Activity 1. BUBBLE ALGORITHM

n	t ordered	t reverse	t random
10000	304	1487	1066
20000	1215	5948	4259
40000	4917	24098	17175
80000	19389	95969	68664
160000	78174	Oot	Oot

The values all grow at the expected rate for the bubble algorithm, O (n^2) in all cases, however there are some noticeable differences between each case, the algorithm works the fastest for an already ordered vector, as it won't perform any swaps, and it is slowest when the vector is reversed as this is when the most swaps will be performed.

Algorithmics	Student information	Date	Number of session
	UO: 300717	20/02/2025	4
	Surname: Almoina Iglesias		
	Name: Martín		

Activity 2. SELECTION ALGORITHM

n	t ordered	t reverse	t random
10000	300	282	309
20000	1196	1115	1213
40000	4867	4460	4864
80000	19210	17892	19633
160000	77218	72860	77646

The times are almost identical in all three cases as expected, the selection algorithm always performs the same number of comparisons and exchanges so the state of the vector before it is sorted does not affect the complexity or the execution time.

Activity 3. INSERTION ALGORITHM

Algorithmics	Student information	Date	Number of session
	UO: 300717	20/02/2025	4
	Surname: Almoina Iglesias		
	Name: Martín		

n	t ordered	t reverse	t random
10000	LoR	296	149
20000	LoR	1165	583
40000	LoR	4633	2346
80000	LoR	18528	9416
160000	LoR	76159	37857
320000	LoR	Oot	150988
640000	LoR	Oot	Oot
1280000	LoR	Oot	Oot
2560000	46	Oot	Oot
5120000	89	Oot	Oot
10240000	185	Oot	Oot
20480000	367	Oot	Oot
40960000	722	Oot	Oot
81920000	1451	Oot	Oot
163840000	2929	Oot	Oot
327680000	5846	Oot	Oot
655360000	11889	Oot	Oot

As the insertion algorithm has linear complexity for the best case scenario (the vector already being ordered) we can see that in that case its a lot faster and grows linearly, every time n doubles the time doubles, on the other hand when the vector isn't ordered the times grow at the same rate as the previous algorithms as it has the same complexity O(n^2), the execution times are better in the random case as this is a better case than it being inverted.

<table>
<tr><td rowspan="4">Algorithmics</td><td>Student information</td><td>Date</td><td>Number of session</td></tr>
<tr><td>UO: 300717</td><td>20/02/2025</td><td>4</td></tr>
<tr><td>Surname: Almoina Iglesias</td><td></td><td></td></tr>
<tr><td>Name: Martín</td><td></td><td></td></tr>
</table>

Activity 4. QUICKSORT ALGORITHM

n	t ordered	t reverse	t random
250000	31	36	95
500000	60	70	189
1000000	122	140	407
2000000	256	291	856
4000000	527	599	1866
8000000	1111	1226	4252
16000000	2320	2516	10516
32000000	4703	5281	28505

As expected the quicksort algorithm is the most efficient of the four, however due to how it works it is the slowest when the vector is randomized instead of when it is reversed.

Activity 5. QUICKSORT + INSERTION ALGORITHM

n	t random
Quicksort	10516
Q+I (k=5)	1816
Q+I (k=10)	1758
Q+I (k=20)	1738
Q+I (k=30)	1723
Q+I (k=50)	1603
Q+I (k=100)	1580
Q+I (k=200)	1342
Q+I (k=500)	1449
Q+I (k=1000)	1813

Using the insertion algorithm when the vector goes under a specific length (k) makes the quicksort algorithm even quicker, it appears that the best size threshold is around 200 as the times are bigger both when it is higher and lower than 200.