**Bubble Sort**

1. Make a table to show the comparisons and exchanges that would take place in using a bubble sort to put the following data in ascending order.

| 3 | 8 | 3 | 2 | 7 |
| --- | --- | --- | --- | --- |

| Sorting i = 0 to array.length - 1  Pre-Condition: 3 8 3 2 7  **3 8** 3 2 7 to 8 3 3 2 7  8 **3 3** 2 7 to **same** (stable algorithm, no swap)  8 3 **3 2** 7 to **same** (no swap)  8 3 3 **2 7** to 8 3 3 7 2  (First pass: 8 3 3 7 2)  **8 3** 3 7 2 to 8 3 3 7 2  8 **3 3** 7 2 to **same**  (Stable alg, no swap)  8 3 **3 7** 2 to **same** (Stable alg, no swap)  8 3 7 **3 2** to **same**  (Second Pass: 8 3 7 3 2)  **8 3** 7 3 2 to 8 3 7 3 2  8 **3 7** 3 2 to 8 7 3 3 2  8 7 **3 3** 2 to 8 7 3 3 2 (Stable alg, no swap)  8 7 3 **3 2** to 8 7 3 3 2  (Third Pass: 8 7 3 3 2)  **8 7** 3 3 2 (No swaps)  8 **7 3** 3 2 (No swaps)  8 7 **3 3** 2 (No swaps)  8 7 3 **3 2** (No swaps)  Fourth Pass (Sorted) 8 7 3 3 2 | Output (If printed every pass)  Pre condition:  3 8 3 2 7  First Pass:  8 3 3 7 2  Second Pass;  8 3 7 3 2  Third Pass:  8 7 3 3 2  Fourth Pass (Sorted):  8 7 3 3 2 |
| --- | --- |

1. Create a method bubbleSort that uses a bubble sort to arrange an array of double values in ascending order. Create a main method in the class to test the method.

| /\*  \* Program name: BubbleSort.java  \*  \* By: Lucas Chow (Last edited: 2022-10-30)  \*  \* ICS4U1: BubbleSort  \*  \* This program uses bubble Sort, which swaps adjacent values to sort, which it remembers the swap  \* (Doesn't check the last element for this iteration as x+1 will be out of bounds).  \* o(n^2) worst case, o(n) best case.  \*  \* This program then test it on some test data  \*  \*/  public class BubbleSort{  /\*  \* void bubbleSort(int[] inputArr)  \*  \* int[] inputArr -> The inputed array  \*  \* This method uses bubble sort to sort an numerical data set  \*  \*  \*/  public static void bubbleSort(int[] inputArr)  {  boolean ifSorted;  int temp;  int i = 0;    ifSorted = false;  while (i < inputArr.length-1 && !ifSorted)  {  ifSorted = true;  for (int x = 0; x < inputArr.length-1; x++)  {  if (inputArr[x] > inputArr[x+1])  {  ifSorted=false;  temp = inputArr[x];  inputArr[x] = inputArr[x+1];  inputArr[x+1] = temp;  }  }  }  }  public static void main(String[] args)  {  int[] testArr = {1,2,3,43,6,3,6,99,101,103,102};    //precondition of program  System.out.print("Array before: {");  for (int i = 0; i < testArr.length; i++)  {  System.out.print(testArr[i]+",");  }  System.out.println("}");    //performing insertion sort  bubbleSort(testArr);    //post-condition of the program  System.out.print("Array After: {");  for (int i = 0; i < testArr.length; i++)  {  System.out.print(testArr[i]+",");  }  System.out.println("}");    }  } |
| --- |

1. A modification of bubble sort is the *cocktail shaker sort* in which, on odd-numbered passes, large values are carried to the top of the list while, on even-number passes, small values are carried to the bottom of the list. Make a table to show the first two passes of a cocktail-shaker sort on the following data.

| 2 | 9 | 4 | 6 | 1 | 7 | 8 |
| --- | --- | --- | --- | --- | --- | --- |

| Pre-Condition:  2 9 4 6 1 7 8  First Pass:  2 9 4 6 1 7 8 to 2 9 4 6 1 7 8  2 9 4 6 1 7 8 to 2 4 9 6 1 7 8  2 4 9 6 1 7 8 to 2 4 6 9 1 7 8  2 4 6 9 1 7 8 to 2 4 6 1 9 7 8  2 4 6 9 1 7 6 to 2 4 6 1 7 9 8  2 4 6 1 7 9 8 to 2 4 6 1 7 8 9  Second Pass:  2 4 6 1 7 8 9 to 2 4 6 1 7 8 9  …  2 4 **6 1** 7 8 9 to 2 4 1 6 7 8 9  2 4 1 6 7 8 9 to 2 1 4 6 7 8 9  2 1 4 6 7 8 9 to 1 2 4 6 7 8 9 | First Pass: 2 4 6 1 7 8 9  Second Pass: 1 2 4 6 7 8 9 |
| --- | --- |

1. Write a method shakeSort to implement the cocktail shaker sort algorithm to arrange an array of double values in ascending order. Use a Boolean flag to stop processing once the items have been completely sorted.

| /\*  \* Program name: CocktailShakerSort.java  \*  \* By: Lucas Chow (Last edited: 2022-)  \*  \* ICS4U1: Bubble Sort  \*  \* This uses CocktailSort, an iteration on bubble sort, where it alternates from moving up and down the array sorting lowest to highest  \*  \*  \*/  public class CocktailShakerSort{  /\*  \* cocktailShakerSort(int[] inputArr)  \*  \* inputArr -> the input array  \*  \* This method uses cocktailSort, an iteration on bubble sort, where it goes up on odd passes, down on even passes  \*  \*  \*/  public static void cocktailShakerSort(int[] inputArr)  {  boolean ifSorted;  int temp;  int i = 0;    ifSorted = false;  while (i < inputArr.length-1 && !ifSorted)  {  ifSorted = true;  //odd passes, sorts upwards  if (i%2 == 1)  {  for (int x = 0; x < inputArr.length-1; x++)  {  if (inputArr[x] > inputArr[x+1])  {  ifSorted=false;  temp = inputArr[x];  inputArr[x] = inputArr[x+1];  inputArr[x+1] = temp;  }  }  }  //even passes, sorts downwards  else if (i%2 == 0)  {  for (int x = inputArr.length-1; x > 0; x--)  {  if (inputArr[x] < inputArr[x-1])  {  ifSorted=false;  temp = inputArr[x];  inputArr[x] = inputArr[x-1];  inputArr[x-1] = temp;  }  }  }    }  }  public static void main(String[] args)  {  int[] testArr = {1,2,3,43,6,3,6,99,101,103,102};    //precondition of program  System.out.print("Array before: {");  for (int i = 0; i < testArr.length; i++)  {  System.out.print(testArr[i]+",");  }  System.out.println("}");    //performing insertion sort  cocktailShakerSort(testArr);    //post-condition of the program  System.out.print("Array After: {");  for (int i = 0; i < testArr.length; i++)  {  System.out.print(testArr[i]+",");  }  System.out.println("}");  }  } |
| --- |