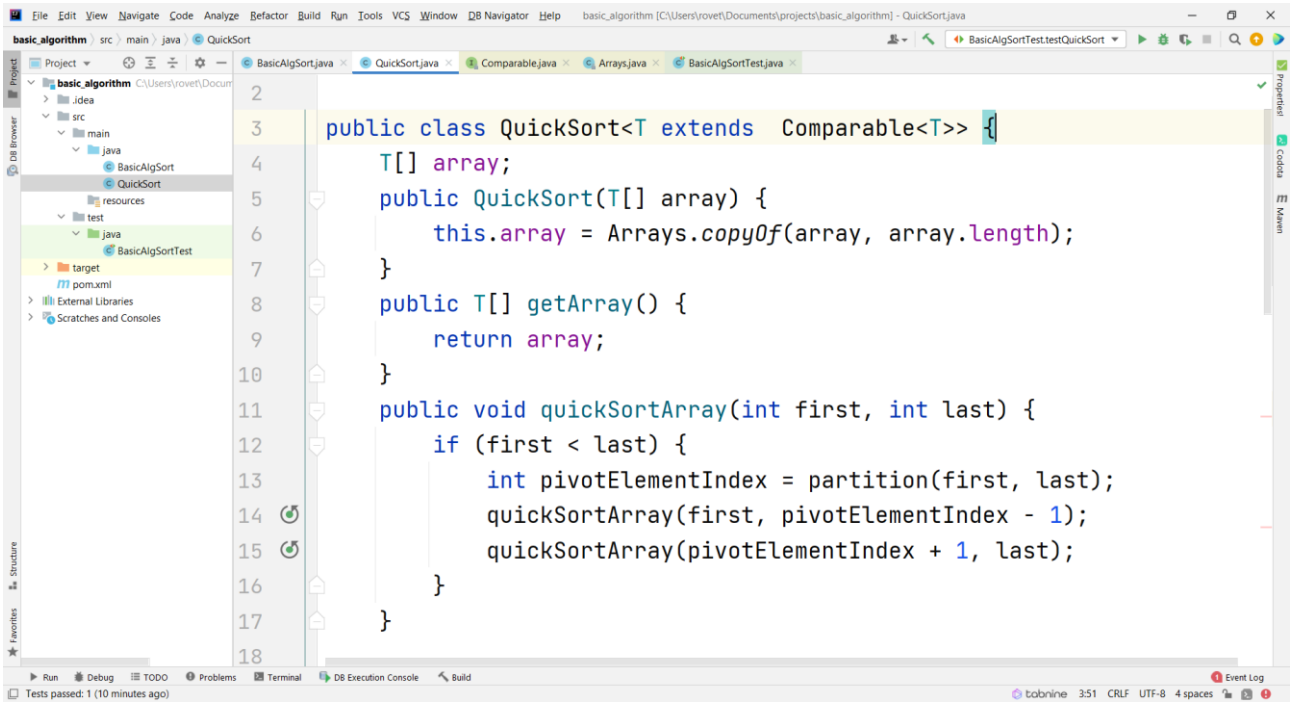
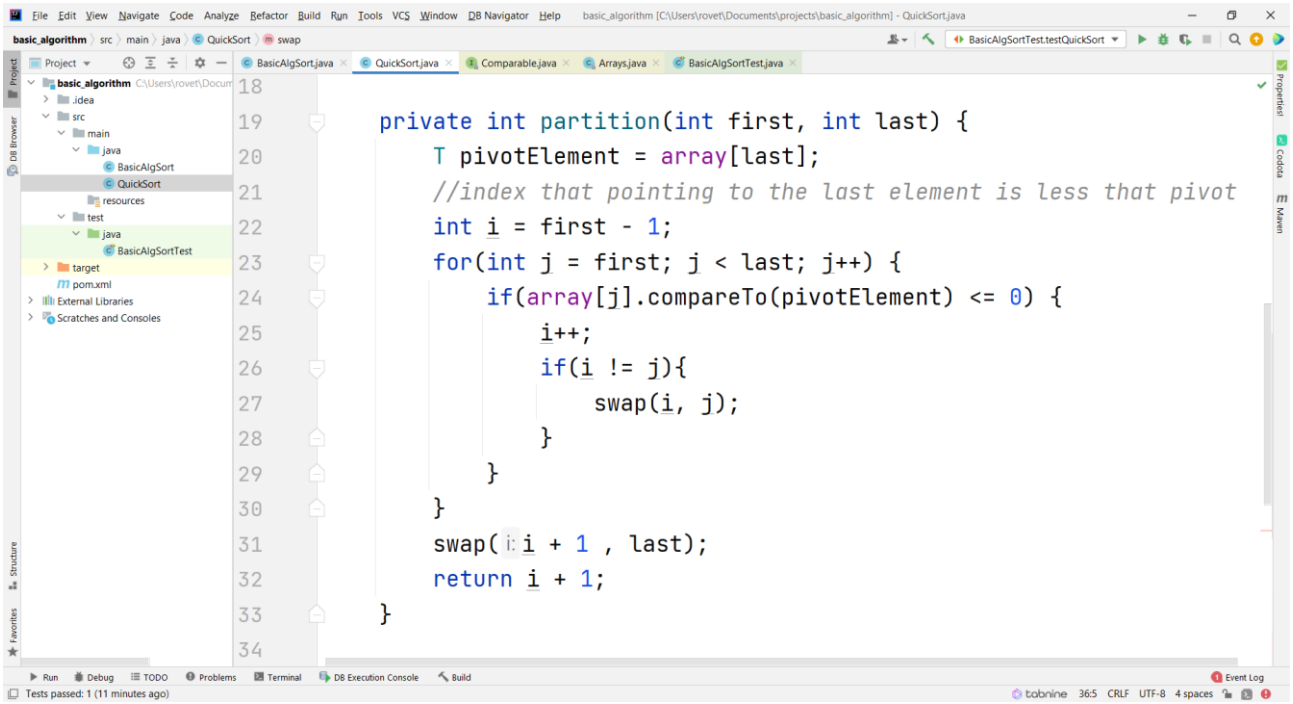
Робота №2. Алгоритм швидкого сортування

В лабораторній роботі №1 було створено Java-бібліотеку для сортування будь-яких стандартних типів даних на основі базових алгоритмів сортування (обміном, вибором та включенням).

В даній лабораторній роботі подано рекурсивну реалізацію алгоритму швидкого сортування з опорним останнім елементом масиву для сортування будь-яких стандартних типів даних за зростанням (рис.1). Реалізацію додано до Java-бібліотеки, створеної у лабораторній роботі №1.



```
2
3   public class QuickSort<T extends  Comparable<T>> {
4       T[] array;
5       public QuickSort(T[] array) {
6           this.array = Arrays.copyOf(array, array.length);
7       }
8       public T[] getArray() {
9           return array;
10      }
11      public void quickSortArray(int first, int last) {
12          if (first < last) {
13              int pivotElementIndex = partition(first, last);
14              quickSortArray(first, pivotElementIndex - 1);
15              quickSortArray(pivotElementIndex + 1, last);
16          }
17      }
18
```



```
18
19      private int partition(int first, int last) {
20          T pivotElement = array[last];
21          //index that pointing to the last element is less that pivot
22          int i = first - 1;
23          for(int j = first; j < last; j++) {
24              if(array[j].compareTo(pivotElement) <= 0) {
25                  i++;
26                  if(i != j){
27                      swap(i, j);
28                  }
29              }
30          }
31          swap( i: i + 1 , last);
32          return i + 1;
33      }
34
```

Рис.1. Рекурсивна реалізація алгоритму швидкого сортування

На рис.2. подано тест для тестування реалізації алгоритму швидкого сортування

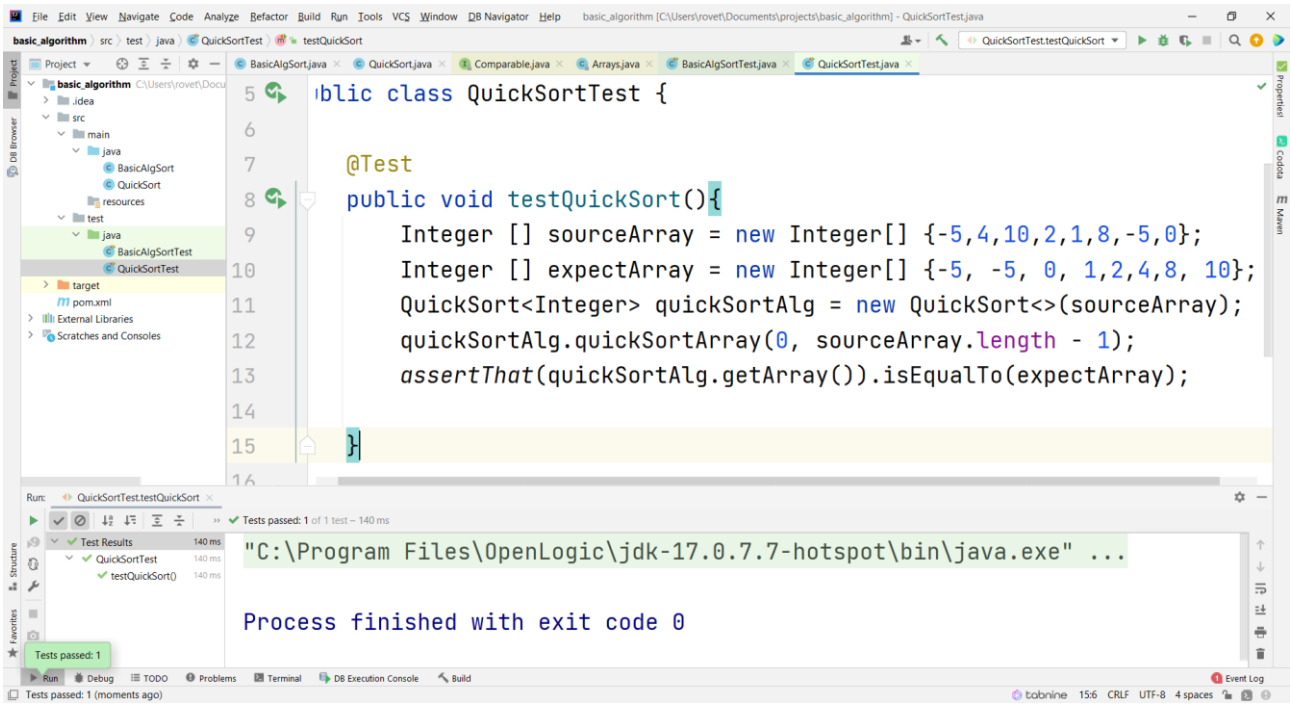


Рис.2. Тестування алгоритму швидкого сортування

Завдання 1. Адаптувати створену реалізацію алгоритму швидкого сортування для випадку сортування масиву даних за спаданням

Завдання 2. Адаптувати створену реалізацію алгоритму швидкого сортування для випадку задання центрального елементу масиву в якості опорного елемента.

Завдання 3. Адаптувати створену реалізацію алгоритму швидкого сортування для випадку задання випадкового обраного елементу масиву в якості опорного елемента.

Завдання 4. Порівняти час виконання алгоритму з вибором опорного елемента останнім, центральним та випадково обраним елементами масиву.