|  |  |  |
| --- | --- | --- |
|  | Project 3  Iterative Linear Search, Recursive Binary Search, and Recursive Selection Sort | C:\Users\irena17\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\F228424A.tmp |

Class River describes river’s name and its length in miles. It provides accessor methods (getters) for both variables, toString() method that returns String representation of the river, and method isLong() that returns true if river is above 30 miles long and returns false otherwise.

Class CTRivers describes collection of CT rivers. It has no data, and it provides the following service methods. None of the methods prints anything, except method printLongRiversRec, which prints all long rivers. Input parameter n in all methods is number of occupied elements in the list.

// Prints all long rivers in the list. Print them in same order as they were in the list . List can be

// empy or not.

* **void printLongRiversRec(River[] list, int n)**

// Returns index for the river object with given name. Returns -1 for unsuccessful search. List can

// be empy or not.

* **int linearSearch(River[] list, int n, String name)**

// Returns ArrayList of rivers with length between min and max inclusive. If no such river was found,

// method returns an empty Arraylist<River>. List can be empy or not.

* **ArrayList <River> searchRange(River[] list, int n, int min, int max)**

// Sorts list of rivers by comparing them by names. Apply selection sort recursively. List of rivers can be

// empy or nonempty. Empty list and list with one river only are sorted. Lists with two or more rivers are

// sorted by swapping last river in the list with river object that has name that is last in lexicographic order // in the array, and after that recursively sorting sublist of first n-1 rivers.

* **void sortByNameRec(River[] list, int n)**

// PRECONDITION: Method assumes that input list is sorted by names. First and last are indices of the first

// and last river of the current sublist. Method returns index of river object with given name or returns -1

// if none of the rivers has that name. List of rivers can be empy or not.

* **int binarySearchRec(River[] list, int first, int last, String name)**

The three methods highlighted in yellow must be implemented recursively.

**File “input.txt”**

Naugatuck 40

Pawcatuck 34

Quinebaug 69

Shepaug 26

Connecticut 407

Still 25

Quinnipiac 46

Housatoic 139

Class Driver has main method in it. Read data from an input file "input.txt" into an array of River objects, named **riverList**, and keep track of number of rivers stored in variable **counter**. Array **riverList** has capacity 100. Input file should be as shown.

* Print all long rivers.
* Apply one successful and one unsuccessful linear search.
* Print all rivers with length between min and max (min and max are provided by user). Must use for each loop in the main method to print resulting ArrayList<River> returned by method searchRange .
* Sort myList by river names, and print sorted list.
* Apply one successful and one unsuccessful binary search on sorted list.

Must print appropriate explanation in English of all steps performed in outcome.

**SUBMIT** a single word or PDF document named **p3\_yourLastName\_CS152** with the following in it:

* Your name, class section, project number and date of submission in the upper left corner.
* Copy of the code for each class and input file in separate rectangle.
* Picture of program run from BlueJ.
* Picture of UML diagram.