Algorithmics	Student information	Date	Number of session
	UO: 294067	28/02/22	3
	Surname: Díaz Álvarez		
	Name: Paula		

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Activity 1. Subtraction

Subtraction 1 has O(n) complexity (a=1, b=1, k=0), and the times are the ones expected because n doubles and the time also doubles.

Subtraction 2 has O(n^2) complexity (a=1, b=1, k=1), and the times are the ones expected because n doubles and the time is multiplied by 4 (2^2).

Subtraction 3 has O(2^n) complexity (a=2, b=1, k=0), and the times are the ones expected because n increases by 1 and the time is by 2 ($t_2= (2^{n2}/2^{n1})\ t_1 = 2^{n2-n1}\ t_1 = 2^{n1+1-n1}\ t_1 = 2^1\ t_1$)

Subtraction 1 and Substraction 2 doesn't give more times after n=8192, due to an StackOverFlow

Years it would take to complete the Subtraction 3 execution for n=80:

$t_2=(2^{n2}/2^{n1})\ t_1 = (2^{80} / 2^{31}) * 83171$ ms = $4.68 * 10^{19}$ ms = $4.68 * 10^{16}$ s = $7.8 * 10^{14}$ min =

= $1.3 * 10^{13}$ h = $5.42 * 10^{11}$ days = $1.48 * 10^{9}$ years

n	Subtraction 4 (ms)	n	Subtraction 5 (ms)
100	2	30	413
200	17	32	1215
400	130	34	3697
800	1021	36	23876
1600	9420	38	47889
3200	OoT (92878)	40	OoT (105406)
6400	OoT	42	OoT

Algorithmics	Student information	Date	Number of session
	UO: 294067	28/02/22	3
	Surname: Díaz Álvarez		
	Name: Paula		

Years it would take to complete the Subtraction5 execution for n=80:

t_2=($3^{n2\ /2}/3^{n1/2}$) t_1 = ($3^{80/2}/3^{42/2}$) * 105406 ms = 1.23 * 10^{14} ms = 1.23 * 10^{11} s = 2.04 * 10^9 min = 3.4 * 10^7 h = 1.42 * 10^6 days = 3.88 * 10^3 years

Activity 2. Division

Division 1 has O(n) complexity (a=1, b=3, k=1), and the times are the ones expected because n doubles and the time also doubles.

Division 2 has O(n log n) complexity (a=2, b=2, k=1), and the times are the ones expected because n doubles and the time also doubles (more or less).

Division 3 has O(n) complexity (a=2, b=2, k=0), and the times are the ones expected because n doubles and the time also doubles.

n	Division 4 (ms)	n	Division 5 (ms)
1000	10	1000	23
2000	36	2000	99
4000	142	4000	392
8000	518	8000	1588
16000	2053	16000	6719
32000	9088	32000	33809
64000	50705	64000	OoT(128722)
128000	OoT	128000	OoT

Algorithmics	Student information	Date	Number of session
	UO: 294067	28/02/22	3
	Surname: Díaz Álvarez		
	Name: Paula		

Activity 3. VectorSum & Fibonacci

Vector sum			
n	**Option 1 (ms)**	**Option 2 (ms)**	**Option 3 (ms)**
3	0.00004	0.00007	0.00010
6	0.00007	0.00010	0.00020
12	0.00009	0.00022	0.00043
24	0.00014	0.00040	0.00088
48	0.00024	0.00078	0.00178
96	0.00042	0.00157	0.00294
192	0.00073	0.00299	0.00592
384	0.00138	0.00586	0.01216
768	0.00269	0.01178	0.02395
1536	0.00549	0.02357	0.07217
3072	0.01075	0.05208	0.22989
6144	0.02182	0.30385	0.31138
12288	0.04737	StackOverFlow	0.43770
24576	0.23567	StackOverFlow	0.87445
49152	0.30187	StackOverFlow	OoT
98304	0.39259	StackOverFlow	OoT

Algorithmics	Student information	Date	Number of session
	UO: 294067	28/02/22	3
	Surname: Díaz Álvarez		
	Name: Paula		

In terms of efficiency the first option (the iterative version) is the best, as it gives small times and doesn't overflow.

The second best option to use would be option 3 (divide and conquer by division), as even with big numbers doesn't overflow.

Option 2 (divide and conquer by subtraction) gives better times than 3, but it only works for sizes less than 12288.

Fibonacci				
n	**Option 1 (ms)**	**Option 2 (ms)**	**Option 3 (ms)**	**Option 4 (ms)**
10	0.000084	0.000116	0.000169	0.003
11	0.000086	0.000117	0.000174	0.005
12	0.000089	0.000119	0.000213	0.006
13	0.000096	0.000122	0.000224	0.010
14	0.000096	0.000128	0.000240	0.015
15	0.000103	0.000133	0.000252	0.025
16	0.000107	0.000142	0.000271	0.038
17	0.000109	0.000147	0.000289	0.062
18	0.000113	0.000153	0.000311	0.100
19	0.000118	0.000161	0.000320	0.164
20	0.000120	0.000168	0.000333	0.266
21	0.000125	0.000175	0.000352	0.431
22	0.000131	0.000180	0.000360	0.681

23	0.000139	0.000182	0.000372	1.099
24	0.000142	0.000194	0.000381	1.769
25	0.000150	0.000208	0.000400	2.875
26	0.000152	0.000212	0.000417	4.671
27	0.000157	0.000214	0.000428	7.494
28	0.000160	0.000228	0.000441	12.299
29	0.000167	0.000225	0.000455	40.522
30	0.000170	0.000235	0.000468	62.577
31	0.000173	0.000241	0.000490	OoT
32	0.000180	0.000251	0.000498	OoT
33	0.000182	0.000252	0.000505	OoT
34	0.000186	0.000260	0.000529	OoT
35	0.000190	0.000266	0.000555	OoT
36	0.000191	0.000273	0.000566	OoT
37	0.000195	0.000279	0.000654	OoT
38	0.000201	0.000285	0.000674	OoT
39	0.000204	0.000295	0.000690	OoT
40	0.000204	0.000297	0.000706	OoT
41	0.000213	0.000303	0.000721	OoT
42	0.000215	0.000322	0.000852	OoT

43	0.000219	0.000324	0.001522	OoT
44	0.000227	0.000329	0.001540	OoT
45	0.000227	0.000331	0.001579	OoT
46	0.000234	0.000344	0.001585	OoT
47	0.000235	0.000345	0.001619	OoT
48	0.000240	0.000352	0.001691	OoT
49	0.000245	0.000360	0.001692	OoT
50	0.000248	0.000370	0.001728	OoT
51	0.000252	0.000371	0.001738	OoT
52	0.000263	0.000375	0.001793	OoT
53	0.000263	0.000382	0.001802	OoT
54	0.000265	0.000388	0.001859	OoT
55	0.000269	0.000398	0.001903	OoT
56	0.000274	0.000409	0.001905	OoT
57	0.000276	0.000413	0.001967	OoT
58	0.000276	0.000417	0.002002	OoT
59	0.000280	0.000422	0.002166	OoT

In terms of efficiency, the best option would be option 1 (iterative version), then 2 (iterative version with array), then 3 (recursive option O(n)), and then 4 (recursive option O(1.6 n), specially for bigger sizes

Algorithmics	Student information	Date	Number of session
	UO: 294067	28/02/22	3
	Surname: Díaz Álvarez		
	Name: Paula		

Activity 4. Mergesort

n	t ordered	t reverse	t random
31250	LoR	LoR	LoR
2*31250	LoR	LoR	64
2^2*31250	81	96	144
2^3*31250	166	197	282
2^4*31250	393	410	475
2^5*31250	856	849	880
2^6*31250	1712	1775	1705
2^7*31250	3450	3476	3560
2^8*31250	9645	9552	7067
2^9*31250	20846	21880	14830
2^{10}*31250	44054	43612	31076
2^{11}*31250	OoT	OoT	OoT

n	t Mergesort (t1)	t Quicksort (t2)	t1/t2
250000	166	123	1.35
2*250000	415	254	1.63
2^2*250000	869	955	0.91

Algorithmics	Student information	Date	Number of session
	UO: 294067	28/02/22	3
	Surname: Díaz Álvarez		
	Name: Paula		

2^3*250000	1720	1222	1.41
2^4*250000	3588	2677	1.34
2^5*250000	7243	8454	0.86
2^6*250000	14965	18286	0.82
2^7*250000	31185	54440	0.57
2^8*250000	OoT	OoT	-