El lenguaje de programación elegido es Python.

He creado tres algoritmos : el primero ordena una lista de números de menor a mayor, el segundo hace una búsqueda secuencial de un número dado dentro de la lista , y el tercero hace la búsqueda binaria de un valor dado dentro de esta lista.

A continuación se explica el funcionamiento de cada algoritmo, el rendimiento en notación O Grande y el tiempo de ejecución.

1. Algoritmo para ordenar una lista dada:
   He tratado de simular la misma estrategia que seguiríamos para ordenar una lista por ejemplo de personas por edades que estuviesen en fila. Empezaría por el segundo por la izquierda y la movería a la posición 1 si su edad fuera menor que la del de la posición 1. A continuación cogería a la persona que estuviera en la posición 2 y lo iría moviendo hacia la izquierda si tuviera menos años que el de su izquierda. De esta manera barrería toda la fila y la terminaría ordenando.
   Para este caso he creado una función “ordenar”, que tiene un solo argumento que es la lista de números a ordenar. Lo primero que hago con la función es hacer un for para ir barriendo todas las posiciones desde la 1 hasta la última de la lista. Una vez en la posición, creo una función recursiva “orden” que tiene dos argumentos. El primero llamado valor, que es el valor del número y un segundo argumento llamado count , que lo uso como contador para ir comparando con el elemento anterior. Este contador va disminuyendo en uno cada vez que hace la iteración.
   Como para poder ordenar todos los elementos del listado necesitamos realizar una iteración por cada elemento, la notación es O(n2):cuadrática.
   Para el cálculo de tiempo de ejecución he usado la función default timer del módulo timeit. He sacado la media de diez tiempos registrados dando un valor de 0,399 ms.

2. Algoritmo para búsqueda secuencial:
   El algoritmo sigue las pautas marcadas por el enunciado del ejercicio.
   Como para poder ordenar todos los elementos del listado necesitamos realizar una iteración por cada elemento, la notación es O(n2):cuadrática.
   Para el cálculo de tiempo de ejecución he usado la función default timer del módulo timeit. He sacado la media de diez tiempos registrados dando un valor de 0,882 ms.

3. Algoritmo para búsqueda binaria:
   El algoritmo sigue las pautas marcadas por el enunciado del ejercicio.
   Si consideramos que partimos de una lista ordenada, el algoritmo lo que hace es trocear la lista por lo que la notación es O(logn):logarítmica. En el caso del ejercicio el algoritmo también ordena la lista por lo que será de notación O(n2) cuadrática.
   Para el cálculo de tiempo de ejecución he usado la función default timer del módulo timeit. He sacado la media de diez tiempos registrados dando un valor de 0,574 ms.