**SVKM’s NMIMS**

**Mukesh Patel School of Technology Management & Engineering**

**Computer Engineering Department**

Program: B.Tech. Sem V

**Course: Design and Analysis of Algorithms**

w.e.f. 1st Jul 2020

**Faculty:** Abhay Kolhe.

LAB Manual 1

PART B

|  |  |
| --- | --- |
| Roll No. B032 | Name: Naman Garg |
| Class : Btech CS B | Batch : B2 |
| Date of Experiment: 14-7-2020 | Date of Submission |
| Grade : | Time of Submission: |
| Date of Grading: |  |

**B.1 Software Code written by student:**

# Name: naman garg RollNo: B032

# lab 1 DAA

# AIM: Implementation of Insertion Sort.

# driver code.

if \_\_name\_\_ == "\_\_main\_\_":

    # taking input, storing as a list

    print("enter your numbers")

    l1 = list(map(int, input().split()))

# initalizing the no. of total swaps and comparisons

    ctr1, ctr2 = 0, 0

# pritning the inputed array as it is

    print("Initial Array: ", l1)

# the for loop works for the length of the list i.e the no. of

    for i in range(1, len(l1)):

        # initalizing variables for current iteration's swaps and comparisons

        temp1, temp2 = 0, 0

        x = i

        for j in range(i-1, -1, -1):

            # increasing the no of comparisons by 1

            temp2 += 1

            # swaping inside this if

            if l1[j] > l1[x]:

                l1[x], l1[j] = l1[j], l1[x]

            # increasing the no of swaps by 1

                temp1 += 1

                x -= 1

            else:

                break

        ctr1 += temp1

        ctr2 += temp2

        # pritning info

        print(f"\nPass {i}: ", l1)

        print("No of Swaps Done: ", temp1)

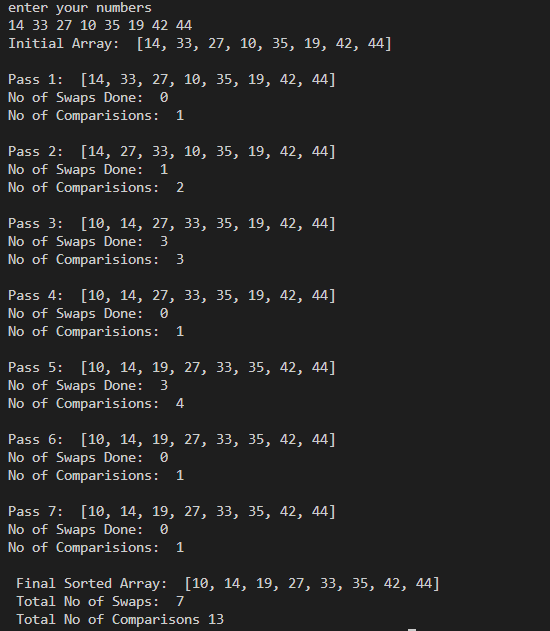
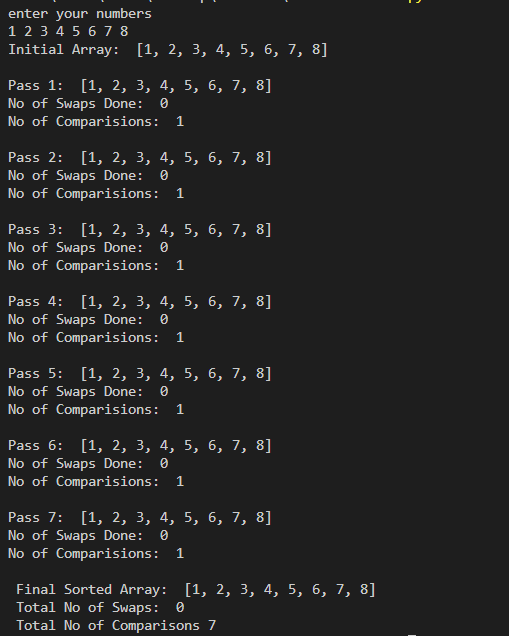
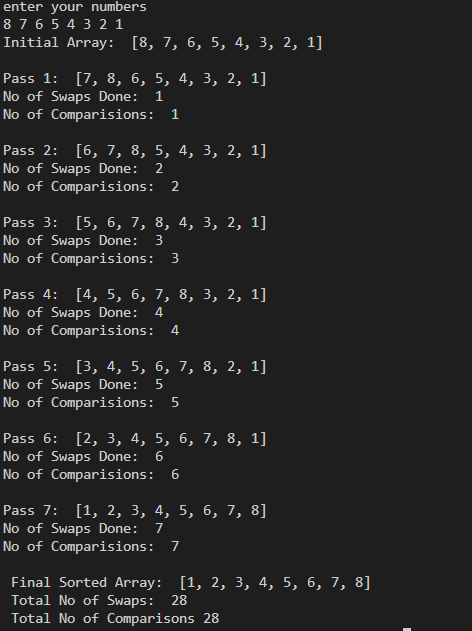
        print("No of Comparisions: ", temp2)

    # pritning final info

    print("\n Final Sorted Array: ", l1)

    print(" Total No of Swaps: ", ctr1)

    print(" Total No of Comparisons", ctr2)

**B.2 Input and Output: **  

**B.3 Observations and learning:**

I absorbed observed that the number of swaps or number of comparisons that is a total number of swaps or total number of comparisons are higher for a more unsorted array I also understood that the total number of comparisons and total number of swaps ARE directly related to the time complexity of an algorithm or the total amount of time it takes for its execution These were preliminary observations made during the program insertion sort is a fairly straightforward way of sorting elements but is very inefficient when it comes to large amount of data

**B.4 Conclusion:**

In conclusion, the insertion sort is an uncomplicated way of sorting elements but it should be used only when the sortedness of the arrays is insured that is it is already pre sorted in some ways or we have similar range of data sorted we implemented the insertion sort and kept track of the number of comparison run swaps this allowed us to analyze the algorithm further by comparing the number of swaps and comparisons to directly correlated to the amount of time it takes to be executed since these are computational units and take time for the computer to compute

**B.5 Question of Curiosity**

Q.1 Identify the applications of Insertion sort Technique.

* It is useful for sorting arrays whose items are displaced by at most 25
* Sorting “almost sorted” lists. If you know that no element is more than say 30 locations from its final location in sorted order (the data could have been produced from another program)
* Sorting Small Sub-lists in Quick-sort. When the current sub-list is small, it is more efficient to use Insertion-sort

Q.2 Analyze the algorithm for insertion sort – Best case, Worst Case and Average case.

* Average Time Complexity Big-Theta: Θ (n2)
* Best Case Time Complexity Big-Omega: Ω (n)
* Space Complexity: O(1)
* Worst Case Time Complexity Big-O: O(n2)

Q.3 Comment on the performance of Insertion Sort, after filling up the following table.

|  |  |  |  |
| --- | --- | --- | --- |
| **Array Size(n)** | **Data** | **No. of Comparisons** | **No. of Swaps** |
| 4 | 5 3 15 8 | 4 | 2 |
| 8 | 117 210 222 43 251 34 147 47 | 12 | 5 |
| 16 | 122 38 119 37 186 94 179 10 200 252 207 203 99 181 164 83 | 65 | 53 |
| 32 | 155 103 249 86 88 140 69 6 196 156 129 34 222 21 64 200 163 251 117 39 111 119 186 197 236 50 159 237 254 145 231 12 | 270 | 243 |
| 64 | 155 240 251 10 19 139 16 154 36 184 30 106 14 57 134 17 112 37 133 208 42 187 119 52 75 199 60 182 78 207 65 81 25 28 141 174 218 15 226 200 215 45 223 89 70 8 55 237 49 213 1 109 22 90 191 47 51 230 147 58 131 84 145 216 | 1067 | 1010 |
| 128 | 209 70 242 140 43 31 133 103 185 193 220 23 124 197 170 105 89 167 217 126 233 231 134 109 48 252 211 159 254 106 146 153 139 36 225 249 84 45 180 69 196 72 190 78 77 138 192 163 145 229 9 213 161 50 160 122 169 201 28 91 219 212 56 199 203 165 90 135 184 54 95 198 39 83 96 228 92 218 238 74 214 44 18 | 3978 | 3857 |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*