Sort and Search Algorithms

Objectives

Using Python to solve complex problems

- Implement complex programs using Python
- Implement recursive algorithms
- Implement search algorithms

Requirements

Use a **single** search/sort algorithm for all requirements.
Use both **own** search/sort algorithm and **Python functions** filter and sorted.

i. Given a list of numbers, search the list in order to:

1. Determine all the numbers that are Armstrong numbers.

   Note: A number is Armstrong if the sum of cubes of each digit of the number is equal to the number itself.
   Example: $153 = 1^3 + 5^3 + 3^3$

2. Determine all numbers that are even and Armstrong numbers.
3. Determine all numbers that are even or primes or perfect square or Armstrong numbers using a single algorithm with correct parameters.

ii. Sort a list of numbers using:

1. Bubble sort
2. Selection sort
3. Insertion sort
4. Quick sort

iii. Develop an ADT $GeometricalShape$ – information about name (e.g. square, triangle, hexagon), number of sides and length of each side.

1. Create a repository to manage a list of shapes.
2. Filter shapes with more than k sides.
3. Filter shapes with perimeter higher than a given value and name in a given length.
4. Sort shapes based on their perimeter ascending/descending.
5. Sort shapes having name starting with given letter according to perimeter.