**Sequential Search**

1. Write a method seqSearch that takes as parameters an array of String list and a single String item. It searches item in list and returns the index of item in the array if successful, and -1 if not. Create a main method in the class to test the method.

| /\*  \* Program name: SequentialSearch.java  \*  \* By: Lucas Chow (Last edited: 2022-10-17)  \*  \* ICS4U1: 1 - Sequential Sorting  \*  \* This program has a method called "seqSearch" that uses sequential sorting (aka linear sorting)  \* To looping through an inputted array until the input item equals the same element in the array  \*  \*/  public class SequentialSearch{  /\*  \* int seqSearch(String[] list, String item)  \*  \* String[] list -> this is the inputted array of Strings  \*  \* String item -> this is the item the method is trying to locate  \*  \* This method uses sequential search to search for an item in a list  \*  \*  \*/  public static int seqSearch(String[] list, String item)  {  int output;  output = -1;  for (int i = 0; i < list.length; i++)  {  if (item.equals(list[i]))  {  output = i;  }  }  return output;  }      public static void main(String[] args)  {  ///test values, testing the sequential search  String[] testList = {"apple","banana","orange"};  String testItem = "orange";  System.out.println("Array: {\"apple\",\"banana\",\"orange\"}");  System.out.print("The index of the String \"orange\" is: ");  System.out.println(seqSearch(testList, testItem));  }  } |
| --- |

1. Write a program that creates an array of 20 integers and fills it with random numbers. It prompts user to enter a number, then searches, using the sequential search algorithm, for the number and outputs its position or indicates if the number is not present in the array.

| /\*  \* Program name: RandomSequentialSearch.java  \*  \* By: Lucas Chow (Last edited: 2022-10-30)  \*  \* ICS4U1:  \*  \* This program creates an array for 20 random number, prompting  \* the user to enter a number. If the number appears in the array,  \* it returns the index  \*  \*  \*/  //importing scanner  import java.util.\*;  public class RandomSequentialSearch{  public static void main(String[] args)  {    //initializing variables  final int ARRAY\_LENGTH = 20;  int input;  int index;  int count;  int[] randomIntArray = new int[ARRAY\_LENGTH];  Scanner sc = new Scanner(System.in);    //creating array of random integers  for (int i = 0; i < randomIntArray.length; i++)  {  randomIntArray[i] = (int) Math.round(100\*(Math.random()));  }      //prompting user for a number from 1 to 100 to check if it exist in the array  System.out.print("Enter a number from 1 to 100: ");  input = Integer.parseInt(sc.nextLine());    //using the sequential search algorithm  index = -1;  count = 0;  for (int i = 0; i < randomIntArray.length; i++)  {  if (input == randomIntArray[i])  {  index = i;  count++;  }  }    //Outputs if the integer appears and if so how many times, otherwise outputing it does not exist  if (count != 0)  {  System.out.println("The input "+input+" is located at index "+ index +" and occurs for a count of " +count);  }  else  {  System.out.println("input does not appear in the array");  }    //closing scanner  sc.close();  }  } |
| --- |

1. Write a program that does exactly the same job as question 1, except that the sequential search algorithm searches the array of values starting at the top and moving downward.

| /\*  \* Program name: RandomReverseSequentialSearch.java  \*  \* By: Lucas Chow (Last edited: 2022-10-30)  \*  \* ICS4U1: Sequential Search  \*  \* This program uses sequential search, but instead loops backwards, similiar to SequentialSearch.java  \*  \* It then test it in the main class  \*  \*  \*/  public class ReverseSequentialSearch{  /\*  \* int reverseSeqSearch(String[] list, String item)  \*  \* String[] list -> this is the inputted array of Strings  \*  \* String item -> this is the item the method is trying to locate  \*  \* This method uses sequential search to search for an item in a list in a reversed fashion  \*  \*  \*/    public static int reverseSeqSearch(String[] list, String item)  {  int output;  output = -1;  for (int i = list.length-1; i > 0; i--)  {  if (item.equals(list[i]))  {  output = i;  }  }  return output;  }    public static void main(String[] args)  {  ///test values, testing the Reverse sequential search  String[] testList = {"apple","banana","orange"};  String testItem = "orange";  System.out.println("Array: {\"apple\",\"banana\",\"orange\"}");  System.out.print("The index of the String \"orange\" is: ");  System.out.println(reverseSeqSearch(testList, testItem));  }  } |
| --- |

1. A modification of the basic sequential search operates in the following way. If the item being sought is found, it is interchanged with the item that preceded it. If, for example, we were searching for 7 in the list

| 3 | 9 | 5 | 7 | 2 | 8 | 4 |
| --- | --- | --- | --- | --- | --- | --- |

then, after finding 7, the list would be rearranged in the order

| 3 | 9 | 7 | 5 | 2 | 8 | 4 |
| --- | --- | --- | --- | --- | --- | --- |

1. Write a method that implements this technique to search an array of int values.
2. Test your method in a program that first asks for the length of the list to be searched and then reads that many integers into an array. The program should then repeatedly prompt the user for values until the user supplies a sentinel of zero. The program should print the initial list and then, for each non-zero value read, it should use your modified sequential search to try to locate that item and then print the resulting array.
3. Why might this modification sometimes improve the efficiency of a sequential search?

| /\*  \* Program name: modifiedSequentialSearch.java  \*  \* By: Lucas Chow (Last edited: 2022-10-30)  \*  \* ICS4U1: Sequential Search  \*  \* This program uses an iteration of sequential search that  \* swaps the sought value with the preceding value. It then  \* prompts the user to enter values,  \*  \*  \*/  //importing scanner  import java.util.Scanner;  public class modifiedSequentialSearch {  /\*  \* int specialSequentialSearch(int[] array, int item)  \*  \* int[] array -> this is the input array  \*  \* int item -> the item the program is looking for  \*  \* this method is an interation on sequential search that swaps the sought integer with the preceding value,  \* if not with the last value.  \*  \*/  public static int specialSequentialSearch(int[] array, int item)  {  int index;  int temp;  index = -1;  for (int i = 0; i < array.length; i++)  {  if (array[i] == item)  {  index = i;  //swapping array[i], array[i-1]  if (i != 0)  {  temp = array[i];  array[i] = array[i-1];  array[i-1] = temp;  }  else  {  temp = array[i];  array[i] = array[array.length-1];  array[array.length-1] = temp;  }  }  }    return index;  }      public static void main(String[] args)  {    //declaring variables and objects  Scanner sc = new Scanner(System.in);  int[] array;  int array\_length;  int input;    //prompting user for array length  System.out.print("Enter the length of the array: ");  array\_length = sc.nextInt();  array = new int[array\_length];    //prompting user for array values  for (int i = 0; i < array.length; i++)  {  System.out.print("Enter array value for index "+ i + ": ");  array[i] = sc.nextInt();  }    //exit request message  System.out.println("Enter 0 to quit");    //prompting user to fill in values to search array with  do  {  System.out.print("Enter input to search array with: ");  input = sc.nextInt();  if (input != 0)  {  System.out.print("Array before: ");  System.out.print("[");  for (int i = 0; i < array.length; i++)  {  System.out.print(array[i]+",");  }  System.out.println("]");    System.out.println("Item located at index " + specialSequentialSearch(array, input));    System.out.print("Array after: ");  System.out.print("[");  for (int i = 0; i < array.length; i++)  {  System.out.print(array[i]+",");  }  System.out.println("]");    }  } while (input != 0);    //closing scanner  sc.close();  }  } |
| --- |