Trabajo realizado por:
- Álvaro de la Rosa Zarzuelo
- Francisco Javier Dujo Villacorta

Reparto del trabajo:
La mayoría del trabajo lo hemos realizado de forma conjunta en las horas de laboratorio por lo que el reparto ha sido de un 50% cada uno.

Explicación realización practica:
Para calcular la k hemos realizado Pruebas con k desde 3 hasta 30 como se pide en el enunciado, para cada valor de k hemos realizado 40 pruebas para eliminar lo máximo posible la aleatoriedad. El array que hemos utilizado es de tamaño 1000000. Con un tamaño tan grande intentábamos maximizar el numero de operaciones del algoritmo de ordenación y así tener unos valores mas constantes. También con esto se consigue ver que los tiempos del algoritmo no son exagerados, ya que para arrays de tamaños menores puede dar la sensación de que se realiza rápido aunque no sea así.

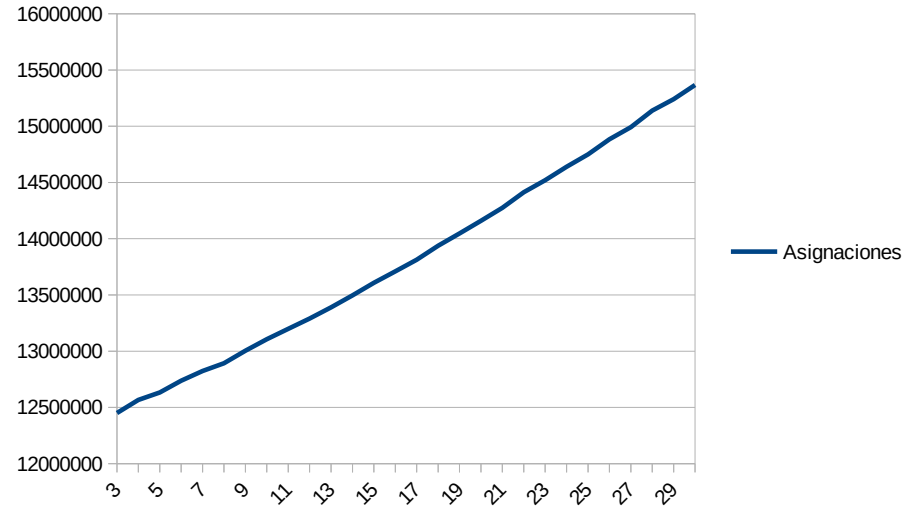
Para analizar la diferencia de rendimiento entre los dos algoritmos hemos usado un array de tamaño inicial 110000 que va aumentando 10000 unidades cada vez. Para cada tamaño realizamos la ordenación con cada vector 40 veces. El numero de veces elegido y el tamaño tienen el mismo motivo que en el apartado de la obtención de k.

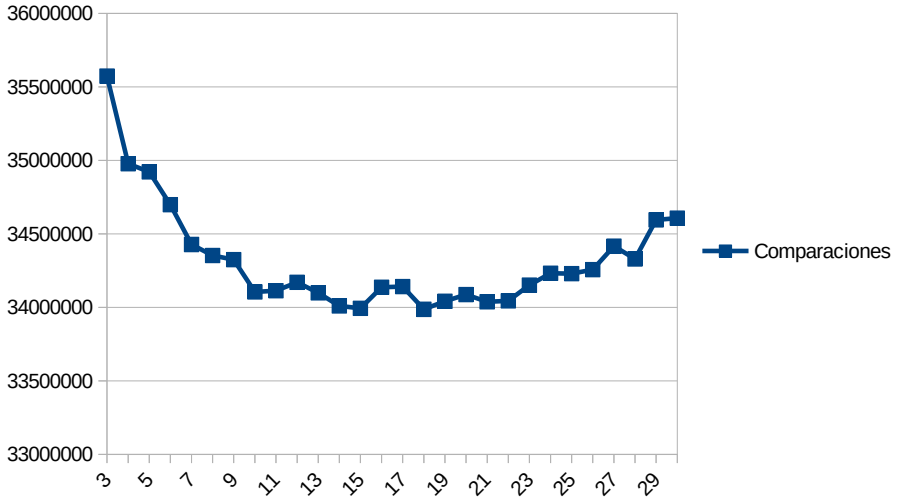
Para analizar los datos obtenidos, creamos desde el programa una hoja de calculo, la cual luego usamos mediante LibreOffice.

Calcular valor de K:

Para obtener el valor mas optimo de K hemos realizado pruebas con un array de 1000000 elementos y 40 repeticiones para cada valor de K. Hemos obtenido los siguientes gráficos que representan las comparaciones y asignaciones.

K	Asignaciones	Comparaciones
3	12449918	35572977
4	12566975	34977762
5	12632369	34922195
6	12737831	34698523
7	12824114	34428010
8	12893471	34353219
9	13003522	34325196
10	13106580	34105232
11	13199066	34113970
12	13289971	34170658
13	13389288	34099299
14	13497004	34010318
15	13608051	33994258
16	13709476	34137403
17	13812961	34141954
18	13937243	33987042
19	14046103	34041789
20	14159085	34087291
21	14273918	34038059
22	14413649	34044273
23	14520567	34151455
24	14640084	34232278
25	14750497	34229500
26	14884034	34257251
27	14992328	34416075
28	15139091	34330115
29	15240548	34595506
30	15366964	34606980





Viendo estos gráficos hemos considerado que el mejor valor de K es 15.

Comparación de los algoritmos:

Teniendo la k, hemos realizado la comparación de los dos algoritmos, el Quicksort normal y el Quicksort modificado con la k obtenida, y obteniendo el pivote mediante la mediana. Importante destacar que estos análisis se han realizado con el peor caso posible para el algoritmo Quickshort que es cuando los vectores están casi ordenados.

Los valores que hemos obtenido son los siguientes:

Tamaño	AsignacionesNormales	AsignacionesModificado	ComparacionesNormales	ComparacionesModificado	TiempoNormal	TiempoModificado
110000	464530	404433	7875811	6846433	29767900	27151250
120000	508318	442981	8714175	7321519	26660525	23338850
130000	552765	481871	9548706	8123672	29126750	25742600
140000	596970	520863	10303770	8817014	31802075	27194125
150000	641821	559387	11086802	9420609	34935050	29798900
160000	686112	598397	12065142	10064150	37897225	31604425
170000	730415	637958	13176229	10976136	41892825	34836625
180000	775204	677231	13699294	11844346	44443425	36694975
190000	821148	717125	14796470	12376774	69069050	54136800
200000	866400	756973	15186354	12906511	55139100	44152275

Calculo Constante Formula de cada uno de los algoritmos:

Una vez tenemos estos datos, hemos calculado la constante para cada algoritmo en cada una de sus mediciones (Asignaciones, Comparaciones y Tiempo). Los valores que hemos obtenido son los siguientes

n*log(n)	AsigN	AsigM	ComN	CompM	TieN	TieM
1276905,92093	0,36379	0,31673	6,16789	5,36174	23,31252	21,26331
1403429,64261	0,36220	0,31564	6,20920	5,21688	18,99670	16,62987
1530787,66483	0,36110	0,31479	6,23777	5,30686	19,02730	16,81657
1658915,67822	0,35986	0,31398	6,21115	5,31493	19,17040	16,39271
1787758,58596	0,35901	0,31290	6,20151	5,26951	19,54126	16,66830
1917268,65507	0,35786	0,31211	6,29288	5,24921	19,76626	16,48409
2047404,13173	0,35675	0,31159	6,43558	5,36100	20,46143	17,01502
2178128,18338	0,35590	0,31092	6,28948	5,43786	20,40441	16,84702
2309408,07672	0,35557	0,31052	6,40704	5,35928	29,90769	23,44185
2441214,52911	0,35491	0,31008	6,22082	5,28692	22,58675	18,08619
Media:	0,35869	0,31293	6,26733	5,31642	21,31747	17,96449

Se aprecia que en todas las medidas es mejor el algoritmo modificado.

Explicación Razonada que algoritmo es mejor:
Partimos de la base de que ambos algoritmos tienen un rendimiento de O(n*ln(n)) pero también sabemos que Quickshort empeora cuando los vectores son muy pequeños. Gracias a que en el algoritmo modificado cambiamos a una ordenación por inserción cuando el tamaño es menor que 15 paliamos un poco este defecto del Quickshort que se aprecia sobretodo en el número de comparaciones y en el tiempo. Además al comenzar con un pivote que es la mediana de 3 de sus elementos (inicial, medio y final) en vez de directamente con el primero también hay una mejora en el rendimiento aunque consideramos que menos significativa que solucionar el problema del algoritmo con vectores pequeños, ya que además en nuestro algoritmo particular si que necesita de alguna comparación y asignación a mayores.
Por todo esto el algoritmo modificado es mejor que el básico cosa que podemos apreciar no solo en las gráficas si no en la constante de sus funciones.
La mejoría del algoritmo modificado respecto al básico es:

- Asignaciones: 14,625%
- Comparaciones: 17,886%
- Tiempo: 18,664%

Como hemos mencionado al inicio, la mejora mas significativa se produce en el tiempo seguido de las comparaciones.