**package** searchsort;

**import** java.util.Arrays;

**import** java.util.Scanner;

**public** **class** Binary {

**public** **static** **void** selectionSort(**int**[] arr) {

/\*\*

\* method selectionSort performs selection sort on given array for binary searching

\* input :

\* int arr[] : list of integers that are to be sorted

\* output :

\* there is no output in this method, it is an in-place function to perform a selection sort

\*\*/

**for** (**int** i=0;i<arr.length;i++) {

**int** index = i;

**for** (**int** j=(i+1);j<arr.length;j++) {

**if** (arr[j]<arr[index]) {

index = j;//index is the position of the smallest value in arr

}

}

**int** min=arr[index];

arr[index]=arr[i];

arr[i]=min;

}

}

**public** **static** **int**[] binarySearch(**int** arr[], **int** startIndex, **int** endIndex, **int** searchItem){

/\*\*

\* method binarySearch performs binary search recursively on a given array arr.

\* input :

\* int arr[] : list of integers to perform binary search on

\* int startIndex : the first index which would generally be 0 when input

\* int endIndex : the last index of the array, which would be arr.length-1 when input

\* int searchItem : the item being searched

\* output :

\* int result[] : array containing the index and the value being searched when the variable is found,

\* else contains [-1,-1].

\*\*/

**int** midIndex = (startIndex + endIndex)/2;

**int** midValue = arr[midIndex];

**if** (midValue == searchItem){

**int**[] result = {midIndex, midValue};

//Result is array because I wanted to contain 2 informations: the value and the index, thus I created array which can hold many numbers in once

**return** result;

} **else** **if** (midValue < searchItem) {

**return** *binarySearch*(arr, midIndex + 1, endIndex, searchItem);

//midIndex has been incremented by 1 as we know that midvalue is not the searched, thus we are setting the midindex to value that is one bigger

} **else** **if** (midValue > searchItem) {

**return** *binarySearch*(arr, startIndex, midIndex - 1, searchItem);

//in contrast to first example, the midvalue is bigger than the item being searched so the midindex is being decremented by 1.

} **else** {

**int**[] result = {-1, -1};

**return** result;

}

}

**public** **static** **void** main(String args[]) {

Scanner keyboard = (**new** Scanner(System.***in***));

System.***out***.println("Input the length of the array");

String slen = keyboard.nextLine();

**int** len = Integer.*parseInt*(slen);

**int**[] myarray= **new** **int**[len]; //declaring the array with length input by the user

System.***out***.println("Input elements of the array");

**for** (**int** d=0;d<len;d++) { //for loop assigning arrays in order to the user's input

String sel = keyboard.nextLine();

**int** el = Integer.*parseInt*(sel);

myarray[d]=el;

}

System.***out***.print("Original array: ");

**for** (**int** e=0;e<myarray.length;e++) {

System.***out***.print(myarray[e]+" ");

}

*selectionSort*(myarray); //adopts the selectionSort method to perform a sort in preparation to the binary search

System.***out***.println();

System.***out***.print("Sorted array: ");

**for** (**int** e=0;e<myarray.length;e++) {

System.***out***.print(myarray[e]+" ");

}

System.***out***.println();

System.***out***.println("Please input value being searched");

String ssearch = keyboard.nextLine();

**int** search = Integer.*parseInt*(ssearch);

**int**[] result = *binarySearch*(myarray,0,myarray.length-1,search);

**if** (result[0]==-1) {

System.***out***.println("Item not found");

}

**else** {

System.***out***.println("value "+result[1]+" has been searched at index "+result[0]);

}

}

}

