Experiment No- 05 (Group B)

Title:- Python program to compute Insertion Sort and Shell sort algorithm

Objectives:- To understand the use of sorting using algorithm

Problem Statement:-

Write a Python program to store second year percentage of students in array. Write function for sorting array of floating point numbers in ascending order using

a) Insertion sort

b) Shell Sort and display top five scores

Outcomes:- Result of applying Insertion sort algorithm and Shell sort algorithm

Software and Hardware requirements:-

1. **Operating system:** Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10,
2. **RAM-** 2GB RAM (4GB preferable)
3. You have to install **Python3** or higher version

Theory:-

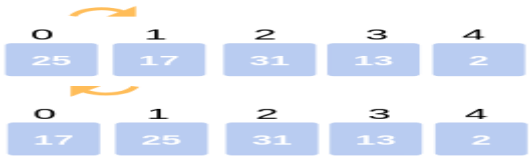
Insertion sort algorithm

1. Consider the first element to be sorted and the rest to be unsorted.
2. Take the first element in the unsorted part(u1) & compare it with sorted part elements(s1).
3. if u1<s1 then insert ul in the correct index, else leave it as it is.
4. Take next element in the unsorted part and
5. Repeat 3 and 4 until all the elements are sorted.

Consider the following array: 25, 17, 31, 13, 2

First Iteration: Compare 25 with 17. The comparison shows 17< 25. Hence swap 17 and 25. The array now looks like:

17, 25, 31, 13, 2



First Iteration

Second Iteration: Begin with the second element (25), but it was already swapped on for the correct position, so we move ahead to the next element.

Now hold on to the third element (31) and compare with the ones preceding it.

Since 31> 25, no swapping takes place.

Also, 31> 17, no swapping takes place and 31 remains at its position.

The array after the Second iteration looks like:

17, 25, 31, 13, 2

Second Iteration

Third Iteration: Start the following Iteration with the fourth element (13), and compare it with its preceding elements.

Since 13< 31, we swap the two.

Array now becomes: **17, 25, 13, 31, 2.**

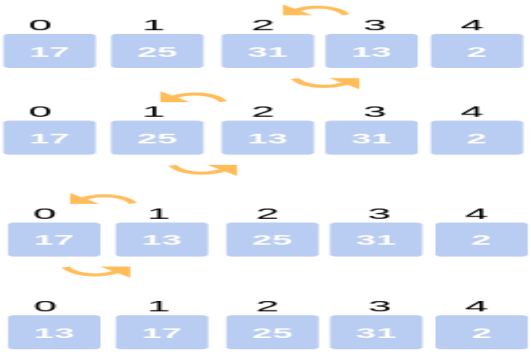
But there still exist elements that we haven't yet compared with 13. Now the comparison takes place between 25 and 13. Since, 13 < 25, we swap the two.

The array becomes **17, 13, 25, 31, 2**.

The last comparison for the iteration is now between 17 and 13. Since 13 < 17, we swap the two.

The array now becomes **13, 17, 25, 31, 2**.

Third Iteration



Fourth Iteration: The last iteration calls for the comparison of the last element (2), with all the preceding elements and make the appropriate swapping between elements.

Since, 2< 31. Swap 2 and 31.

Array now becomes: **13, 17, 25, 2, 31.**

Compare 2 with 25, 17, 13.

Since, 2< 25. Swap 25 and 2.

13, 17, 2, 25, 31.

Compare 2 with 17 and 13.

Since, 2<17. Swap 2 and 17.

Array now becomes:

13, 2, 17, 25, 31.

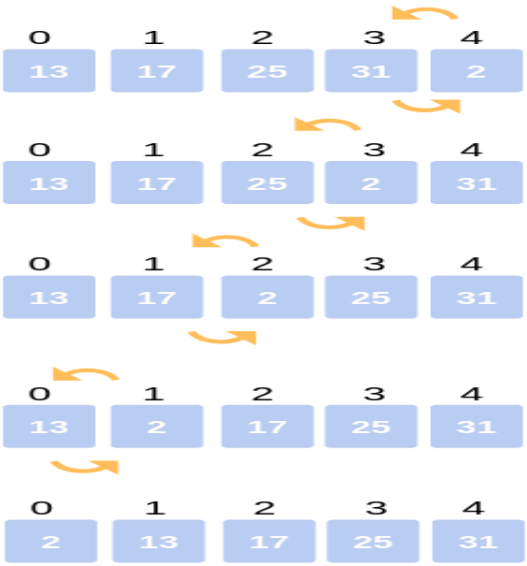
The last comparison for the Iteration is to compare 2 with 13.

Since 2< 13. Swap 2 and 13.

The array now becomes:

2, 13, 17, 25, 31.

This is the final array after all the corresponding iterations and swapping of elements.

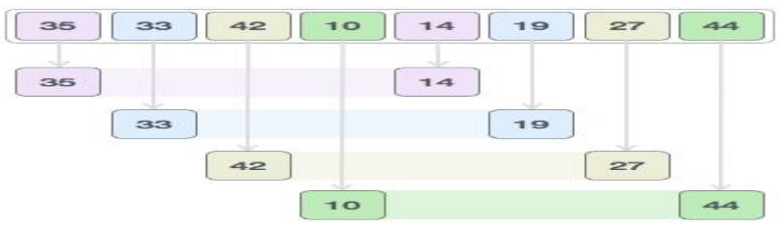


Fourth Iteration

Shell Sort algorithm:-

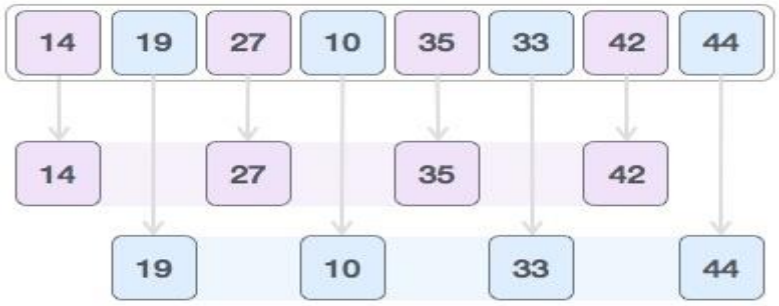
1. Take the list of numbers
2. Find out the gap/incrementor
3. Create the sublist based on gap and sort them using insertion sort algorithm.
4. Reduce gap and repeat step 3.
5. Stop when gap is 0. """

Consider the following array: 35,33,42,10,14,19,27,44

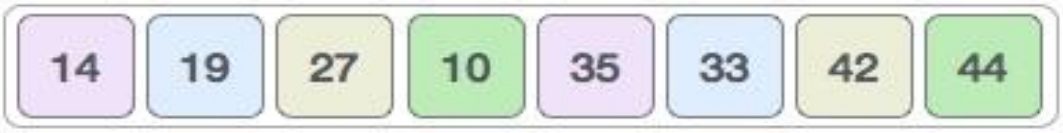


We compare values in each sub-list and swap them (if necessary) in the original array. After this step, the new array should look like this −

Then, we take interval of 1 and this gap generates two sub-lists - {14, 27, 35, 42}, {19, 10, 33, 44}

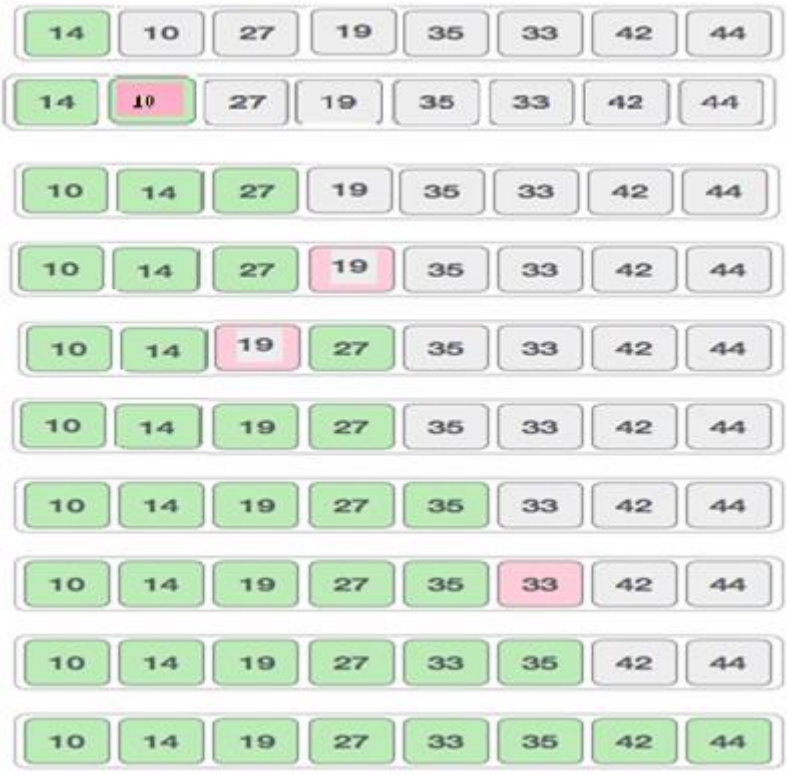


We compare and swap the values, if required, in the original array. After this step, the array should look like this −



Finally, we sort the rest of the array using interval of value 1. Shell sort uses insertion sort to sort the array.

Following is the step-by-step depiction −



We see that it required only four swaps to sort the rest of the array.

Algorithm:-

1) Start

2) Declare the variables such as,n array,item,ch,i=0,flag=1

3) Input how many no. of students in first year

4) Input the Enter percentage marks of students

5) Print Menu:

1) Insertion Sort

2) Shell Sort and display Top 5 scores

3) Exit

ch= Input enter your choice

6) If ch==1 then call function InsertionSort(array,n)

a= input value of a Do you want to continue(y/n)

if a=="y" then go to step 5

else Thank you for using this program!

7) else if ch==2 then call function ShellSort(array,n)

a= input value of a Do you want to continue(y/n)

if a=="y" then go to step 5

else Thank you for using this program!

8) else If ch==3 then go to step 9

else print wrong choice

a=input value of a Do you want to continue(y/n)

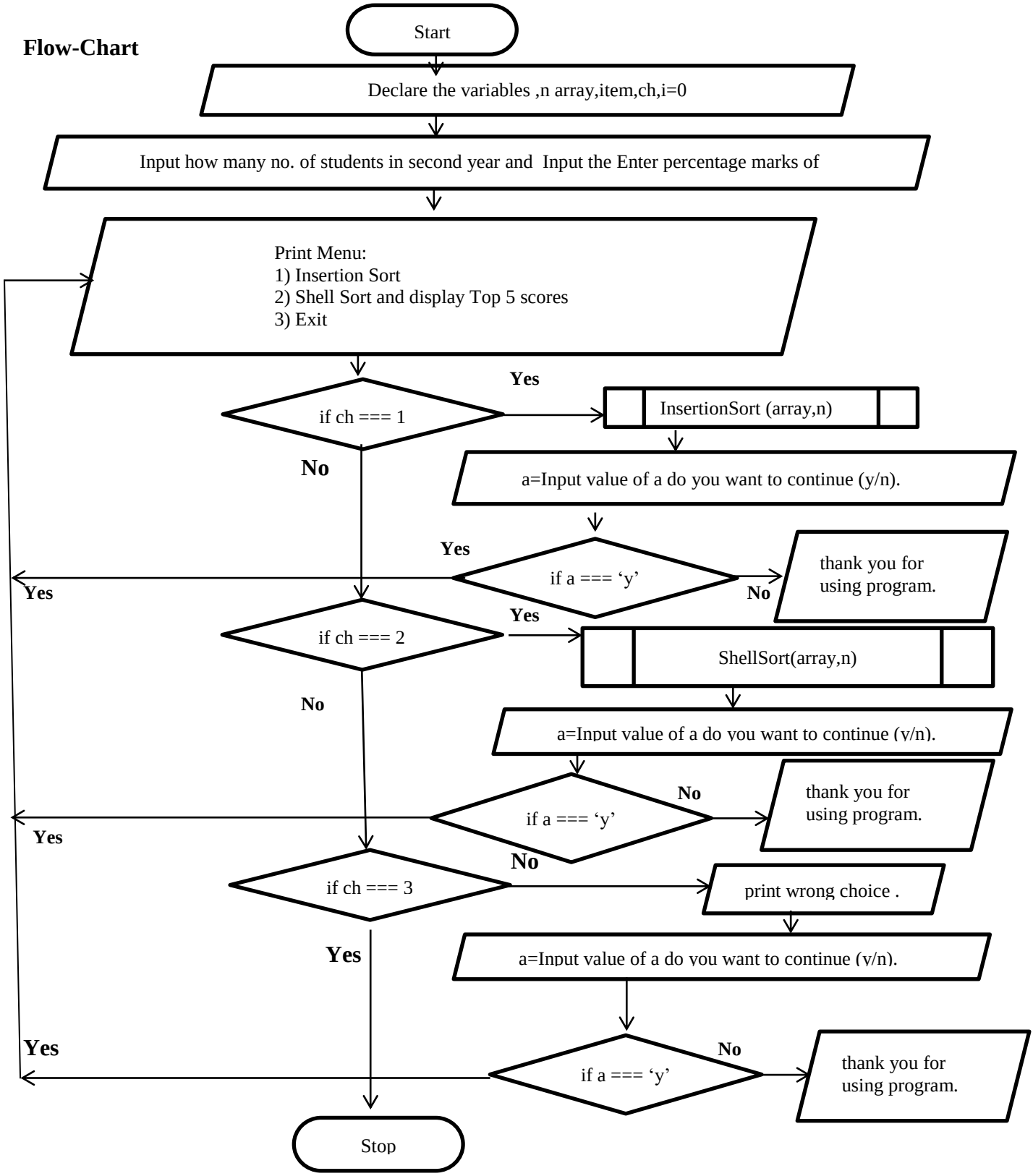
if a=="y" then go to step 5

else

Thank you for using this program!

9) Stop

Flow-Chart



Conclusion:

In this way, we perform Sorting of marks using insertion sort and shell sort algorithm.