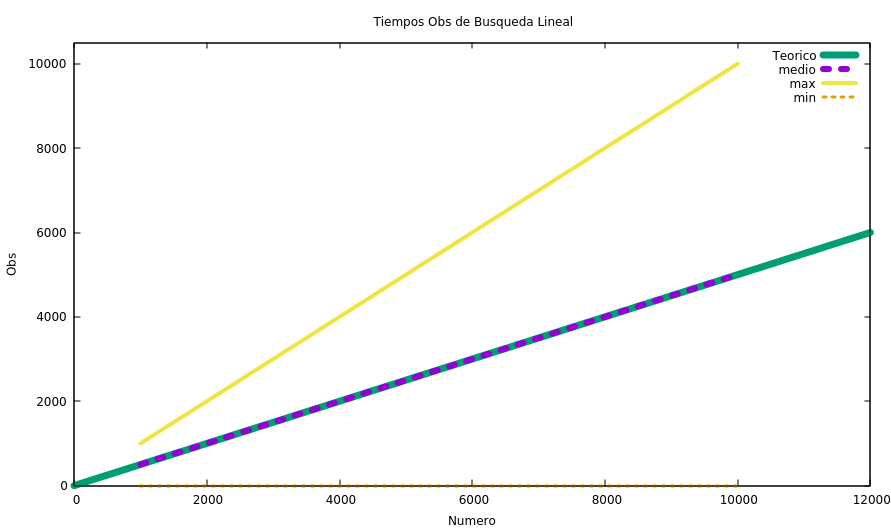
Análisis de Algoritmos 2018/2019

Práctica 3

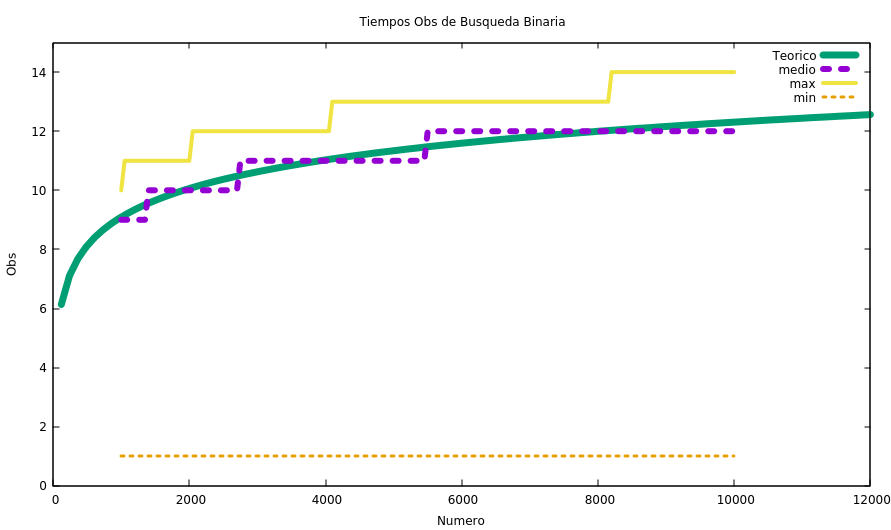
Alba Ramos, Javier Lozano,  Grupo 1212, Pareja 10.

**5. Resultados, Gráficas**

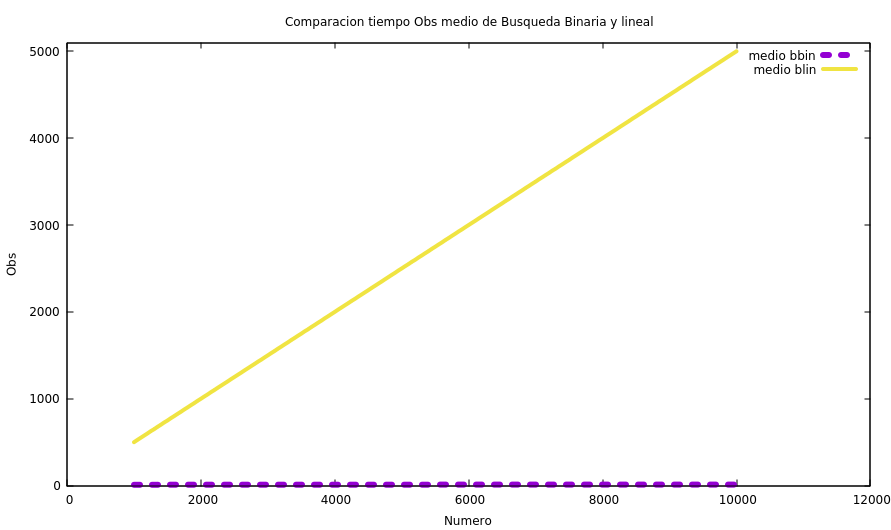
5.2 Apartado 2

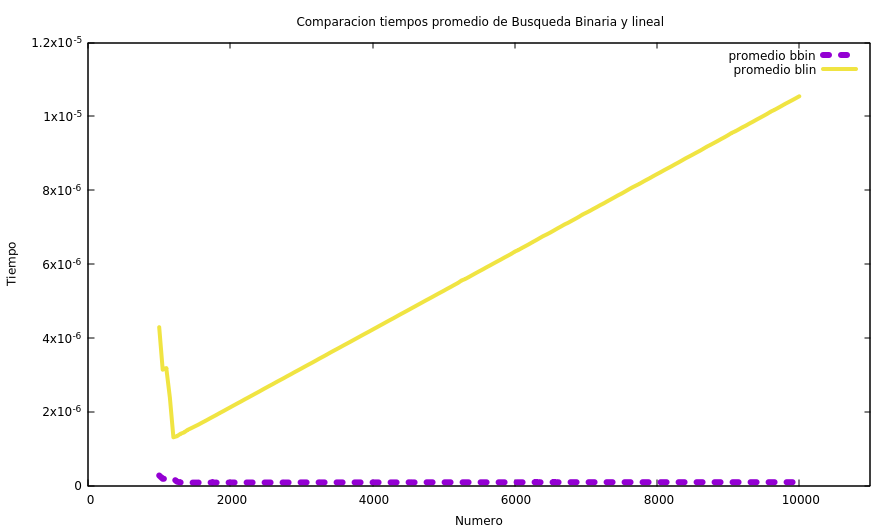


Podemos observar que el caso medio (morado) se ajusta al valor teórico esperado para la búsqueda lineal: N/2 + O(1) que aparece en verde. El caso mejor siempre es uno, por los casos en los que se encuentra el número a la primera, y el caso peor se puede observar que aumenta a la par que la tabla dado que en los casos en los que la tabla esté ordenada el número de operaciones básicas será igual al tamaño de la tabla.



En este caso el caso mejor (naranja) es 1 igual que en la búsqueda lineal, debido a los casos en los que se encuentra el número a la primera. El caso medio (morado) se ajusta dentro de los valores esperados para el caso medio de la búsqueda binaria que es log2(N) + O(1), aunque como se puede observar va haciendo una escalera debido a que se tomara el mismo valor para varios tamaños de tabla al aplicar el logaritmo y quedarnos con su parte entera. El caso peor (amarillo) se puede ver que sigue un crecimiento similar al del caso medio pero en valores más altos.

Podemos observar que mientras que en el caso de la búsqueda lineal (amarillo) alcanza 5000 operaciones básicas, en el caso de la búsqueda binaria (morado) no supera las 15 operaciones básica, por esta razón en la gráfica parece que sean 0.

En este caso, también se observa la diferencia de tiempo entre la búsqueda lineal (amarillo) y la búsqueda binaria (amarillo) que es tan rápida que se aproxima a 0.

**5. Respuesta a las preguntas teóricas.**

5.1 Pregunta 1

La operación básica de bbin es la comparación de claves que, aunque aparecen dos, sólo se contará una vez:

if(tabla[m] == clave)

if(tabla[m] > clave)

La operación básica de blin es la comparacion de claves:

if(tabla[i] == clave)

5.2 Pregunta 2

Para el caso peor de bbin, tenemos Wbbin (N) = log2 (N)

Para el caso mejor de bbin, tenemos Bbbin (N) = 1, si el elemento a buscar está justo en medio de la tabla.

Para el caso peor de blin, tenemos Wblin (N) = N, si el elemento a buscar es el último de la tabla.

Para el caso mejor de blin, tenemos Bblin (N) = 1, si el elemento a buscar es el primero de la tabla.

5.3 Pregunta 3

El orden de bbin en el caso medio es ABbin (N) = log2 (N) + O(1)

El orden de blin en el caso medio es ABlin (N) = N/2 + O(1)

5.4 Pregunta 4

La búsqueda binaria es un algoritmo eficaz cuando la tabla está ordenada. El algoritmo va dividiendo la tabla en mitades y viendo si el elemento a buscar es menor o mayor que el elemento medio, descartando automáticamente la mitad contraria, lo que reduce notablemente el tiempo de búsqueda. Vamos a aplicar la inducción para justificar la corrección del algoritmo. En la cabecera de la función tenemos varios parámetros: tabla es el array ordenado de elementos en el que vamos a buscar, P y U son las cotas inferior y superior del intervalo en el que vamos a buscar, y clave es el número que queremos buscar. La función debe almacenar en pos la posición de la clave si la encuentra, o NO\_ENCONTRADO si no se encuentra, y devolver el número de OBs realizadas para ello.

Es cierto que funciona para tablas de 1 elemento, ya que devuelve 1 operación básica, que es lo que esperaríamos tanto si encuentra el elemento como si no.

Si la clave es el elemento medio, se devolverá una operación básica, que es lo que esperamos.

Asumimos que se cumplirá para un intervalo de n elementos, y vamos a probar que se cumple para intervalos de n+1 elementos. Si la clave es menor que el elemento medio, entonces también es menor que todos los elementos del intervalo [m, U], ya que la tabla está ordenada, y por tanto solo hay que seguir buscando en el intervalo [P, m). Pero este intervalo es menor que el inicial, [P, U], y como la hipótesis de inducción asumía que el método funcionaba para intervalos menores, entonces funciona. Podemos usar el mismo razonamiento para ver que también se cumple para cuando la clave es mayor que el elemento medio.

Para demostrar que la búsqueda lineal es un algoritmo de búsqueda correcto no vamos a usar inducción, sino que vamos a comprobar que el algoritmo termina y que devuelve la salida correcta. Vemos que el algoritmo termina porque el bucle que recorre la tabla va desde el primer elemento hasta el último, y una vez ha acabado el bucle significa que la clave no ha sido encontrada, por tanto devuelve todas las comparaciones realizadas y almacena NO\_ENCONTRADO en el puntero ppos, que es lo que esperaríamos teóricamente si la clave no se encontrase en la tabla. Para el caso de que la clave se encuentre en la tabla, el bucle terminará cuando se encuentre esa clave en la posición i-ésima, la cual se almacenará en ppos y se saldrá de la función devolviendo el número de Obs realizadas hasta el momento. Por estos motivos, podemos concluir que el algoritmo busca correctamente.