1. Prime Number Finder

Create a prime\_finder() function that takes in a number, n, and returns all the prime numbers from 1 to n (inclusive). As a reminder, a prime number is a number that is only divisible by 1 and itself.

For example, prime\_finder(11) should return [2, 3, 5, 7, 11].

1. Find the Xth Number In Order

Write a function, getX, that given an integer x and a list nums returns the Xth number if the list was in sorted order. In other words, the Xth smallest number.

Function Name: getX

Input: An integer, x, and an unsorted list of integers nums that aren’t necessarily distinct

Output: The integer corresponding to the Xth number in the sorted list

Example:

getX(2, [5, 10, -3, -3, 7, 9]) => -3

The second number in order is -3.

getX(4, [5, 10, -3 , -3, 7, 9]) => 7

The fourth number in order is 7.

Note that this assumes the first number is position 1, not 0. If the input x is greater than the length of the list, or nums is empty, return 0.

1. Incorporate Binary Search method to find the index position of an element in a list (already sorted in ascending order).

Lst1 = [1, 6, 13, 24, 25, 26, 29, 31]

search = 26

Required output = Element 26 is at index position 5

**Binary search method** divides the input array(or list) based on whether the number to be searched is higher or lower than the value of the middle element of input. If number to search is higher than mid value, the only elements to the right of mid value are taken as new input. If number to search is lower than mid value then only elements of left side are taken as new input. Iteratively we get smaller and smaller inputs until either the element to be searched falls exactly on a midpoint or the input is only left with one element.

For e.g. – above lst1, search = 26.

Mid value = 24 – *For even numbers midvalue is taken just before midpoint i.e. here midpoint is between 24 & 25 so 24 is taken as midpoint. If it was an odd number and there was a number exactly at midpoint then that number would be taken as mid value.*

Since 26 is higher than 24 all elements to right of 24 are taken i.e. my input now has only 4 elements i.e

New\_input = [25, 26, 29, 31]

Midvalue = 26

Element to be searched = 26

Both are equal – so index position of 26 should be output.

If the element to search would have been 25 – then the input list would have been divided further.

Note : Write your program such that if the element is not found – it should print ‘Element not found’

1. Calculate the Mean and Mode

Create a stats\_finder() function that takes in a list of numbers and returns a list containing the mean and mode, in that order. As a reminder, the mean is the average of the values and the mode is the most occurring value. If there are multiple modes, return the mode with the lowest value. Make sure that you write your functions and find these answers from scratch – don’t use imported tools!

For example, stats\_finder([500, 400, 400, 375, 300, 350, 325, 300]) should return [368.75, 300].

1. Top Score Sorter

Write a function, score\_sorter(), that will take a list of unsorted scores along with the highest possible score. It should return a sorted list of all of the scores, in descending order from high to low. You are expected to create and implement your own sorting algorithm for this challenge.

For example, score\_sorter([1, 2, 3, 9999, 13], 10000) should return [9999, 13, 3, 2, 1].

1. Sum of Prime Factors

Create a sum\_of\_prime\_factors() function that takes in an integer n and returns the sum of all of its prime factors. As a reminder, a prime number is a number whose only factors are one and itself. Therefore, a prime factor is a factor of a given number that itself is a prime number.

For example, sum\_of\_prime\_factors(91) should return 20 since its prime factors are 13 and