Enunciado:

Resuelva los siguientes ejercicios sobre algoritmos de búsqueda y de ordenamiento básicos. Analice el peor y el mejor casos para el tiempo de ejecución en los ejercicios que corresponda. Si lo considera más fácil, puede usar tipos de dato `std::vector<int>` o `std::vector<string>`, según corresponda.

1. Implemente en C++ los algoritmos de búsqueda lineal y binaria para el caso de enteros. Compare su complejidad computacional considerando arreglos de números enteros de diferentes tamaños. Repita ahora la implementación de estos algoritmos, pero considere esta vez un arreglo de cadenas de caracteres (`string`).

2. Considere un arreglo ordenado de $N-1$ enteros de 1 a N con un número faltante, escriba un algoritmo que encuentra el número faltante tal que el tiempo de ejecución de este es $O(\lg n)$. En la implementación de su algoritmo use una función con el siguiente prototipo:

   ```
   int find_missing(int array[], int N);
   ```

   `N` es el tamaño del arreglo `array`.

3. Dado un arreglo de enteros que inicialmente aumenta y luego disminuye, encuentre el valor máximo en el arreglo. Su código solución debe tener el siguiente prototipo:

   ```
   int findMaximum(int arr[], int low, int high);
   ```

   donde `arr` es el arreglo de enteros, `low` y `high` son los índices inferior y superior donde se debe buscar en el arreglo `arr`. Existen al menos dos formas de implementar este algoritmo con diferentes tiempos de ejecución: una lineal y otra logarítmica.

4. *Identity*. Dado un arreglo `a[]` de `N` enteros distintos (positivos o negativos) en orden ascendente. Diseñe un algoritmo para encontrar un índice `i` tal que `a[i] = i`, si tal índice existe.

5. Para los algoritmos de ordenamiento básicos realice un análisis empírico del tiempo de ejecución promedio, $T_{\text{ave}}(N)$, como función del *input size*, N. En una gráfica muestre estas funciones $T_{\text{ave}}^{\text{alg}}(N)$, donde alg = bogosort, selection sort, bubble sort e insertion sort. Considere únicamente arreglos de enteros. Para obtener dicha gráfica proceda de la siguiente manera para cada uno de los algoritmos.

   a) Verifique que su implementación funciona correctamente en arreglos de tamaño moderado, $N = 10$ o 20.

   b) Calcule el tiempo de ejecución del algoritmo para un tamaño fijo de N diez veces, por ejemplo.

   c) Calcule el promedio de los tiempos de ejecución para obtener el valor de la función $T_{\text{ave}}^{\text{alg}}(N)$ para el N dado.

   d) Cree una tabla de dos columnas donde la primera columna tiene el valor de N y la segunda columna contiene $T_{\text{ave}}^{\text{alg}}(N)$.

e) Ejecute este procedimiento para cada uno de los valores del *input size*, N, de interés. Considere por ejemplo, la sucesión $N = 10^s$, donde $s = 2$, 3, 4, 5 y 6.

Finalmente, note que los tamaños máximos de los arreglos que se puedan ordenar dependerán de los recursos computacionales usados.

Tome todas las funciones de ordenamiento básico implementadas e incorpórelas en una librería con archivo encabezado `basord.hh`. Incorpore esta librería en un proyecto que demuestre su utilización.