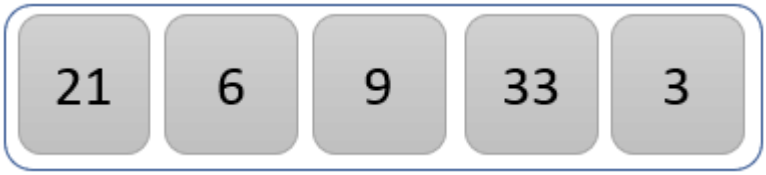
Sorting of list items – Bubble Sorting

- Bubble sort is a sorting algorithm that is used to sort items in a list in ascending order.
- This is done by comparing two adjacent values. If the first value is higher than the second value, then the first value takes the position of the second value while the second value takes the position that was previously occupied by the first value.
- If the first value is lower than the second value, then no swapping is done.
- This process is repeated until all the values in a list have been compared and swapped if necessary. Each iteration is usually called a pass.
- The number of passes in a bubble sort is equal to the number of elements in a list minus one.

Visual Representation

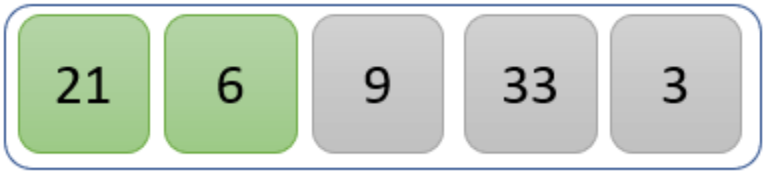
Given a list of five elements, the following images illustrate how the bubble sort iterates through the values when sorting them

The following image shows the unsorted list

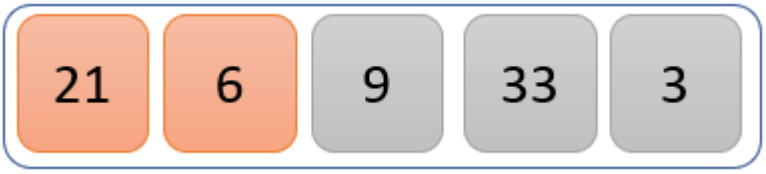


First Iteration

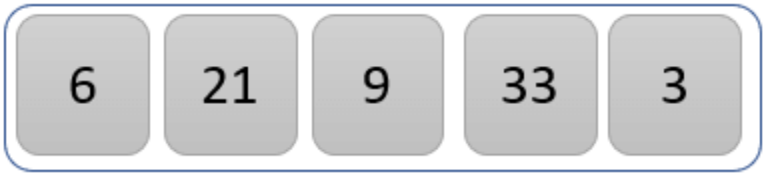
Step 1)



The values 21 and 6 are compared to check which one is greater than the other.

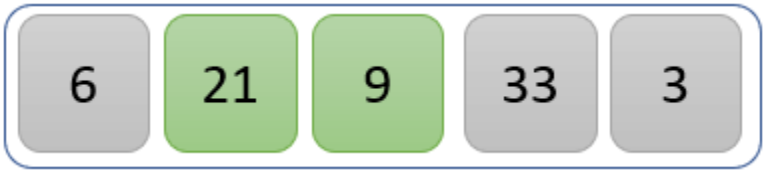


21 is greater than 6, so 21 takes the position occupied by 6 while 6 takes the position that was occupied by 21

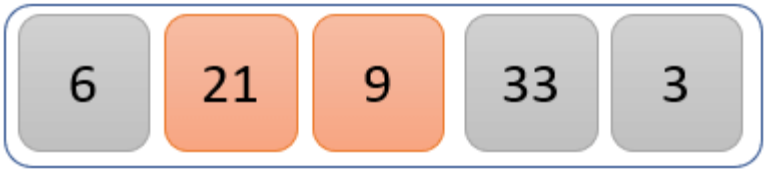


Our modified list now looks like the one above.

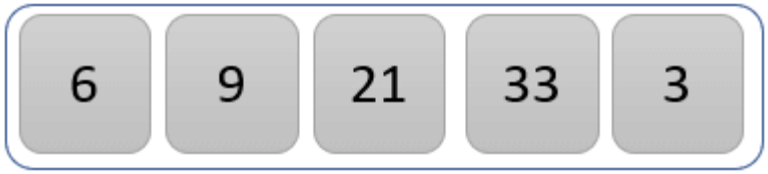
Step 2)



The values 21 and 9 are compared.

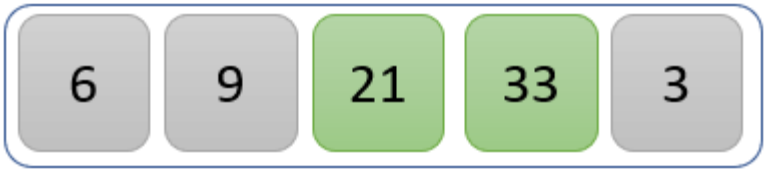


21 is greater than 9, so we swap the positions of 21 and 9

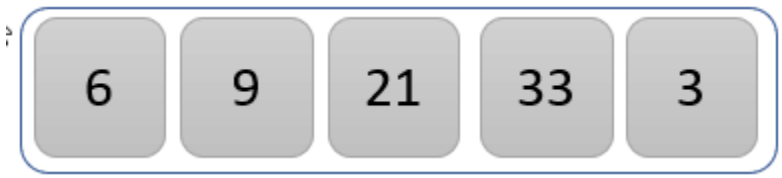


The new list is now as above

Step 3)

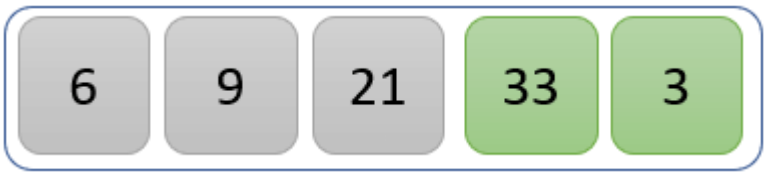


The values 21 and 33 are compared to find the greater one.

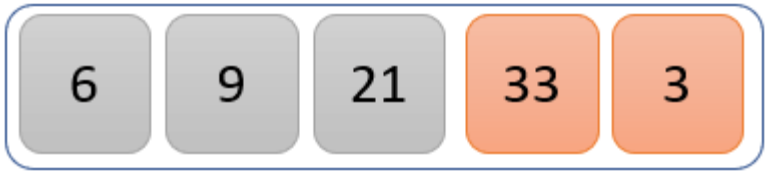


The value 33 is greater than 21, so no swapping takes place.

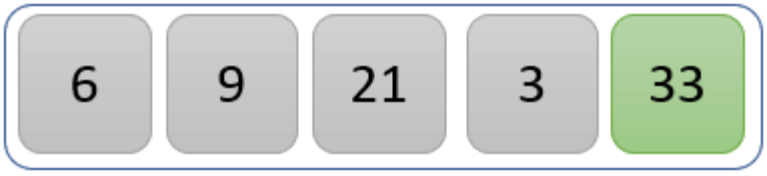
Step 4)



The values 33 and 3 are compared to find the greater one.



The value 33 is greater than 3, so we swap their positions.



The sorted list at the end of the first iteration is like the one above

Second Iteration

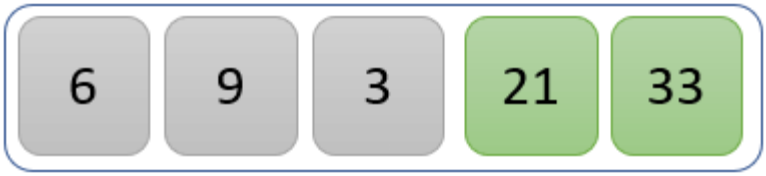
Step 1

Step 2

Step 3

Step 4

The new list after the second iteration is as follows



Third Iteration

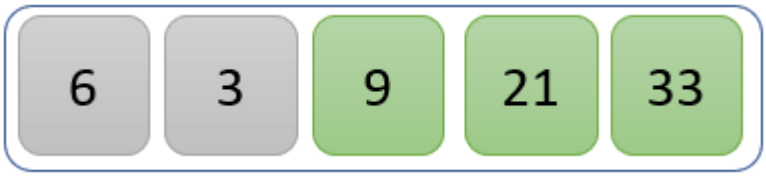
Step 1

Step 2

Step 3

Step 4

The new list after the third iteration is as follows



Fourth Iteration

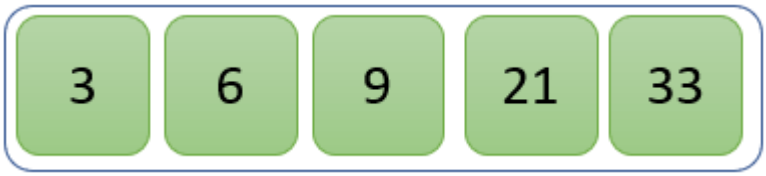
Step 1

Step 2

Step 3

Step 4

The new list after the fourth iteration is as follows



Example: Bubble Sorting – Python Code

```
print("\n*****************************************\n")

nlist = [100,-2,2,0,21,34,66,8,1]
print("\nFirst list of numbers - Before Sorting")
print(nlist)
for i in range(8):
        for j in range(8):
            #print("-Inner loop: Iteration-", j)
            if nlist[j]>nlist[j+1]:
                temp = nlist[j]
                nlist[j] = nlist[j+1]
                nlist[j+1] = temp
print("\nFirst list of numbers - After Sorting")
print(nlist)


print("\n*****************************************\n")

nlist = [14,46,43,27,57,41,45,21,70]
print("\nSecond list of numbers - Before Sorting")
print(nlist)
for i in range(8):
        for j in range(8):
            #print("-Inner loop: Iteration-", j)
            if nlist[j]>nlist[j+1]:
                temp = nlist[j]
                nlist[j] = nlist[j+1]
                nlist[j+1] = temp
print("\nSecond list of numbers - After Sorting")
print(nlist)
```

```
(base) F:\CSE1001\Python-Programs>python prgm24-BubbleSorting.py

*************************************


First list of numbers - Before Sorting
[100, -2, 2, 0, 21, 34, 66, 8, 1]

First list of numbers - After Sorting
[-2, 0, 1, 2, 8, 21, 34, 66, 100]

*************************************


Second list of numbers - Before Sorting
[14, 46, 43, 27, 57, 41, 45, 21, 70]

Second list of numbers - After Sorting
[14, 21, 27, 41, 43, 45, 46, 57, 70]

*************************************
```

Output – Showing number of iterations for the above program

```
Second list of numbers - Before Sorting
[14, 46, 43, 27, 57, 41, 45, 21, 70]
***outer loop: Iteration*** 0
-Inner loop: Iteration- 0
-Inner loop: Iteration- 1
-Inner loop: Iteration- 2
-Inner loop: Iteration- 3
-Inner loop: Iteration- 4
-Inner loop: Iteration- 5
-Inner loop: Iteration- 6
-Inner loop: Iteration- 7
***outer loop: Iteration*** 1
-Inner loop: Iteration- 0
-Inner loop: Iteration- 1
-Inner loop: Iteration- 2
-Inner loop: Iteration- 3
-Inner loop: Iteration- 4
-Inner loop: Iteration- 5
-Inner loop: Iteration- 6
-Inner loop: Iteration- 7
***outer loop: Iteration*** 2
-Inner loop: Iteration- 0
-Inner loop: Iteration- 1
-Inner loop: Iteration- 2
-Inner loop: Iteration- 3
-Inner loop: Iteration- 4
-Inner loop: Iteration- 5
-Inner loop: Iteration- 6
-Inner loop: Iteration- 7
```

```
***outer loop: Iteration*** 3
-Inner loop: Iteration- 0
-Inner loop: Iteration- 1
-Inner loop: Iteration- 2
-Inner loop: Iteration- 3
-Inner loop: Iteration- 4
-Inner loop: Iteration- 5
-Inner loop: Iteration- 6
-Inner loop: Iteration- 7
***outer loop: Iteration*** 4
-Inner loop: Iteration- 0
-Inner loop: Iteration- 1
-Inner loop: Iteration- 2
-Inner loop: Iteration- 3
-Inner loop: Iteration- 4
-Inner loop: Iteration- 5
-Inner loop: Iteration- 6
-Inner loop: Iteration- 7
***outer loop: Iteration*** 5
-Inner loop: Iteration- 0
-Inner loop: Iteration- 1
-Inner loop: Iteration- 2
-Inner loop: Iteration- 3
-Inner loop: Iteration- 4
-Inner loop: Iteration- 5
-Inner loop: Iteration- 6
-Inner loop: Iteration- 7
```

```
***outer loop: Iteration*** 6
-Inner loop: Iteration- 0
-Inner loop: Iteration- 1
-Inner loop: Iteration- 2
-Inner loop: Iteration- 3
-Inner loop: Iteration- 4
-Inner loop: Iteration- 5
-Inner loop: Iteration- 6
-Inner loop: Iteration- 7
***outer loop: Iteration*** 7
-Inner loop: Iteration- 0
-Inner loop: Iteration- 1
-Inner loop: Iteration- 2
-Inner loop: Iteration- 3
-Inner loop: Iteration- 4
-Inner loop: Iteration- 5
-Inner loop: Iteration- 6
-Inner loop: Iteration- 7
```

Activities

1. Write an algorithm for bubble sorting technique.
2. Draw a flow chart for bubble sorting technique.
3. Write a python program for the following requirements.
    a. Read ‘n’ numbers from users and insert them into a list (say **list_numbers**). Make sure that **list_numbers** contains negative, positive, odd, even numbers.
    b. Print the **list_numbers**.
    c. Using **bubble sorting technique**, sort the numbers in **list_numbers** and print them in ascending order. Also store the numbers in ascending order into a file (say ‘ascending_numbers.txt’).
    d. Store the numbers in descending order into a file (say ‘descending_numbers.txt’).
    e. Take the even numbers from **list_numbers** and store them into a file (say ‘even_numbers.txt’).
    f. Take the odd numbers from **list_numbers** and store them into a file (say ‘odd_numbers.txt’).