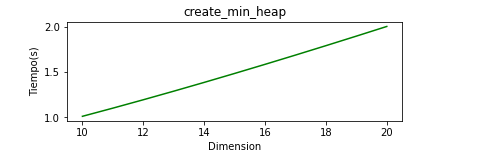
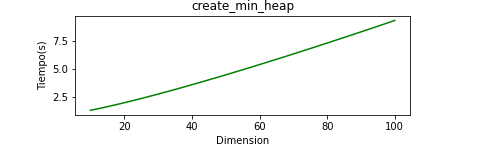
**1. Analizar visualmente los tiempos de ejecución de nuestra función de creación de min heaps. ¿A qué función f se deberían ajustar dichos tiempos?**

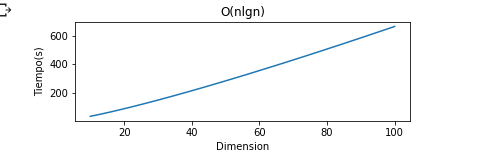
Con un script obtenemos la siguiente función producto de utilizar dimensiones entre 10 y 20 para llamar a create\_min\_heap.



Sin embargo, aquí no se nota mucho de que función podría tratarse así que aumentamos el rango a una dimensión de 100.

Entre 10 y 100:  


Aquí podemos ver una ligera curvatura. Si representamos la función de nlogn veremos que es muy similar:

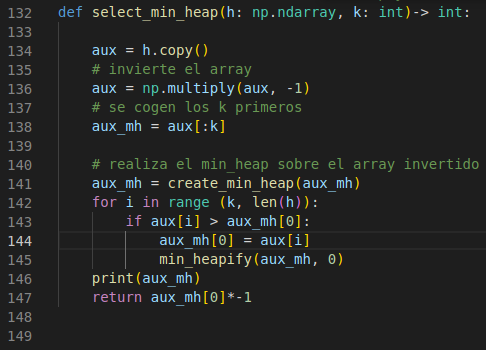


**2. Expresar en función de k y del tamaño del array cual debería ser el coste de nuestra función para el problema de selección.**

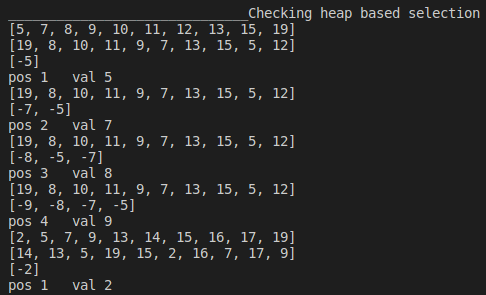
Partimos de que el coste realizar un heapify es O(logn) y que el coste de llamar a create\_min\_heap es el coste de realizar heapify (len(h)-1//2) veces, es decir =O(nlogn), como ya vimos en el apartado anterior, con todo esto, viendo que la selección realiza una inversión del array, costando n después una llamada a create\_min\_heap que cuesta klogk y por último ejecuta en un bucle (len(h) - k) veces min\_heapify con coste logk, el coste en función de K y n, siendo n el tamaño del array, sería n + klogk + (n-k)logk, lo que resulta en O(nlogk).

**3. Una ventaja de nuestra solución al problema de selección es que también nos da los primeros k elementos de una ordenación del array. Explicar por qué esto es así y como se obtendrían.**

Puesto que como podemos observar en nuestro código lo que estamos manteniendo es un heap de los K menores elementos del array cuyos valores se han convertido a su negativo y cuya raíz es el valor que buscamos, es decir, el elemento en la posición K, sabemos que ese heap contiene esos K elementos, menores que el elemento raíz. Para obtenerlos bastaría con devolver el heap al finalizar el bucle. Podemos probar esto con un simple print de la siguiente forma:

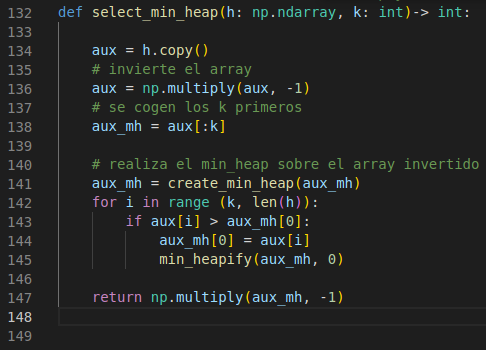


En nuestro caso print(aux\_mh), que nos mostrará los K elementos, aunque con valor negativo, haría falta convertirlos de nuevo a positivo. Como podemos ver:

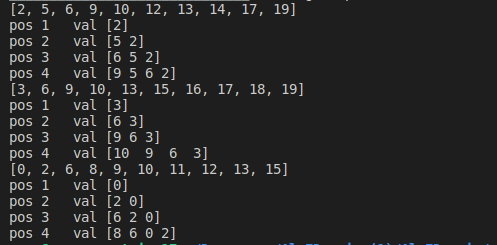


Por ejemplo, en la posición 3 se encuentra el valor 8, si mostramos aux\_mh se mostrarán los valores –8,-5 y –7 que corresponden a los valores de los K elementos primeros mencionados en negativo.

O bien podemos probar el siguiente código:



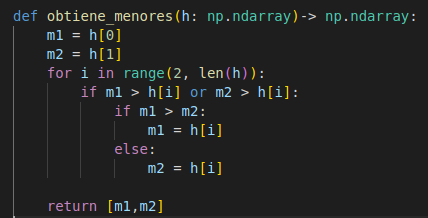
Con el que se obtiene:



se puede observar el correcto funcionamiento.

**4. La forma habitual de obtener los dos menores elementos de un array es mediante un doble for donde primero se encuentra el menor elemento y luego el menor de la tabla restante. ¿Se podrían obtener esos dos elementos con un único for sobre el array? ¿Cómo?**

Si, podemos guardar los 2 primeros valores del array y en un for recorrer el resto de los valores comparando de la siguiente forma:



Si lo probamos veremos que funciona:

H = [1,2,3,52,12,43,513,1,0,-1]

H = [4,-3,-4,-5,-12,-2,-1,0,2,-24,-1]

H = [5,4,3,2,1,0,1,2,3,4,5]

H = [5,4,3,2,1,0,0,1,2,3,4,5]