**TECHNOLOGICAL INSTITUTE OF THE PHILIPPINES**

**QUEZON CITY**

**COLLEGE OF INFORMATION TECHNOLOGY EDUCATION (CITE)**

**CS 201 - Data Structures and Algorithms**

|  |  |
| --- | --- |
| **Name: Aristotle Buenaventura** | **Date: September 22, 2021** |
| **Program/Section: IT21S1** | **Instructor: Ms. Rosmina Joy M. Cabauatan** |
| **Assessment Task: Quiz 3** | |

Bubble Sort

|  |  |
| --- | --- |
| Steps | Simulation |
| There’s no swapping because 4<10 | 4 10 12 5 1 12 6 8 |
| There’s no swapping because 10<12 | 4 10 12 5 1 12 6 8 |
| Since 12 is greater than 5, swap 12 and 5 | 4 10 12 5 1 12 6 8 |
| Since 12 is greater than 1, swap 12 and 1 | 4 10 5 12 1 12 6 8 |
| There’s no swapping because 12=12 | 4 10 5 1 12 12 6 8 |
| Since 12 is greater than 6, swap 12 and 6 | 4 10 5 1 12 12 6 8 |
| Since 12 is greater than 8, swap 12 and 8 | 4 10 5 1 12 6 12 8 |
| There’s no swapping because 4<10 | 4 10 5 1 12 6 8 12 |
| Since 10 is greater than 5, swap 10 and 5 | 4 10 5 1 12 6 8 12 |
| Since 10 is greater than 1, swap 10 and 1 | 4 5 10 1 12 6 8 12 |
| There’s no swapping because 10<12 | 4 5 1 10 12 6 8 12 |
| Since 12 is greater than 6, swap 12 and 6 | 4 5 1 10 12 6 8 12 |
| Since 12 is greater than 8, swap 12 and 8 | 4 5 1 10 6 12 8 12 |
| There’s no swapping because 12=12 | 4 5 1 10 6 8 12 12 |
| There’s no swapping because 4<5 | 4 5 1 10 6 8 12 12 |
| Since 5 is greater than 1, swap 5 and 1 | 4 5 1 10 6 8 12 12 |
| There’s no swapping because 5<10 | 4 1 5 10 6 8 12 12 |
| Since 10 is greater than 6, swap 10 and 6 | 4 1 5 10 6 8 12 12 |
| Since 10 is greater than 8, swap 10 and 8 | 4 1 5 6 10 8 12 12 |
| There’s no swapping because 10<12 | 4 1 5 6 8 10 12 12 |
| There’s no swapping because 12=12 | 4 1 5 6 8 10 12 12 |
| Since 4 is greater than 1, swap 4 and 1 | 4 1 5 6 8 10 12 12 |
| There’s no swapping because 4<5 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 5<6 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 6<8 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 8<10 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 10<12 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 10<12 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 1<4 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 4<5 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 5<6 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 6<8 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 8<10 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 10<12 | 1 4 5 6 8 10 12 12 |
| There’s no swapping because 10<12 | 1 4 5 6 8 10 12 12 |
| Sorted Array using Bubble Sort | 1 4 5 6 8 10 12 12 |

Selection Sort

|  |  |
| --- | --- |
| Steps | Simulation |
| Swap 4 and 1, since 1 is the lowest value | 4 10 12 5 1 12 6 8 |
| Swap 10 and 4, since 4 is the lowest value | 1 10 12 5 4 12 6 8 |
| Swap 12 and 5, since 5 is the lowest value | 1 4 12 5 10 12 6 8 |
| Swap 12 and 6, since 6 is the lowest value | 1 4 5 12 10 12 6 8 |
| Swap 10 and 8, since 8 is the lowest value | 1 4 5 6 10 12 12 8 |
| Swap 12 and 10, since 10 is the lowest value | 1 4 5 6 8 12 12 10 |
| Sorted Array using Selection Sort | 1 4 5 6 8 10 12 12 |

Insertion Sort

|  |  |
| --- | --- |
| Steps | Simulation |
| Both 4 and 10 are already in ascending order. | 4 10 12 5 1 12 6 8 |
| Both 10 and 12 are already in ascending order. | 4 10 12 5 1 12 6 8 |
| Swap 12 and 5 since 12 > 5 | 4 10 12 5 1 12 6 8 |
| Swap 10 and 5 since 10 > 5 | 4 10 5 12 1 12 6 8 |
| Both 4 and 5 are already in ascending order. | 4 5 10 12 1 12 6 8 |
| Both 5 and 10 are already in ascending order. | 4 5 10 12 1 12 6 8 |
| Both 10 and 12 are already in ascending order. | 4 5 10 12 1 12 6 8 |
| Swap 12 and 1 since 12 > 1 | 4 5 10 12 1 12 6 8 |
| Swap 10 and 1 since 10 > 1 | 4 5 10 1 12 12 6 8 |
| Swap 5 and 1 since 5 > 1 | 4 5 1 10 12 12 6 8 |
| Swap 4 and 1 since 4 > 1 | 4 1 5 10 12 12 6 8 |
| Both 4 and 5 are already in ascending order. | 1 4 5 10 12 12 6 8 |
| Both 5 and 10 are already in ascending order. | 1 4 5 10 12 12 6 8 |
| Both 10 and 12 are already in ascending order. | 1 4 5 10 12 12 6 8 |
| Both 12 and 12 are already in ascending order. | 1 4 5 10 12 12 6 8 |
| Swap 12 and 6 since 12 > 6 | 1 4 5 10 12 12 6 8 |
| Swap 12 and 6 since 12 > 6 | 1 4 5 10 12 6 12 8 |
| Swap 10 and 6 since 10 > 6 | 1 4 5 10 6 12 12 8 |
| Both 5 and 6 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Both 6 and 10 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Both 10 and 12 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Both 12 and 12 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Swap 12 and 8 since 12 > 8 | 1 4 5 6 10 12 12 8 |
| Swap 12 and 8 since 12 > 8 | 1 4 5 6 10 12 8 12 |
| Swap 10 and 8 since 10 > 8 | 1 4 5 6 10 8 12 12 |
| Both 6 and 8 are already in ascending order. | 1 4 5 6 8 10 12 12 |
| Both 8 and 10 are already in ascending order. | 1 4 5 6 8 10 12 12 |
| Both 10 and 12 are already in ascending order. | 1 4 5 6 8 10 12 12 |
| Both 12 and 12 are already in ascending order. | 1 4 5 6 8 10 12 12 |
| Sorted Array using Insertion Sort | 1 4 5 6 8 10 12 12 |

Shell Sort

|  |  |
| --- | --- |
| Steps | Simulation |
| Make a virtual sub-list of all values located at the interval of 4 positions. Here these values are {4,1}, {10,12},{12,6}, and {5,8} | 4 10 12 5 1 12 6 8 |
| We compare values in each sub-list and swap them (if necessary) in the original array. | 1 10 6 5 4 12 12 8 |
| Then, we take interval of 1 and this gap generates two sub-lists – {1,6,4,12} and {10,5,12,8} | 1 10 6 5 4 12 12 8 |
| We compare and swap the values, if required, in the original array | 1 10 6 5 4 12 12 8 |
| Both 1 and 10 are already in ascending order. | 1 10 6 5 4 12 12 8 |
| Swap 10 and 6 since 10 > 6 | 1 10 6 5 4 12 12 8 |
| Both 1 and 6 are already in ascending order. | 1 6 10 5 4 12 12 8 |
| Swap 10 and 5 since 10 > 5 | 1 6 10 5 4 12 12 8 |
| Swap 6 and 5 since 6 > 5 | 1 6 5 10 4 12 12 8 |
| Both 1 and 5 are already in ascending order. | 1 5 6 10 4 12 12 8 |
| Both 5 and 6 are already in ascending order. | 1 5 6 10 4 12 12 8 |
| Both 6 and 10 are already in ascending order. | 1 5 6 10 4 12 12 8 |
| Swap 10 and 4 since 10 > 4 | 1 5 6 10 4 12 12 8 |
| Swap 6 and 4 since 6 > 4 | 1 5 6 4 10 12 12 8 |
| Swap 5 and 4 since 5 > 4 | 1 5 4 6 10 12 12 8 |
| Both 1 and 4 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Both 4 and 5 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Both 5 and 6 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Both 6 and 10 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Both 10 and 12 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Both 12 and 12 are already in ascending order. | 1 4 5 6 10 12 12 8 |
| Swap 12 and 8 since 12 > 8 | 1 4 5 6 10 12 12 8 |
| Swap 12 and 8 since 12 > 8 | 1 4 5 6 10 12 8 12 |
| Swap 10 and 8 since 12 > 8 | 1 4 5 6 10 8 12 12 |
| Both 6 and 8 are already in ascending order. | 1 4 5 6 8 10 12 12 |
| Both 8 and 10 are already in ascending order. | 1 4 5 6 8 10 12 12 |
| Both 10 and 12 are already in ascending | 1 4 5 6 8 10 12 12 |
| Both 12 and 12 are already in ascending | 1 4 5 6 8 10 12 12 |
| Sorted Array using Shell Sort | 1 4 5 6 8 10 12 12 |

Heap Sort

*Build a Heap*

|  |  |  |
| --- | --- | --- |
| Heap | Newly added element | Swap elements |
| Null | 4 |  |
| 4 | 10 |  |
| 4,10 | 12 |  |
| **12**,10,**4** |  | 4,12 |
| 12,10,4 | 5 |  |
| 12,10,4,5 | 1 |  |
| 12,10,4,5,1 | 12 |  |
| 12,10,**12**,5,1,**4** |  | 4,12 |
| 12,10,12,5,1,4 | 6 |  |
| 12,10,12,5,1,4,6 |  |  |
| 12,10,12,5,1,4,6 | 8 |  |
| 12,10,12,**8**,1,4,6,**5** |  | 5,8 |

*Sort the Heap*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Heap | Swap elements | Delete element | Sorted array | Details |
| **12**,10,12,8,1,4,6,**5** | 12,5 |  |  | Swap 12 and 5 in order to delete 12 from heap |
| 5,10,12,8,1,4,6,**12** |  | 12 |  | Delete 12 from heap and add to sorted array |
| **5**,10,**12**,8,1,4,6 | 5,12 |  | 12 | Swap 5 and 12 as they are not in order in the heap |
| 12,10,**5**,8,1,4,**6** | 5,6 |  | 12 | Swap 5 and 6 as they are not in order in the heap |
| **12**,10,6,8,1,4,**5** | 12,5 |  | 12 | Swap 12 and 5 in order to delete 12 from heap |
| 5,10,6,8,1,4,**12** |  | 12 | 12 | Delete 12 from heap and add to sorted array |
| **5**,**10**,6,8,1,4 | 5,10 |  | 12,12 | Swap 5 and 10 as they are not in order in the heap |
| 10,**5**,6,**8**,1,4 | 5,8 |  | 12,12 | Swap 5 and 8 as they are not in order in the heap |
| **10**,8,6,5,1,**4** | 10,4 |  | 12,12 | Swap 10 and 4 in order to delete 10 from heap |
| 4,8,6,5,1,**10** |  | 10 | 12,12 | Delete 10 from heap and add to sorted array |
| **4**,**8**,6,5,1 | 4,8 |  | 10,12,12 | Swap 4 and 8 as they are not in order in the heap |
| 8,**4**,6,**5**,1 | 4,5 |  | 10,12,12 | Swap 4 and 5 as they are not in order in the heap |
| **8**,5,6,4,**1** | 8,1 |  | 10,12,12 | Swap 8 and 1 in order to delete 8 from heap |
| 1,5,6,4,**8** |  | 8 | 10,12,12 | Delete 8 from heap and add to sorted array |
| **1**,5,**6**,4 | 1,6 |  | 8,10,12,12 | Swap 1 and 6 as they are not in order in the heap |
| **6**,5,1,**4** | 6,4 |  | 8,10,12,12 | Swap 6 and 5 in order to delete 6 from heap |
| 4,5,1,**6** |  | 6 | 8,10,12,12 | Delete 6 from heap and add to sorted array |
| **4**,**5**,1 | 4,5 |  | 6,8,10,12,12 | Swap 4 and 5 as they are not in order in the heap |
| **5**,4,**1** | 5,1 |  | 6,8,10,12,12 | Swap 5 and 1 in order to delete 5 from heap |
| 1,4,**5** |  | 5 | 5,6,8,10,12,12 | Delete 5 from heap and add to sorted array |
| **1**,**4** | 1,4 |  | 5,6,8,10,12,12 | Swap 1 and 4 as they are not in order in the heap |
| **4**,**1** | 4,1 |  | 5,6,8,10,12,12 | Swap 4 and 1 in order to delete 4 from heap |
| 1,**4** |  | 4 | 5,6,8,10,12,12 | Delete 4 from heap and add to sorted array |
| **1** |  | 1 | 4,5,6,8,10,12,12 | Delete 1 from heap and add to sorted array |
|  |  |  | 1,4,5,6,8,10,12,12 | completed |