Nicolas P. Java code

package Sorting;

import java.util.Random;

import java.util.\*;

import javax.swing.JOptionPane;

public class AssessmentAll {

static boolean flag=false;

static String temp;

static int inttemp;

static String SearchValue;

static int SearchValueint;

static boolean found=false;

static String printer="";

static boolean Found=false;

static int index;

static int tempor;

static String ErrorMessage="";

static int foundfound=0;

public static void main(String[] args) {

String a[]= new String [1];

int b[]=new int[1];

int z=0;

String Decision1=JOptionPane.showInputDialog(null, "You can perform 4 different algorithms. Do you want to use a bubblesort(b)(it also shows you the time complexity if you sort numbers), a selection sort(s), a linear search(l) or a binary search (n)?");

Decision1=Decision1.toLowerCase();

if(Decision1.contains("b")){

bubblesort(a);

}

if(Decision1.contains("s")){

selectionsort(b);

}

if(Decision1.contains("l")) {

linearSearch(a);

}

if(Decision1.contains("n")) {

Binary(a);

}

}

public static void bubblesort (String[] a) {

String WordsorNum=JOptionPane.showInputDialog(null, "Do you want to enter numbers(n) or words(w) ?");

WordsorNum=WordsorNum.toLowerCase();

; if(WordsorNum.equals("w")){

String Decision=JOptionPane.showInputDialog(null, "Do you want to enter your own array(o) or use a sample array(s)");

Decision=Decision.toLowerCase();

//The user can choose here between entering his own words which will create an array and between using a pre made array

// with the values "q","w","e","r","t","y","u","i","o","p","a","f"

if(Decision.equals("o")) {

int len=Integer.parseInt(JOptionPane.showInputDialog(null, "How many words do you want to sort "));

//Length of array has to be defined before initializing array so program asks user for this information

String array[]= new String [len];

//different methods are now executed all with the argument array which is therefore passed on from one method to the next

assignmentString(array);

bubblesortsort(array);

printstatementString(array);

}

if(Decision.equals("s")){

String array[]= {"q","w","e","r","t","y","u","i","o","p","a","f"};

//Here is the pre-made array. It doesn't require any user input so program just directly sorts this.

bubblesortsort(array);

printstatementString(array);

}

}

if(WordsorNum.equals("n")) {

String Decision=JOptionPane.showInputDialog(null, "Do you want to enter your own array(o) or use a sample array. This sample array will first sort an array "+"\n containing 1000 randomly generated integers, then 10000 and then 50000 compairing their time complexity.(s)");

Decision=Decision.toLowerCase();

//The user can choose here between entering his own words which will create an array and between using a pre made array

// with the values "q","w","e","r","t","y","u","i","o","p","a","f"

if(Decision.equals("o")) {

int len=Integer.parseInt(JOptionPane.showInputDialog(null, "How many numbers do you want to sort "));

//Length of array has to be defined before initializing array so program asks user for this information

int array[]= new int [len];

//different methods are now executed all with the argument array which is therefore passed on from one method to the next

assignmentint(array);

bubblesortsortint(array);

printstatementint(array);

}

if(Decision.equals("s")){

Random random = new Random();

int[] array = random.ints(100, 1,100001).toArray();

//Here is the pre-made array. It doesn't require any user input so program just directly sorts this.

long timefor100;

long timefor1000;

long timefor10000;

long start = System.currentTimeMillis();

bubblesortsortint(array);

long end = System.currentTimeMillis();;

timefor100=(end - start);

printstatementinthorizontal(array);

System.out.println(" ");

System.out.println("For an array of 100 it took : "+timefor100+" ms.");

end=0;

start=0;

array = random.ints(1000, 1,100001).toArray();

//Here is the pre-made array. It doesn't require any user input so program just directly sorts this.

start = System.currentTimeMillis();

bubblesortsortint(array);

end = System.currentTimeMillis();;

timefor1000=(end - start);

printstatementinthorizontal(array);

System.out.println(" ");

System.out.println("For an array of 1000 it took : "+timefor1000+" ms.");

end=0;

start=0;

array = random.ints(10000, 1,100001).toArray();

//Here is the pre-made array. It doesn't require any user input so program just directly sorts this.

start = System.currentTimeMillis();

bubblesortsortint(array);

end = System.currentTimeMillis();;

timefor10000=(end - start);

printstatementinthorizontal(array);

System.out.println(" ");

System.out.println("For an array of 10000 it took : "+timefor10000+" ms.");

}

}

}

public static void selectionsort(int[] a) {

String Decision=JOptionPane.showInputDialog(null, "Do you want to enter your own array(o) or use a sample array(s) or 1000 random numbers between 1 and 100000(r)");

Decision=Decision.toLowerCase();

if(Decision.equals("o")) {

//own array chosen

int len=Integer.parseInt(JOptionPane.showInputDialog(null, "How many numbers do you want to sort "));

int array[]= new int [len];

assignmentint(array);

//own array created

selectionsortsort(array);

//selection sort run

printstatementint(array);

//array print method run

}

if(Decision.equals("s")){

int array[]= {9999,997,65,43,1,23,12,35,64,33,37,88,999,22,212};

//sample array 9999,997,65,43,1,23,12,35,64,33,37,88,999,22,212

selectionsortsort(array);

//selection sort run

printstatementint(array);

//array printed

}

if(Decision.equals("r")) {

Random random = new Random();

int[] array = random.ints(1000, 1,100000).toArray();

//each index of array of length 1000 random number between 1 and 100000 assigned to show time complexity

selectionsortsort(array);

//selection sort run

printstatementint(array);

//array printed

}

}

public static void bubblesortsort(String[] array) {

while(flag==false) {

//flag acts as signal whether or not two values in the array swapped places. If they didn't the array is sorted

flag=true;

//flag is set to true so program breaks if not set back to false inside of the if statement.

for(int a=1; a<array.length; a++) {

if(array[a].toLowerCase().compareTo(array[a-1].toLowerCase())<0) {

//if the first value is alphabetically behind the second value this line will return a negative number.

//This can be used because whenever this occurs the two values have to be swapped

temp=array[a];

array[a]=array[a-1];

array[a-1]=temp;

//temporary storage needed because both values have to be overwritten so that one of the values doesn't go lost

flag=false;

//flag set to false to signal that two values were sorted

}

}

}

flag=false;

}

public static void bubblesortsortint(int[] array) {

while(flag==false) {

//flag acts as signal whether or not two values in the array swapped places. If they didn't the array is sorted

flag=true;

//flag is set to true so program breaks if not set back to false inside of the if statement.

for(int a=1; a<array.length; a++) {

if(array[a]<array[a-1]) {

//if the first value is alphabetically behind the second value this line will return a negative number.

//This can be used because whenever this occurs the two values have to be swapped

tempor=array[a];

array[a]=array[a-1];

array[a-1]=tempor;

//temporary storage needed because both values have to be overwritten so that one of the values doesn't go lost

flag=false;

//flag set to false to signal that two values were sorted

}

}

}

flag=false;

}

public static void selectionsortsort(int[] array) {

int min=0;

//minimum variable initialized

for(int b=0; b<array.length-1; b++) {

//one by one moves through unsorted part of array

min=b;

for(int c=b+1;c<array.length;c++) {

if(array[c]<array[min]) {

min=c;

//minimum value from unsorted part found

}

}

if(min!=b) {

inttemp=array[min];

array[min]=array[b];

array[b]=inttemp;

//minimum value assigned to sorted part of array. Swapped with first element

}

}

}

public static void printstatementString(String[] array) {

for(int b=0; b<array.length; b++) {

System.out.println(b+1+". "+array[b]);

//Simple statement to print the values of the array

}

}

public static void printstatementint(int[] array) {

for(int b=0; b<array.length; b++) {

System.out.println(b+1+". "+array[b]);

//Simple statement to print the values of the array. Had to be changed to integers to be applicable for the selection sort

}

}

public static void printstatementinthorizontal(int[] array) {

for(int b=0; b<array.length; b++) {

System.out.print(+array[b]+", ");

//Simple statement to print the values of the array. Had to be changed to integers to be applicable for the selection sort

}

}

public static void assignmentString(String[] array) {

for(int i=1;i<=array.length;i++) {

array[i-1] = JOptionPane.showInputDialog(null, "Input number: "+i);

}

//Simple line of code that asks the user to enter a value which is then assigned to a index of the array. Made for Bubble Sort

}

public static void assignmentint(int[] array) {

for(int i=1;i<=array.length;i++) {

array[i-1] = Integer.parseInt(JOptionPane.showInputDialog(null, "Input number: "+i));

}

//Simple line of code that asks the user to enter a value which is then assigned to a index of the array. Made for Selection Sort

}

///Bubble Sort and SelectionSort end

public static void linearSearch(String[] args) {

String a=" ";

int z=0;

String DecisionNum=JOptionPane.showInputDialog(null, "Do you want to search a words(w) or numbers array(n)");

//User decides if he wants to search for a word or a number

if(DecisionNum.equals("w")) {

linearsearchString(a);

}

if(DecisionNum.equals("n")) {

linearsearchint(z);

}

//simple if statement to implement users answer and based on that perform right method

}

public static void linearsearchString(String a) {

found=false;

found=false;

String Decision=JOptionPane.showInputDialog(null, "Do you want to create your own array that is to be searched(o) or use a sample array, a shopping list(s)");

//user can create own array that is to be searched or use sample array

Decision=Decision.toLowerCase();

//Decision converted to lower case so that program also recognises users descision if he accidentally has capslock on

if(Decision.equals("o")){

//own array

int len=Integer.parseInt(JOptionPane.showInputDialog(null, "How many words do you want to enter?"));

//to initialize array length of it needs to be known. Program asks user for this

String array[]= new String [len];

assignmentString(array);

//method which asks user to input a word for each index of the array and assigns that index to the word

SearchValue=JOptionPane.showInputDialog(null, "Enter the word you want to seach for");

//Word user searches for is inputted

linearsearchsearchString(array, SearchValue);

//Linear search performed

}

if(Decision.equals("s")){

//sample array with values "Apple","Banana","Cucumbers","Apple","Banana","Cucumber","Bread" used

SearchValue=JOptionPane.showInputDialog(null, "Enter the word you want to seach for");

//Word entered that is to be searched for

String array[]= {"Apple","Banana","Cucumbers","Apple","Banana","Cucumber","Bread"};

linearsearchsearchString(array, SearchValue);

//linear search method performed

}

}

public static void linearsearchint(int z) {

found=false;

found=false;

String Decision=JOptionPane.showInputDialog(null, "Do you want to create your own array that is to be searched(o) or use a sample array with 1000 randomly generated numbers between 1 and 100(s)");

Decision=Decision.toLowerCase();

//User can enter his decision in caps or not

if(Decision.equals("o")){

//own array used

int len=Integer.parseInt(JOptionPane.showInputDialog(null, "How many numbers do you want to enter?"));

//length of array needed to initialize it

int array[]= new int [len];

//array initialized

assignmentint(array);

//assignment method run

SearchValueint=Integer.parseInt(JOptionPane.showInputDialog(null, "Enter the number you want to seach for"));

//Search value is entered

linearsearchsearchint(array, SearchValueint);

//array and search value passed on as arguments into linear search

}

if(Decision.equals("s")){

//sample array used

SearchValueint=Integer.parseInt(JOptionPane.showInputDialog(null, "Enter the number you want to seach for"));

//Search Value entered

Random random = new Random();

int[] array = random.ints(1000, 1,101).toArray();

//Array made with length of 1000 and random numbers between 1 and 100

linearsearchsearchint(array, SearchValueint);

//linear search method run

}

}

public static void linearsearchsearchString(String[] array, String value) {

System.out.print(value+" has been found at location: ");

for(int b=0;b<array.length;b++) {

//for loop which runs through each value of the array and checks if it is equal to the search value

if(array[b].equals(value)) {

System.out.print((b+1)+", ");

//location printed out. Not index

found=true;

}

}

if(found==false) {

//if its not existing in the array

System.out.println("nowhere. The word "+value+" does not exist in this array");

}

}

public static void linearsearchsearchint(int[] array, int value) {

//linear search for integers

System.out.print(value+" has been found at location: ");

for(int b=0;b<array.length;b++) {

//again loops through each index of the array and compares it to the number

if(array[b]==value){

System.out.print((b+1)+", ");

found=true;

}

}

if(found==false) {

//if not found this message lets the user know that the search value doesnt exist in the array

System.out.println("nowhere. The number "+value+" does not exist in this array");

}

}

///Binary Search

public static void Binary(String[] args) {

// TODO Auto-generated method stuff;

String Decision=JOptionPane.showInputDialog(null, "Do you want to perform a binary search on your own array(o) It has to be sorted though, or on a sample array which is already sorted(s)");

//user can perform a binary search on an array of his choosing or a sample array

Decision=Decision.toLowerCase();

//margin for error. Decision still recognized if written in caps

if(Decision.equals("s")) {

//user picked sample array

int SearchValue=Integer.parseInt(JOptionPane.showInputDialog(null, "Enter the number you want to search for"));

//user enters number he wants to search for

int array[]= {1,2,3,4,5,6,7,8,9,10,11,12};

//pre-made array using values 1,2,3,4,5,6,7,8,9,10,11,12

binarysearch(array, 0, array.length, SearchValue);

//binary search method performed in which the arguments are 1. The array itself 2. The minimum value 3. The maximum value 3. The number that is to be searched for

System.out.println("Found at location:"+index+ErrorMessage);

}

//message that can be read in console is outputted

if(Decision.equals("o")) {

//user chooses to make his own array

int len=Integer.parseInt(JOptionPane.showInputDialog(null, "How many numbers do you want to include in your array"));

//self made array requires length to be known in advance

int array[]=new int [len];

//array initialized

assignmentint(array);

//arrays indexes set equal to user input

bubblesortsortint(array);

System.out.println("The sorted array is:");

printstatementinthorizontal(array);

int SearchValue=Integer.parseInt(JOptionPane.showInputDialog(null, "Enter the number you want to search for"));

//number to be searched for is entered

binarysearch(array, 0, array.length-1, SearchValue);

//binary search method performed in which the arguments are 1. The array itself 2. The minimum value 3. The maximum value 3. The number that is to be searched for

System.out.println("Found at location:"+index+ErrorMessage);

//Message read in console outputted

}

}

public static void binarysearch(int[] array, int min,int max,int SearchValue) {

//actual binary search

int mid=-1;

//variable mid initialized

while(Found==false) {

//while loop using a flag that breaks when variable has been found

tempor=mid;

//value of mid temporarily stored in order to check if variable is not in array as only if the first value of mid

//is equal to its changed value if the array is stuck in an infinite loop which happens when

//the value doesn't exist in the array

mid =(min+max-1)/2;

//finds middle of array and narrows down towards the value that is to be searched for

if(mid>=array.length) {

Found=true;

ErrorMessage="The array is not sorted";

//checks if array isn't sorted and if error message would happen

}

if(array[mid]==SearchValue) {

Found=true;

index=mid+1;

//if SearchValue is exactly at the middle

}

if(array[mid]>SearchValue) {

max=mid;

//maximum lowered to narrow down array

}

else{

min=mid;

//minimum increased to narrow down array

}

if(tempor==mid) {

Found=true;

ErrorMessage=" The value is at location "+tempor;

//error check if value is not in array

}

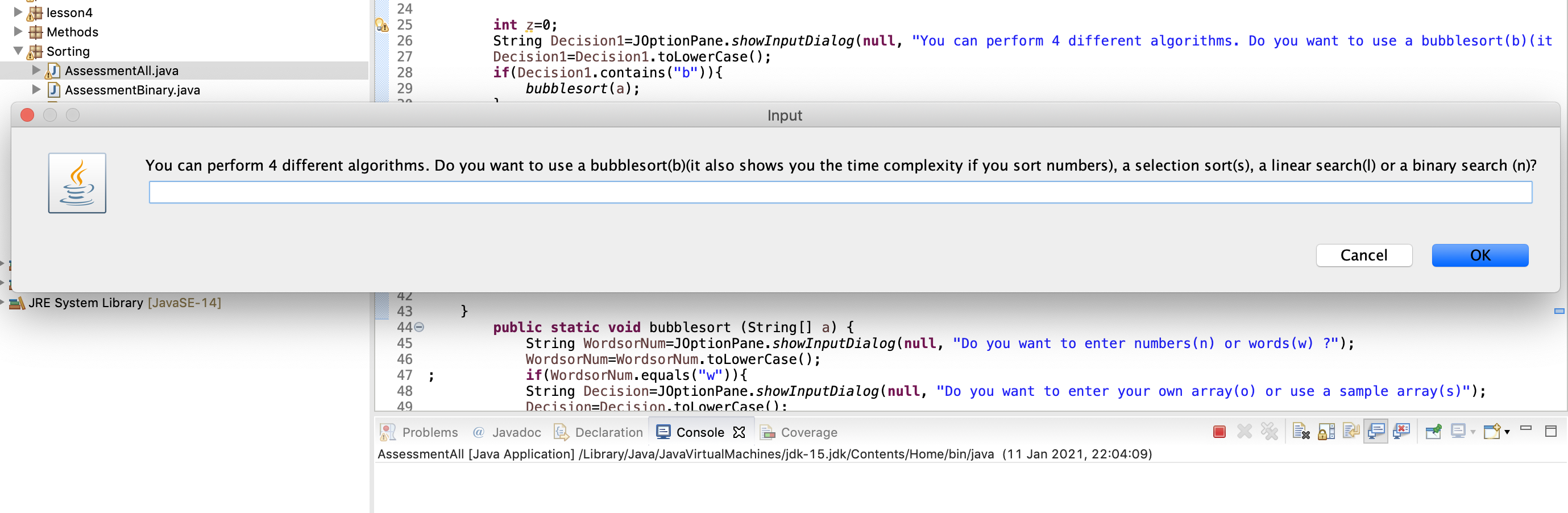
}

}

}

I will only be displaying one path for every algorithm in pictures. The other paths (sorting words etc.) also work.

Bubble Sort



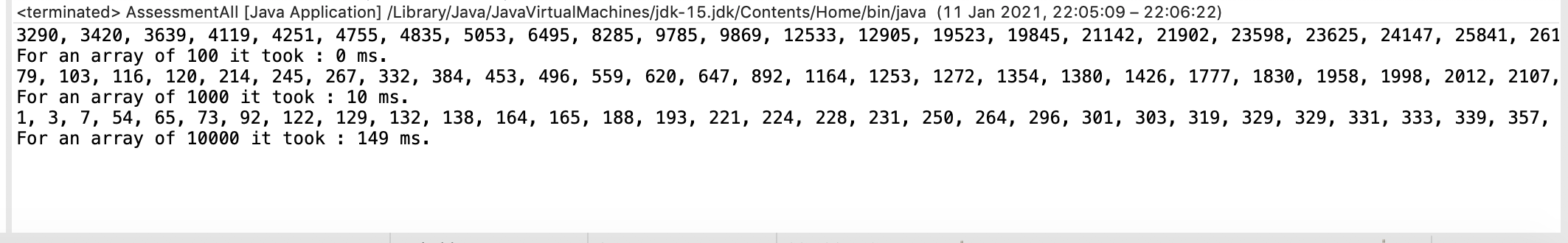
BubbleSort:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, Word

Description automatically generated

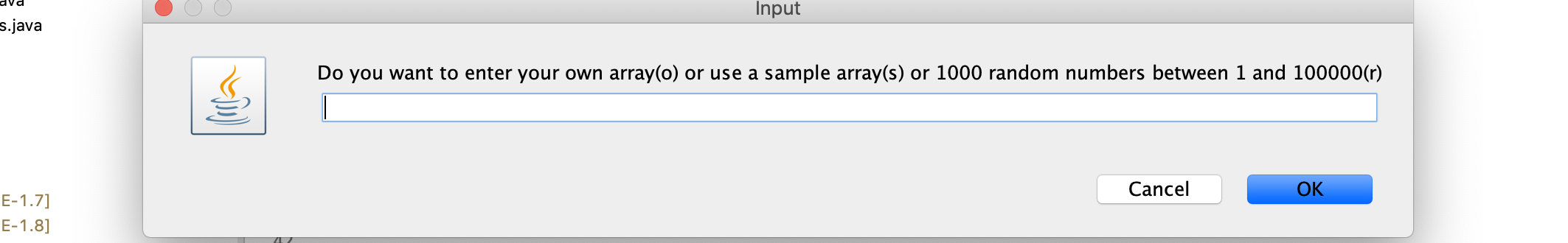


Graphical user interface, text, application

Description automatically generated

It can also ask the user to enter words which are then sorted using a bubble sort.

Selection Sort:



Graphical user interface, text, application

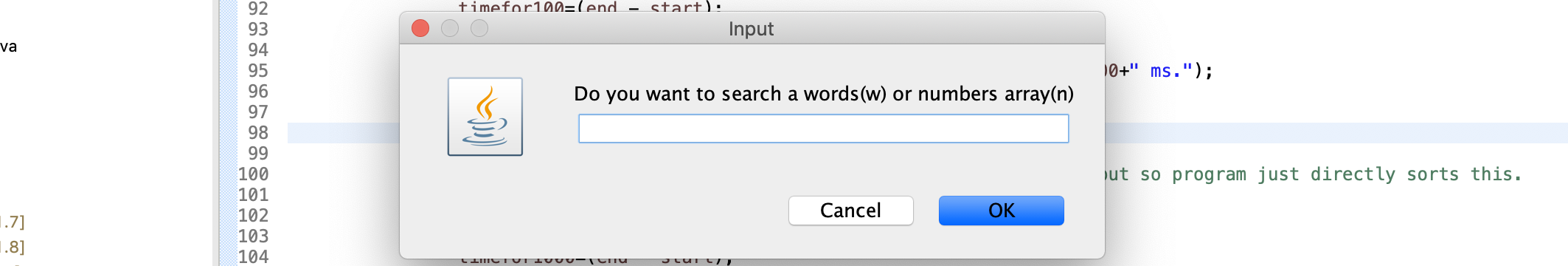
Description automatically generated

Graphical user interface, text, application

Description automatically generated

It also lets user enter his own numbers for the array.

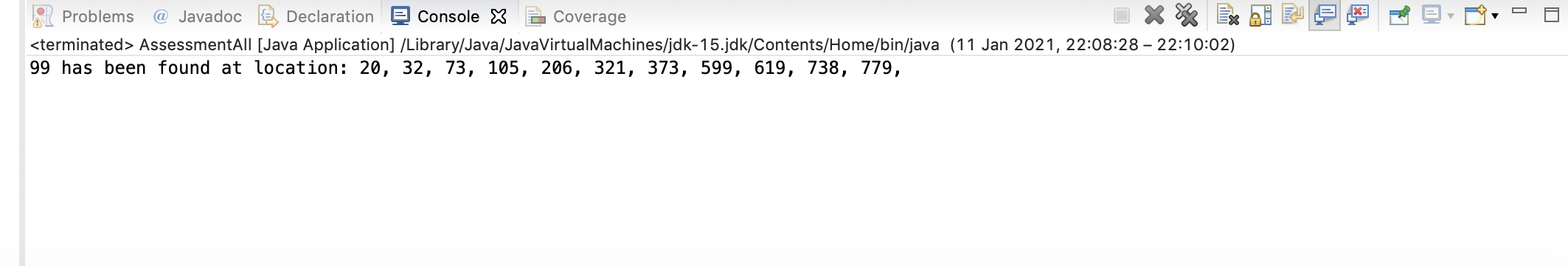
Linear Search:



Graphical user interface, text, application

Description automatically generated

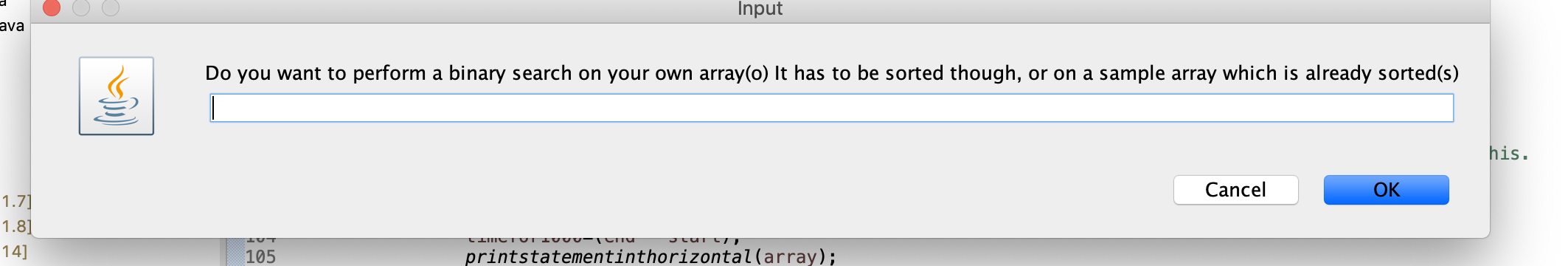
Graphical user interface, application

Description automatically generated



Also lets user create their own array by entering words into JOptionPane and then to search this.

Binary Search



Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Let’s user enter his own numbers which are then sorted by a bubble sort and then searched for a certain value.

Other Paths in program such as sorting words, creating your own arrays that are to be searched etc. can also be chosen in the program.