

Practice Problems for Python (T2) (Chapter 3,4 and 5)

1. Write a Python program to count the number of strings where the string length is 2 or more and the first and last character are same from a given list of strings. Example:

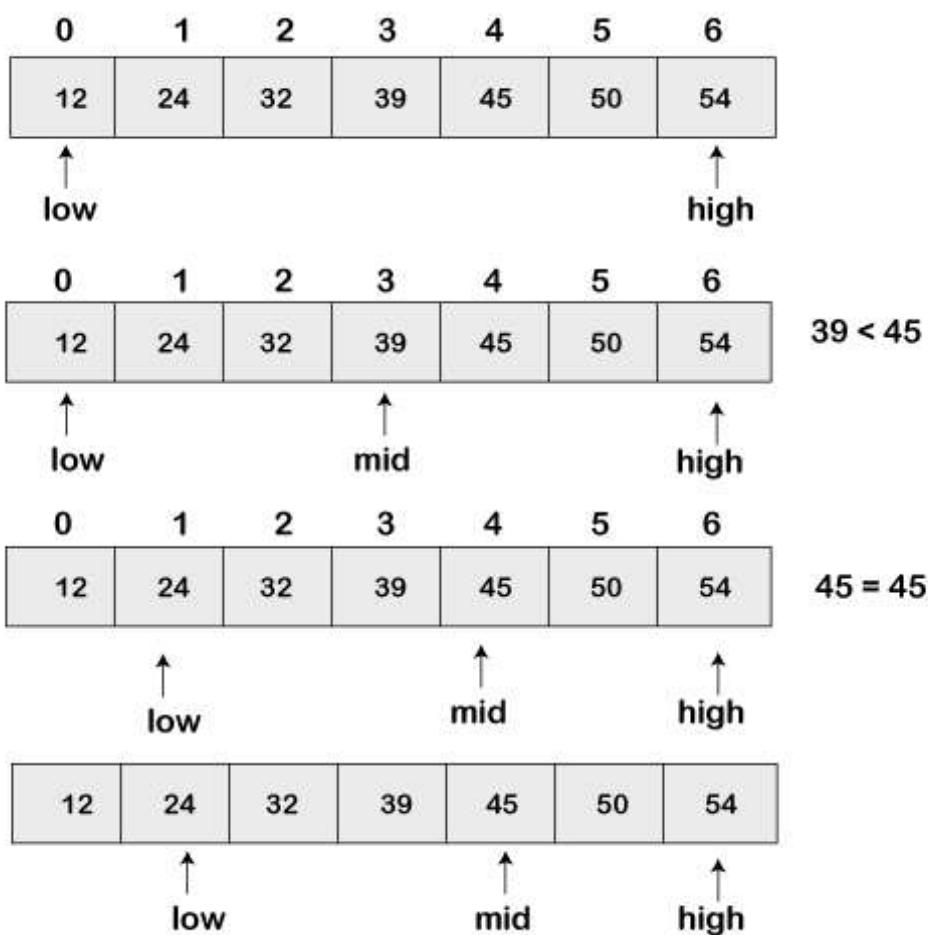
Input: ['abc', 'xyz', 'aba', '1221']

Output: 2

2. Write a Python program for binary search.

Binary Search : In computer science, a binary search or half-interval search algorithm finds the position of a target value within a sorted array. The binary search algorithm can be classified as a dichotomies divide-and-conquer search algorithm and executes in logarithmic time.

Step by step example :



Output: Element is present at index 3

3. Write a Python program to find the indices of all occurrences of target in the uneven matrix.

An irregular/uneven matrix, or ragged matrix, is a matrix that has a different number of elements in each row. Ragged matrices are not used in linear algebra, since standard matrix transformations cannot be performed on them, but they are useful as arrays in computing.

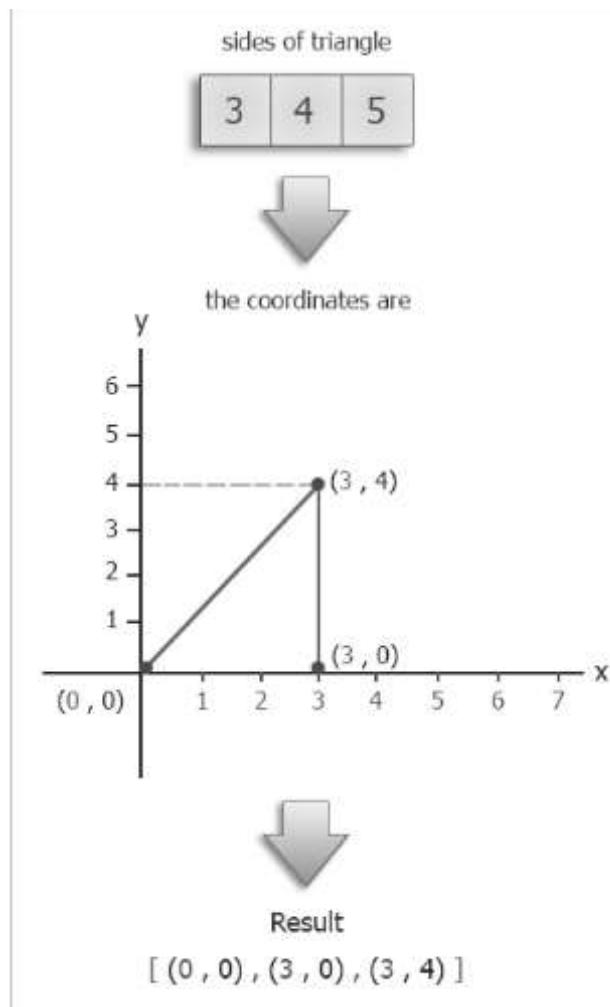
Input: `[[1, 3, 2, 32, 19], [19, 2, 48, 19], [], [9, 35, 4], [3, 19]], 19]`

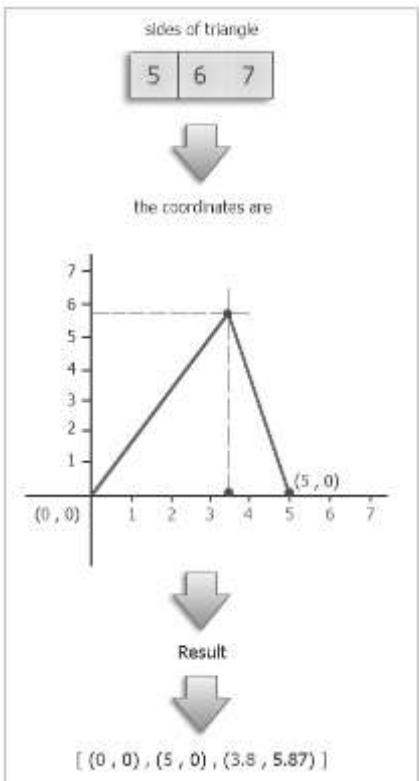
Output: `[[0, 4], [1, 0], [1, 3], [4, 1]]`

Input: `[[[1, 2, 3, 2], [], [7, 9, 2, 1, 4]], 2]`

Output: `[[0, 1], [0, 3], [2, 2]]`

4. Write a Python program to find the coordinates of a triangle with the given side lengths.





Heron's formula to find area $(s * (s - a) * (s - b) * (s - c))^{**} 0.5$ where s=semi perimeter a,b,c are the lengths of sides

Input: [3, 4, 5]

Output: [[0.0, 0.0], [3, 0.0], [3.0, 4.0]]

Input: [5, 6, 7]

Output: [[0.0, 0.0], [5, 0.0], [3.8, 5.878775382679628]]

5. Write a Python program to rescale and shift numbers of a given list, so that they cover the range [0, 1].

List of numbers

18.5	17.0	18.0	19.0	18.0
------	------	------	------	------



find max and min value



18.5	17.0	18.0	19.0	18.0
------	------	------	------	------

min

max



find scaled value of each numbers

Formula →
$$\frac{\text{number} - \text{number}_{\min}}{\text{number}_{\max} - \text{number}_{\min}}$$



$$1^{\text{st}} \text{ number } (18.5 - 17.0) / (19.0 - 17.0) = .75$$

$$2^{\text{nd}} \text{ number } (17.0 - 17.0) / (19.0 - 17.0) = 0.0$$

$$3^{\text{rd}} \text{ number } (18.0 - 17.0) / (19.0 - 17.0) = 0.5$$

$$4^{\text{th}} \text{ number } (19.0 - 17.0) / (19.0 - 17.0) = 1.0$$

$$5^{\text{th}} \text{ number } (18.0 - 17.0) / (19.0 - 17.0) = 0.5$$

after rescaling the values

.75	0.0	0.5	1.0	0.5
-----	-----	-----	-----	-----

Input: [18.5, 17.0, 18.0, 19.0, 18.0] Output: [0.75, 0.0, 0.5, 1.0, 0.5]

Input: [13.0, 17.0, 17.0, 15.5, 2.94] Output: [0.7155049786628734, 1.0, 1.0, 0.8933143669985776, 0.0]

6. Write a Python program to find the numbers that are greater than 10 and have odd first and last digits.

Input: [1, 3, 79, 10, 4, 1, 39, 62]

Output: [79, 39]

Input: [11, 31, 77, 93, 48, 1, 57]

Output: [11, 31, 77, 93, 57]

7. Write a Python program to shift the decimal digits n places to the left, wrapping the extra digits around. If shift > the number of digits of n, reverse the string.

Input: n = 12345 and shift = 1

Output: Result = 23451

Input: n = 12345 and shift = 2

Output: Result = 34512

Input: n = 12345 and shift = 3

Output: Result = 45123

Input: n = 12345 and shift = 5

Output: Result = 12345

Input: n = 12345 and shift = 6

Output: Result = 54321

8. write a Python program to create a list containing that number in between each pair of adjacent numbers.

Given a list of numbers and a number to inject,

Input: [12, -7, 3, -89, 14, 88, -78, -1, 2, 7] Separator: 6

Output: [12, 6, -7, 6, 3, 6, -89, 6, 14, 6, 88, 6, -78, 6, -1, 6, 2, 6, 7]

Input: [1, 2, 3, 4, 5, 6] Separator: 9

Output: [1, 9, 2, 9, 3, 9, 4, 9, 5, 9, 6]

9. Write a Python program to find two indices making a given string unhappy.

A string is happy if every three consecutive characters are distinct.

Input: Python

Output: None

Input: Unhappy

Output: [4, 5]

Input: Find

Output: None

Input: Street

Output: [3, 4]

10. Write a python program to start with a list of integers, keep every other element in place and otherwise sort the list.

Input: [2, 5, 6, 3, 1, 4, 34]

Output: [1, 5, 2, 3, 6, 4, 34]

Input: [8, 0, 7, 2, 9, 4, 1, 2, 8, 3]

Output: [1, 0, 7, 2, 8, 4, 8, 2, 9, 3]

11. Write a python program to get the single digits in numbers sorted backwards and converted to English words.

Input: [1, 3, 4, 5, 11]

Output: ['five', 'four', 'three', 'one']

Input: [27, 3, 8, 5, 1, 31]

Output: ['eight', 'five', 'three', 'one']

12. Write a python program to find the following strange sort of list of numbers: the first element is the smallest, the second is the largest of the remaining, the third is the smallest of the

remaining, the fourth is the smallest of the remaining, etc.

Input: [1, 3, 4, 5, 11]

Output: [1, 11, 3, 5, 4]

Input: [27, 3, 8, 5, 1, 31]

Output: [1, 31, 3, 27, 5, 8]

Input: [1, 2, 7, 3, 4, 5, 6] Output: [1, 7, 2, 6, 3, 5, 4]

13. Write a python program to find four positive even integers whose sum is a given integer.

Input: n = 100

Output: [94, 2, 2, 2]

Input: n = 1000

Output: [994, 2, 2, 2]

Input: n = 10000

Output: [9994, 2, 2, 2]

Input: n = 1234567890

Output: [1234567884, 2, 2, 2]

14. Write a python program to implement linear search.

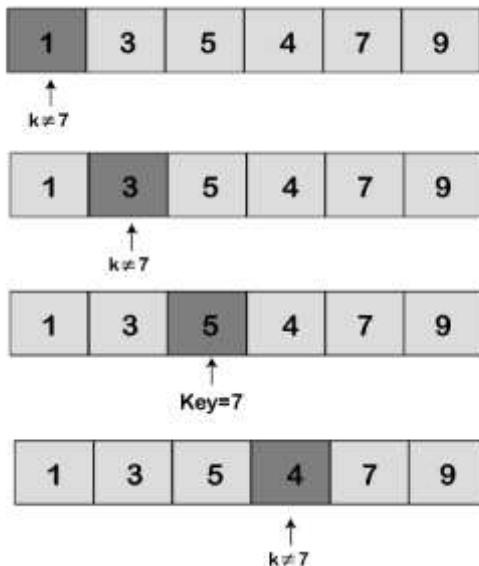
Linear search is a method of finding elements within a list. It is also called a sequential search. It is the simplest searching algorithm because it searches the desired element in a sequential manner. It compares each and every element with the value that we are searching for. If both are matched, the element is found, and the algorithm returns the key's index position.

Step - 1: Start the search from the first element and Check key = 7 with each element of list x.

1	3	5	4	7	9
---	---	---	---	---	---

List to be Searched for

Step - 2: If element is found, return the index position of the key.



Step - 3: If element is not found, return element is not present.

1	3	5	4	7	9
Key=7					

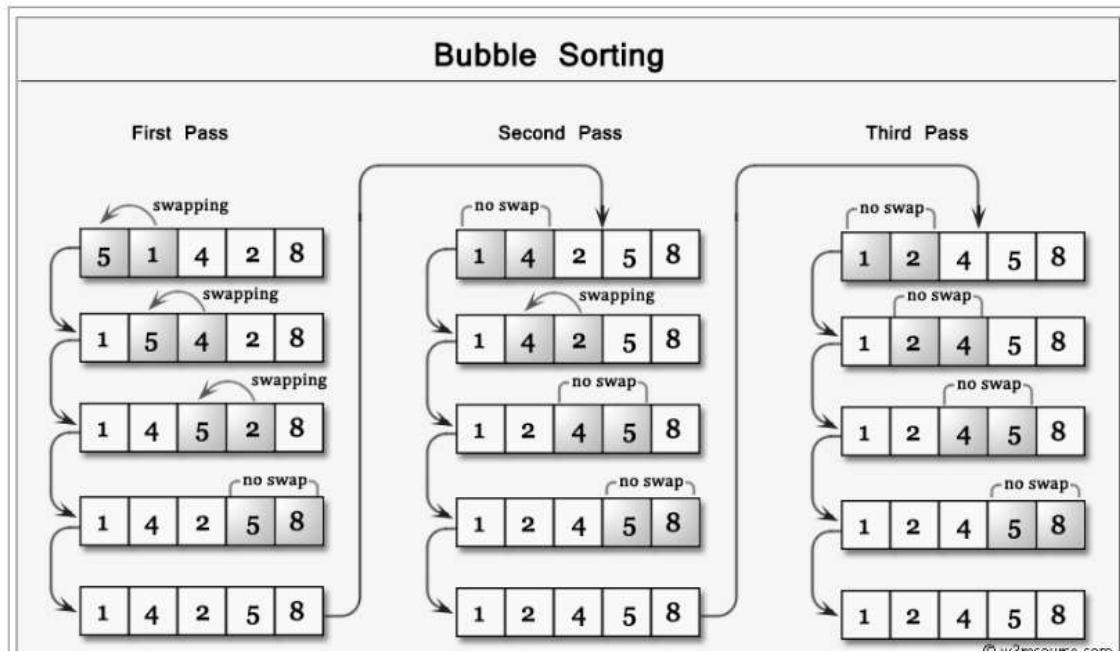
Input: Enter the list of numbers: 5 4 3 2 1 10 11 2 The number to search for: 1

Output: 1 was found at index 4.

15. Write a Python program to sort a list of elements using the bubble sort algorithm.

Note : According to Wikipedia "Bubble sort, sometimes referred to as sinking sort, is a simple sorting algorithm that repeatedly steps through the list to be sorted, compares each pair of adjacent items and swaps them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted. The algorithm, which is a comparison sort, is named for the way smaller elements "bubble" to the top of the list. Although the algorithm is simple, it is too slow and impractical for most problems even when compared to insertion sort. It can be practical if the input is usually in sort order but may occasionally have some out-of-order elements nearly in position.

Step by step pictorial presentation :

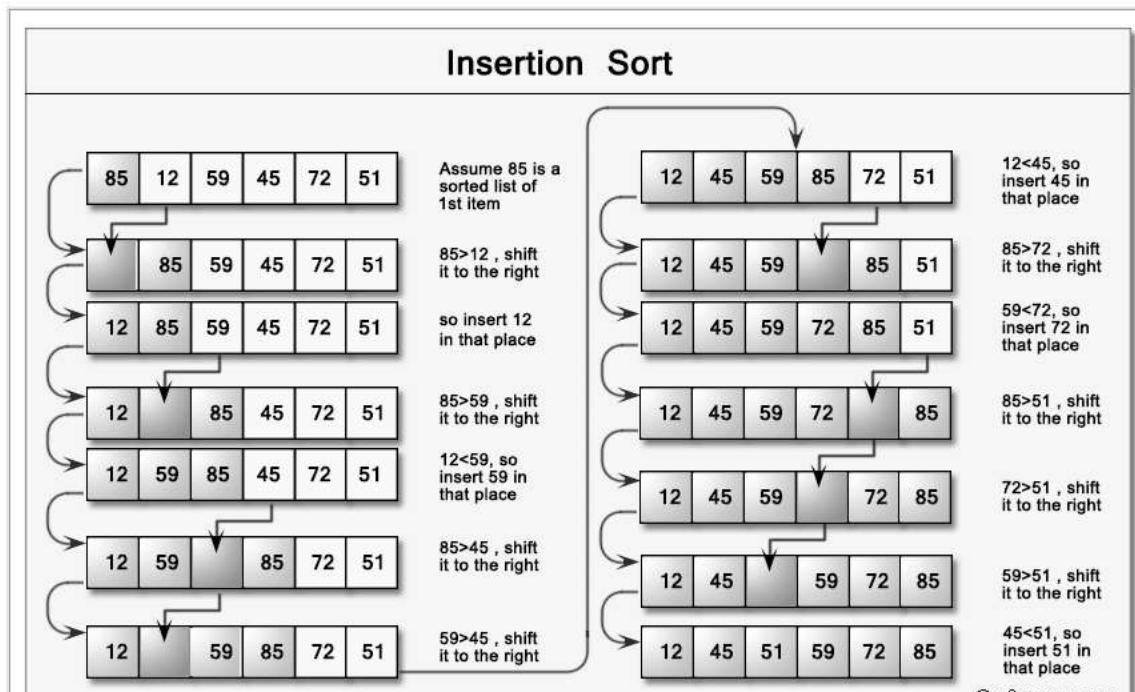


Input:nlist = [14,46,43,27,57,41,45,21,70]

Output: [14, 21, 27, 41, 43, 45, 46, 57, 70]

16. Write a Python program to sort a list of elements using the insertion sort algorithm.

Note: According to Wikipedia "Insertion sort is a simple sorting algorithm that builds the final sorted array (or list) one item at a time. It is much less efficient on large lists than more advanced algorithms such as quicksort, heapsort, or merge sort."



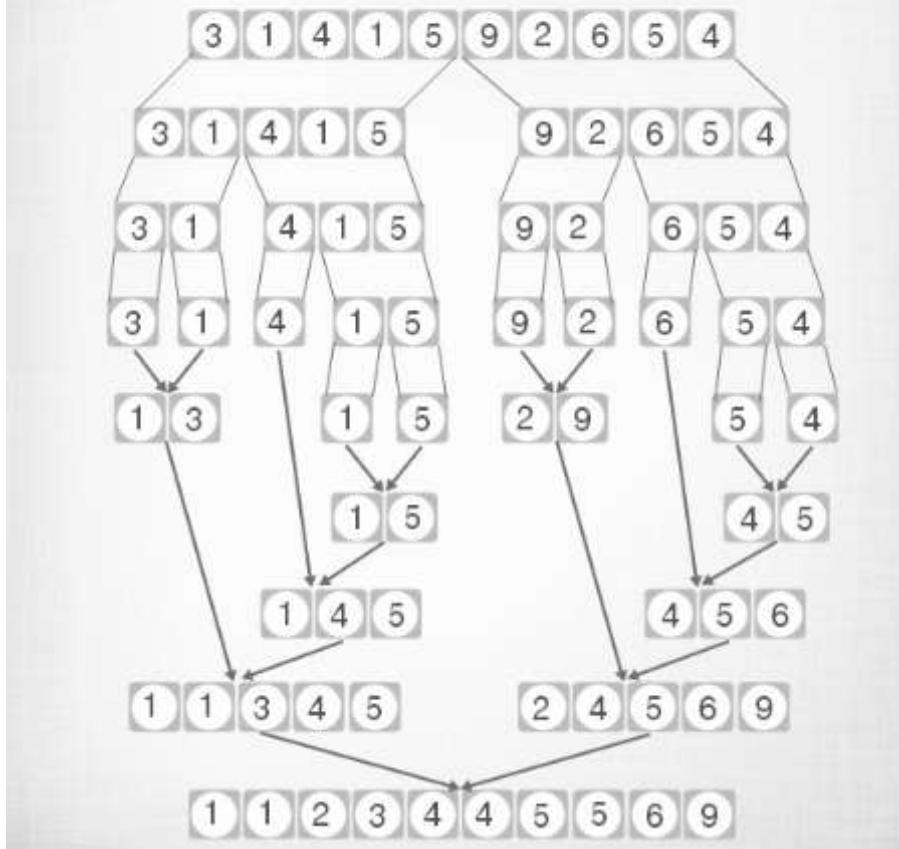
Input: nlist = [14,46,43,27,57,41,45,21,70]

Output: [14, 21, 27, 41, 43, 45, 46, 57, 70]

17. Write a Python program to sort a list of elements using the merge sort algorithm.

Note: According to Wikipedia "Merge sort (also commonly spelled mergesort) is an O (n log n) comparison-based sorting algorithm. Most implementations produce a stable sort, which means that the implementation preserves the input order of equal elements in the sorted output." Algorithm: Conceptually, a merge sort works as follows : Divide the unsorted list into n sublists, each containing 1 element (a list of 1 element is considered sorted).Repeatedly merge sublists to produce new sorted sublists until there is only 1 sublist remaining. This will be the sorted list.

MERGE SORTING ON 3.1.4.1.5.9.2.6.5.4



Input: nlist = [14,46,43,27,57,41,45,21,70]

Output:

Splitting [14, 46, 43, 27, 57, 41, 45, 21, 70] Splitting [14, 46, 43, 27] Splitting [14, 46] Splitting [14] Merging [14] Splitting [46] Merging [46] Merging [14, 46] Splitting [43, 27] Splitting [43] Merging [43] Splitting [27] Merging [27] Merging [27, 43] Merging [14, 27, 43, 46] Splitting [57, 41, 45, 21, 70] Splitting [57, 41] Splitting [57] Merging [57] Splitting [41] Merging [41] Merging [41, 57] Splitting [45, 21, 70] Splitting [45] Merging [45] Splitting [21, 70] Splitting [21] Merging [21] Splitting [70] Merging [70] Merging [21, 70] Merging [21, 45, 70] Merging [21, 41, 45, 57, 70] Merging [14, 21, 27, 41, 43, 45, 46, 57, 70] [14, 21, 27, 41, 43, 45, 46, 57, 70]

18. Write a python program for backtracking example

Backtracking is a form of recursion. But it involves choosing only option out of any possibilities. We begin by choosing an option and backtrack from it, if we reach a state where we conclude that this specific option does not give the required solution. We repeat these steps by going across each available option until we get the desired solution. Below is an example of finding all possible order of arrangements of a given set of letters. When we choose a pair we apply backtracking to verify if that exact pair has already been created or not. If not already created, the pair is added to the answer list else it is ignored.

Input: `permute(1, ["a","b","c"])`

Output: `['a', 'b', 'c']`

Input: `permute(2, ["a","b","c"])`

Output: `['aa', 'ab', 'ac', 'ba', 'bb', 'bc', 'ca', 'cb', 'cc']`

19. Write a python program for Tower of Hanoi program in python

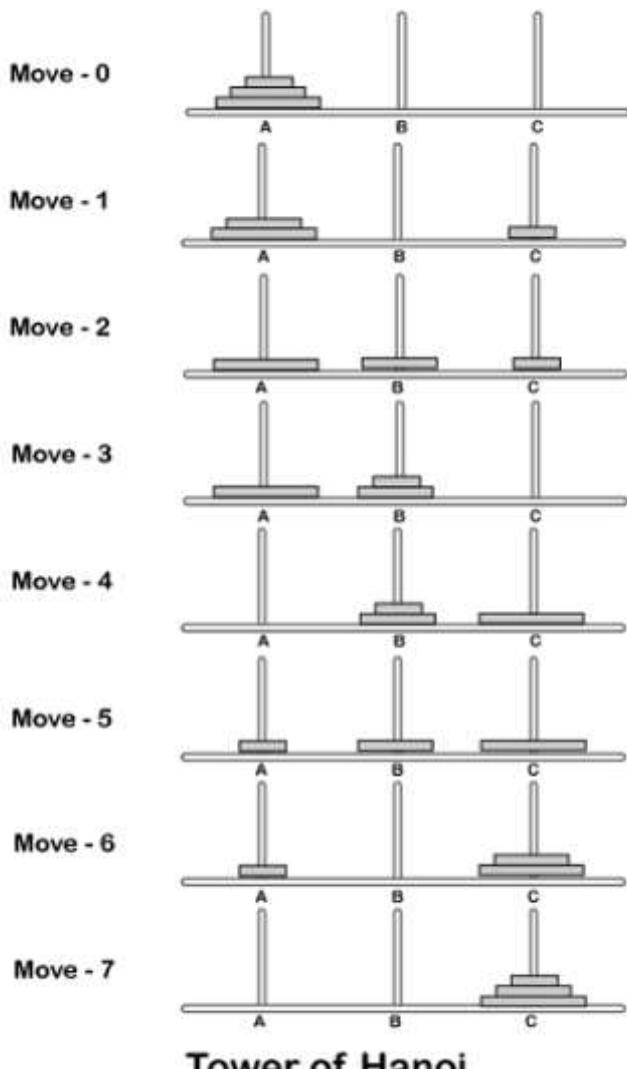
The program prompts the user for the number of disks n and the program prints the procedure to move n disks from peg A to peg C using peg B.

The rules of "Tower of Hanoi" are quite simple, but the solution is slightly hard. There are three rods. The disks are stacked in the descending order; the largest disk stacked at the bottom and the smallest one on top.

The task is to transfer the disks from one source rod to another target rod.

The following rule must not be violated

Only one disk can be moved at a time. The most upper disk from one of the rod can be stimulated in move. The smaller disk cannot be placed at the lower of the largest disk. The number of moves can be calculated as $2n - 1$.



Tower of Hanoi

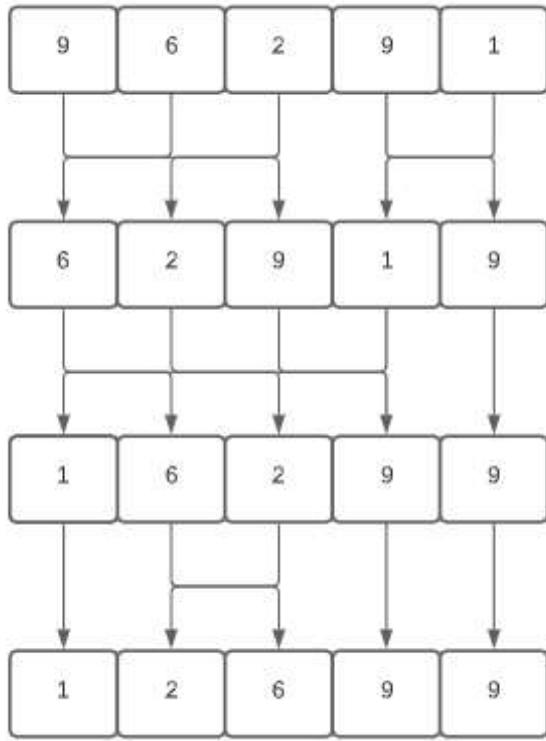
Input: Enter number of disks: 4

Output: Move disk 1 from peg A to peg B. Move disk 2 from peg A to peg C. Move disk 1 from peg B to peg C. Move disk 3 from peg A to peg B. Move disk 1 from peg C to peg A. Move disk 2 from peg C to peg B. Move disk 1 from peg A to peg B. Move disk 4 from peg A to peg C. Move disk 1 from peg B to peg C. Move disk 2 from peg B to peg A. Move disk 1 from peg C to peg A. Move disk 3 from peg B to peg C. Move disk 1 from peg A to peg B. Move disk 2 from peg A to peg C. Move disk 1 from peg B to peg C.

20. Write a Python Program to Implement CockTail Sort

Firstly array is traversed from left to right. During traversal, adjacent items are compared and based on the conditions, the values are swapped. By this, the largest number will be at the end of the array.

Now the array is traversed in the opposite direction and based on the conditions, elements are swapped. By this, the smallest number will be at the start.

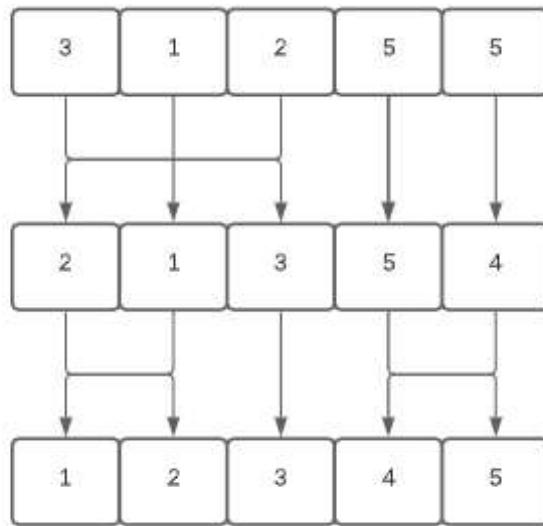


Input: Enter the list of numbers: 3 18 5 2 10 0 7 4

Output: Sorted list: [0, 2, 3, 4, 5, 7, 10, 18]

21. Write a python program to implement Comb sort.

The Comb Sort is a variant of the Bubble Sort. Like the Shell sort, the Comb Sort increases the gap used in comparisons and exchanges. Some implementations use the insertion sort once the gap is less than a certain amount. The basic idea is to eliminate turtles, or small values near the end of the list, since in a bubble sort these slow the sorting down tremendously. Rabbits, large values around the beginning of the list do not pose a problem in bubble sort. In bubble sort, when any two elements are compared, they always have a gap of 1. The basic idea of comb sort is that the gap can be much more than 1.



Input: Enter the list of numbers: 2 8 4 3 7 10 23 4 5

Output: Sorted list: [2, 3, 4, 4, 5, 7, 8, 10, 23]

22. Write a python program to implement bucket sort

According to Wikipedia "Bubble sort, sometimes referred to as sinking sort, is a simple sorting algorithm that repeatedly steps through the list to be sorted, compares each pair of adjacent items and swaps them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted. The algorithm, which is a comparison sort, is named for the way smaller elements "bubble" to the top of the list. Although the algorithm is simple, it is too slow and impractical for most problems even when compared to insertion sort. It can be practical if the input is usually in sort order but may occasionally have some out-of-order elements nearly in position."

Input: Enter the list of (nonnegative) numbers: 2 1 5 10 3 5 7

Output: Sorted list: [1, 2, 3, 5, 5, 7, 10]

23. Write a python program to Minimize Lateness using Greedy Algorithm

We are given n requests numbered 0 to $n - 1$. Each request i has a time that it takes to complete $t(i)$ and a deadline $d(i)$. If a request i starts at time $s(i)$, then its finish time is $f(i) = s(i) + t(i)$. The lateness of request i is $L(i) = f(i) - d(i)$ if $f(i) > d(i)$. The maximum value of $L(i)$ over all i is called the maximum lateness. That is, maximum lateness = $\max(L(0), L(1), \dots, L(n - 1))$. The problem is to schedule all the requests such that the value of maximum lateness is minimized.

Problem Solution

1. The function minimize_lateness defined.

2. It takes two lists ttimes and dtimes as arguments.
3. ttimes[i] is the time taken to complete request i.
4. dtimes[i] is the deadline of request i.
5. The algorithm works by sorting the requests in order of earliest deadline.
6. This ordering is an optimal way to schedule the requests to minimize maximum lateness.
7. The minimum maximum lateness is then calculated.
8. The minimum maximum lateness and the optimal schedule ordering is returned.

Input: Enter number of requests: 5 Enter the time taken to complete the 5 request(s) in order: 2 4 1 3 5 Enter the deadlines of the 5 request(s) in order: 3 1 5 6 3

Output: The minimum maximum lateness: 9 The order in which the requests should be scheduled: [1, 0, 4, 2, 3]

24. Write a python program to find the starting number, under ten thousand will create the longest chain.

From Wikipedia - The Collatz conjecture is a conjecture in mathematics that concerns a sequence defined as follows: start with any positive integer n . Then each term is obtained from the previous term as follows: if the previous term is even, the next term is one half the previous term. If the previous term is odd, the next term is 3 times the previous term plus 1. The conjecture is that no matter what value of n , the sequence will always reach 1.

Output: 6171

25. Write a python program to find three numbers from an array such that the sum of three numbers equal to zero.

Input: nums1=[-1,0,1,2,-1,-4]

Output: [[-1, -1, 2], [-1, 0, 1]]

Input: nums2 = [-25,-10,-7,-3,2,4,8,10]

Output: [[-10, 2, 8], [-7, -3, 10]]

26. Write a python program to check a sequence of numbers is a geometric progression or not.

In mathematics, a geometric progression or geometric sequence is a sequence of numbers where each term after the first is found by multiplying the previous one by a fixed, non-zero number called the common ratio. For example, the sequence 2, 6, 18, 54, ... is a geometric progression with common ratio 3. Similarly, 10, 5, 2.5, 1.25, ... is a geometric sequence with common ratio 1/2.

Input: [2, 6, 18, 54]

Output: True

Input: [5, 8, 9, 11]

Output: False