SORTARI

ENESCU IRINA STEFANIA

GRUPA 133

TIPURI DE SORTARI IMPLEMENTATE

- Python Sort – sortare predefinita (Tim Sort)
- Counting Sort
- Radix sort (cu baza 10, cu baza 2, baza 2^8 si cu baza 2^{16})
- Shell Sort (cu secvente impartite in 2 si 10)
- Quick Sort (cu pivotul ultimul element si mediana din trei)
- Merge Sort

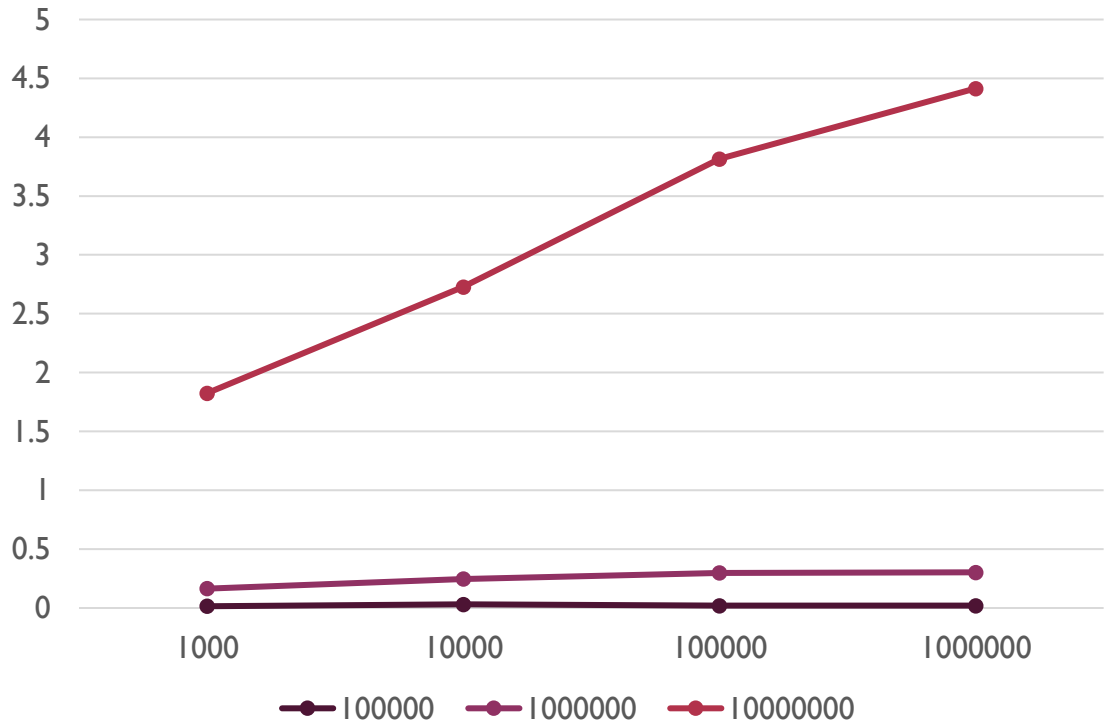
Observatie! Datele din tabel sunt calculate pe baza numarului de elemente care trebuie sortate (mov), numerele fiind generate automat, random si celui mai mare numar care poate fi generat (alb). Acestea reprezinta timpii de rulare.

PYTHON – SORTARE PREDEFINITA

	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.0 s	0.000999 s	0.008002 s	0.084987 s	0.823636 s	8.845091 s
10^2	0.0 s	0.0 s	0.0 s	0.001998 s	0.012003 s	0.110018 s	1.17298 s	12.33223 s
10^3	0.0 s	0.0 s	0.001003 s	0.001998 s	0.015004 s	0.164003 s	1.823999 s	35.89435 s
10^4	0.0 s	0.0 s	0.001001 s	0.001000 s	0.029011 s	0.246004 s	2.727001 s	32.71949 s
10^5	0.0 s	0.0 s	0.000999 s	0.002000 s	0.019001 s	0.298003 s	3.815035 s	53.51286 s
10^6	0.0 s	0.0 s	0.001000 s	0.002000 s	0.019001 s	0.301988 s	4.412701 s	74.21095 s
10^7	0.0 s	0.0 s	0.000999 s	0.000999 s	0.019020 s	0.299999 s	4.507581 s	83.89178 s
10^8	0.0 s	0.0 s	0.001000 s	0.002002 s	0.019997 s	0.303013 s	4.487124 s	77.30881 s

PYTHON – SORTARE PREDEFINITA

- Observam ca odata cu cresterea numarului de elemente si cu cresterea maximului, cresc si timpii de rulare.
- Diferentele dintre timpii de rulare pentru acelasi numar de elemente, dar cu maxim diferit sunt relativ mici pentru valori mici.

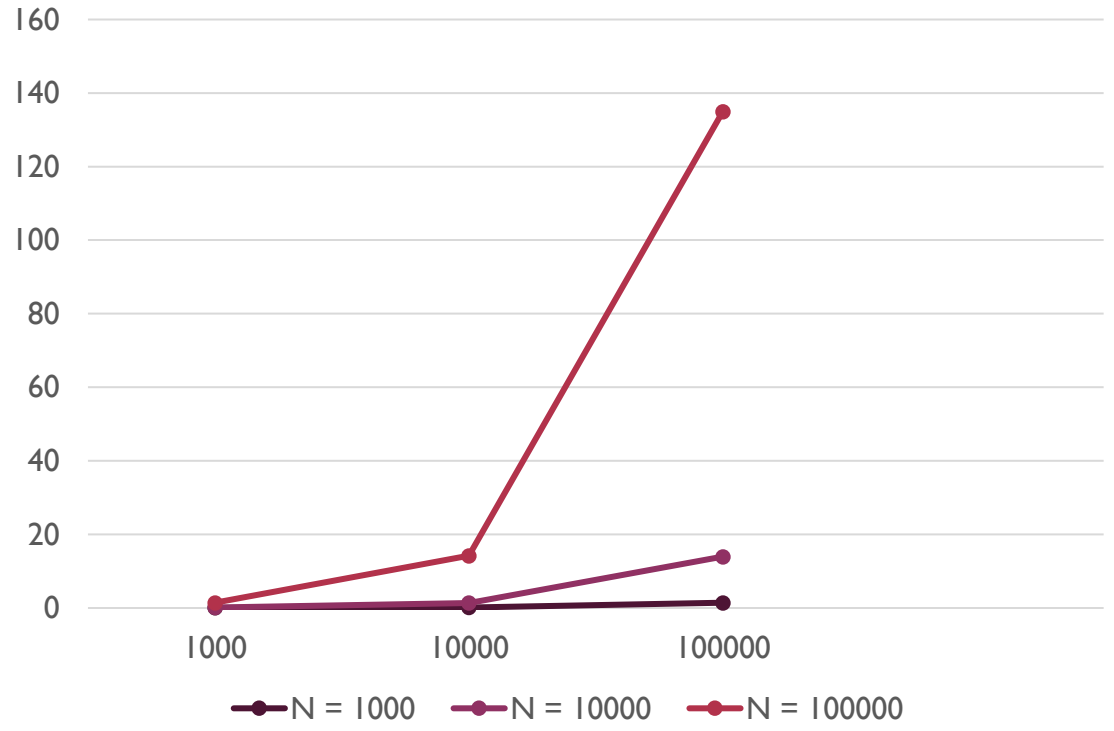


COUNTING SORT

	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.0 s	0.004992 s	0.056999 s	0.436025 s	3.88184 s	38.66556 s
10^2	0.0 s	0.001001 s	0.001999 s	0.021004 s	0.158022 s	1.626016 s	15.67614 s	
10^3	0.0 s	0.000995 s	0.014001 s	0.137001 s	1.389008 s	14.13526 s	140.5477 s	
10^4	0.002994 s	0.014992 s	0.137023 s	1.344999 s	14.16201 s	136.1437 s		
10^5	0.051995 s	0.152024 s	1.392983 s	13.89495 s	134.9139 s			
10^6	0.429025 s	1.603982 s	13.74300 s	130.4229 s				
10^7	3.266975 s	15.78301 s	131.9456 s					
10^8	30.429 s	152.4660 s						

COUNTING SORT

- Observam ca odata cu cresterea numarului de elemente si cu cresterea maximului, cresc si timpii de rulare foarte mult.
- Diferentele dintre timpii de rulare pentru acelasi numar de elemente, dar cu maxim diferit sunt foarte mari chiar si pentru valori relativ mici, dar pentru valori si mai mari.



RADIX SORT (BAZA 10)

	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.000995 s	0.005001 s	0.050022 s	0.477982 s	4.527994 s	
10^2	0.0 s	0.0 s	0.002 s	0.009 s	0.09 s	0.941478 s	8.811442 s	
10^3	0.0 s	0.0 s	0.000998 s	0.015999 s	0.152998 s	1.558776 s	14.98846 s	
10^4	0.0 s	0.0 s	0.001996 s	0.020001 s	0.219022 s	2.738017 s	19.39943 s	
10^5	0.0 s	0.000997 s	0.001998 s	0.020975 s	0.236022 s	2.922266 s	25.99021 s	
10^6	0.0 s	0.0 s	0.001999 s	0.048003 s	0.292019 s	3.281409 s	29.94621 s	
10^7	0.0 s	0.0 s	0.005996 s	0.044974 s	0.33303 s	3.62398 s	34.64791 s	
10^8	0.001 s	0.001001 s	0.003 s	0.038001 s	0.367008 s	6.176559 s	42.02037 s	

RADIX SORT (BAZA 2)

	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.001996 s	0.019001 s	0.171997 s	1.857077 s	20.28343 s	183.6373 s
10^2	0.0 s	0.0 s	0.004 s	0.03 s	0.357999 s	3.299995 s	33.02238 s	
10^3	0.0 s	0.000997 s	0.004998 s	0.042976 s	0.495008 s	5.119043 s	49.64400 s	
10^4	0.0 s	0.142 s	0.005999 s	0.066006 s	0.816009 s	7.306526 s	77.23461 s	
10^5	0.0 s	0.025005 s	0.006997 s	0.085014 s	0.818018 s	8.390724 s	84.33582 s	
10^6	0.0 s	0.001004 s	0.010995 s	0.084976 s	1.011977 s	10.11050 s		
10^7	0.000995 s	0.002002 s	0.010997 s	0.117983 s	1.189977 s	12.15266 s		
10^8	0.0 s	0.001998 s	0.009999 s	0.135997 s	1.318022 s	13.46509 s		

RADIX SORT (BAZA 2^8)

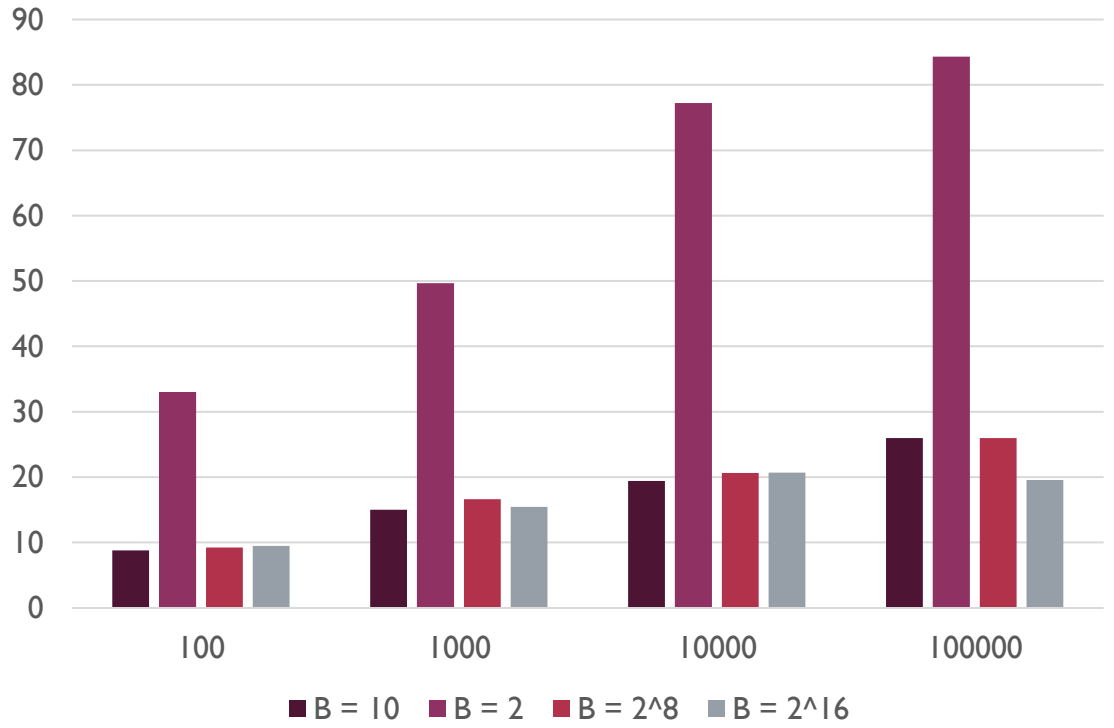
	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.001 s	0.004 s	0.049 s	0.440999 s	4.588997 s	49.25334 s
10^2	0.0 s	0.0 s	0.001 s	0.008002 s	0.091009 s	0.91348 s	9.237066 s	
10^3	0.0 s	0.0 s	0.001999 s	0.012001 s	0.130989 s	1.608972 s	16.61500 s	
10^4	0.0 s	0.0 s	0.002002 s	0.017002 s	0.190428 s	2.030998 s	20.62658 s	
10^5	0.0 s	0.0 s	0.001999 s	0.02 s	0.24435 s	2.54498 s	25.98809 s	
10^6	0.0 s	0.0 s	0.001999 s	0.025003 s	0.291002 s	3.005236 s	31.03997 s	
10^7	0.0 s	0.0 s	0.004 s	0.03 s	0.35302 s	3.569419 s	38.28324 s	
10^8	0.0 s	0.0 s	0.002999 s	0.06 s	0.385982 s	4.073175 s	40.65020 s	

RADIX SORT (BAZA 2^{16})

	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.0 s	0.005001 s	0.048022 s	0.510005 s	4.586060 s	
10^2	0.0 s	0.0 s	0.001998 s	0.007999 s	0.087023 s	0.923023 s	9.459978 s	
10^3	0.0 s	0.0 s	0.000998 s	0.015002 s	0.142982 s	1.658016 s	15.43158 s	
10^4	0.0 s	0.0 s	0.002002 s	0.027997 s	0.200969 s	2.084011 s	20.69903 s	
10^5	0.0 s	0.0 s	0.001998 s	0.018998 s	0.193025 s	2.041024 s	19.53907 s	
10^6	0.0 s	0.0 s	0.002999 s	0.020998 s	0.228001 s	2.503585 s	23.92461 s	
10^7	0.0 s	0.0 s	0.002999 s	0.026025 s	0.28599 s	3.025994 s	29.18582 s	
10^8	0.0 s	0.0 s	0.005001 s	0.03198 s	0.335917 s	3.553986 s	36.40142 s	

RADIX SORT – ANALIZA BAZE

- Observam ca Radix Sort cu baza 10, cu baza 2^8 si cu baza 2^{16} au timpi de rulare apropiati. Cu toate acestea, Radix Sort cu baza 2^{16} pare cel mai rapid. Acest algoritm se poate implementa si cu operatii de shiftare pe biti.
- Odata cu cresterea numarului de elemente si cu cresterea maximului, cresc si timpii de rulare pentru toti algoritmii Radix Sort.



SHELL SORT (SECVENTE IMPARTITE LA 2)

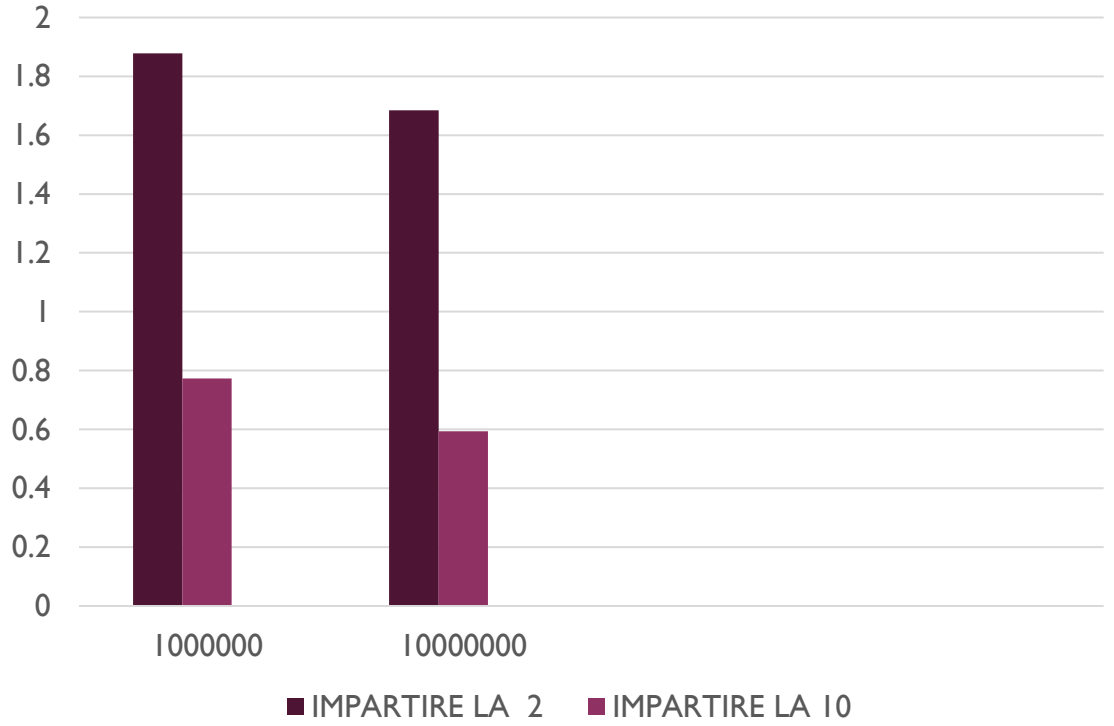
	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.000996 s	0.020004 s	0.133023 s	1.617989 s	19.97416 s	242.9119 s
10^2	0.0 s	0.0 s	0.001004 s	0.011996 s	0.142998 s	1.643005 s	19.53543 s	251.1182 s
10^3	0.0 s	0.0 s	0.001001 s	0.017999 s	0.153977 s	1.76302 s	20.43416 s	257.5039 s
10^4	0.0 s	0.0 s	0.0 s	0.015007 s	0.140021 s	1.790421 s	21.47777 s	266.4759 s
10^5	0.0 s	0.0 s	0.000997 s	0.018013 s	0.145991 s	1.950495 s	21.43550 s	409.3443 s
10^6	0.0 s	0.0 s	0.000997 s	0.014025 s	0.151994 s	1.878165 s	21.62024 s	
10^7	0.0 s	0.0 s	0.0 s	0.013997 s	0.14102 s	1.684001 s	21.48826 s	
10^8	0.0 s	0.0 s	0.002002 s	0.011002 s	0.144974 s	1.732022 s	21.62731 s	

SHELL SORT (SECVENTE IMPARTITE LA 10)

	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.000997 s	0.005 s	0.049 s	0.579997 s	7.196911 s	75.23582 s
10^2	0.0 s	0.0 s	0.0 s	0.003995 s	0.058002 s	0.631017 s	6.696003 s	79.86523 s
10^3	0.0 s	0.0 s	0.0 s	0.004999 s	0.050004 s	0.650999 s	7.573008 s	83.07631 s
10^4	0.0 s	0.0 s	0.0 s	0.004996 s	0.05 s	0.664999 s	7.315013 s	88.69369 s
10^5	0.0 s	0.0 s	0.001003 s	0.004999 s	0.049015 s	0.596996 s	7.158249 s	91.83438 s
10^6	0.0 s	0.0 s	0.000999 s	0.004001 s	0.046 s	0.773017 s	6.669014 s	80.95689 s
10^7	0.0 s	0.0 s	0.0 s	0.004002 s	0.049002 s	0.592994 s	6.915203 s	80.33124 s
10^8	0.0 s	0.0 s	0.0 s	0.003996 s	0.062003 s	0.662978 s	6.776005 s	81.80358 s

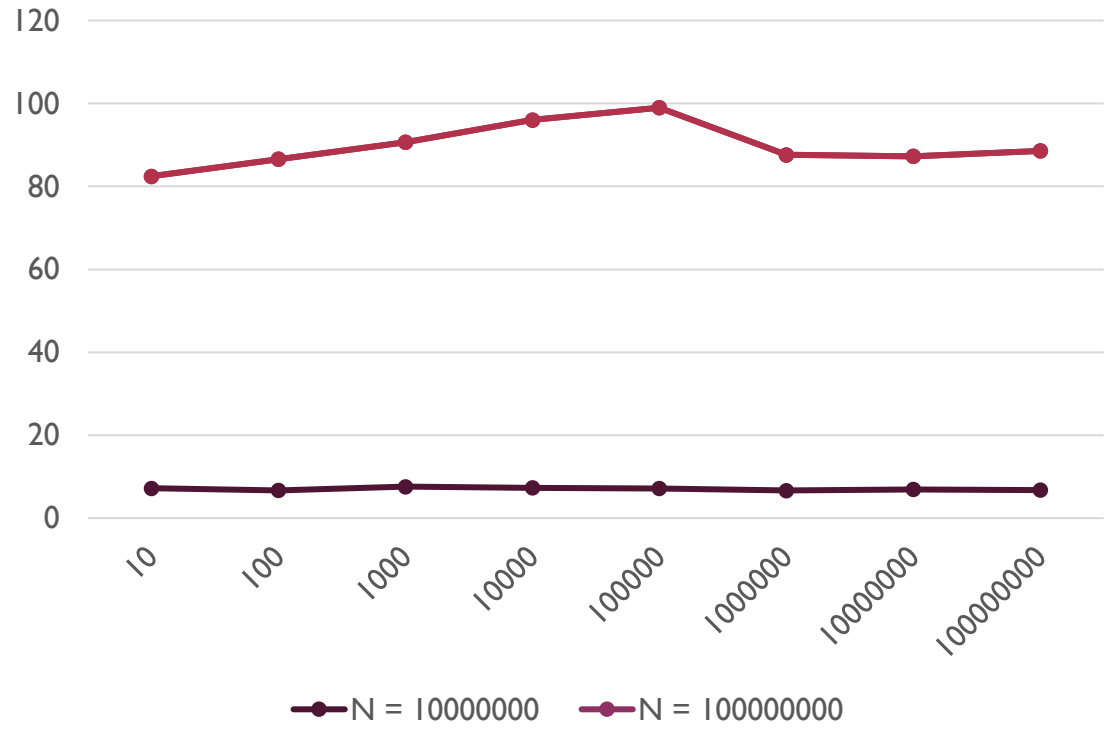
SHELL SORT – ANALIZA IMPARTIRI

- Observam ca daca impartirea se realizeaza la un numar mai mare, algoritmul este mai eficient si timpul de rulare este mai mic. Acest lucru se observa foarte bine in cazul in care avem 10^8 elemente cu maximul 10: pentru Shell Sort (2) timpul de rulare este 242.9119 s, in timp ce Shell Sort (10) sorteaza in 75.23582 s.



SHELL SORT – OBSERVATIE

- Observam (mai ales in cazul N = 100000000) ca timpul de rulare este in crestere pana la maxim = 100000, iar apoi acesta incepe sa scada.



MERGE SORT

	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.003997 s	0.036996 s	0.460021 s	6.459963 s	65.16255 s	
10^2	0.0 s	0.0 s	0.004002 s	0.034998 s	0.477 s	6.66069 s	67.05497 s	
10^3	0.0 s	0.0 s	0.003996 s	0.035 s	0.502024 s	6.656906 s	77.99634 s	
10^4	0.0 s	0.0 s	0.002999 s	0.039025 s	0.485023 s	6.262549 s	74.90290 s	
10^5	0.0 s	0.0 s	0.003006 s	0.044999 s	0.497022 s	6.625016 s	74.57346 s	
10^6	0.0 s	0.0 s	0.002999 s	0.036977 s	0.517973 s	6.473806 s	75.46189 s	
10^7	0.0 s	0.0 s	0.003992 s	0.037024 s	0.475975 s	6.685985 s	78.24316 s	
10^8	0.0 s	0.001 s	0.004002 s	0.03998 s	0.514018 s	6.515384 s	83.6076 s	

QUICK SORT (PIVOT – ULTIMUL ELEMENT)

	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.002001 s	-	-	-	-	-
10^2	0.0 s	0.0 s	0.002 s	-	-	-	-	-
10^3	0.0 s	0.0 s	0.002002 s	-	-	-	-	-
10^4	0.0 s	0.0 s	0.001998 s	-	-	-	-	-
10^5	0.0 s	0.0 s	0.001003 s	-	-	-	-	-
10^6	0.0 s	0.0 s	0.000999 s	-	-	-	-	-
10^7	0.0 s	0.0 s	0.002002 s	-	-	-	-	-
10^8	0.0 s	0.0 s	0.001 s	-	-	-	-	-

QUICK SORT (PIVOT – MEDIANA DIN 3)

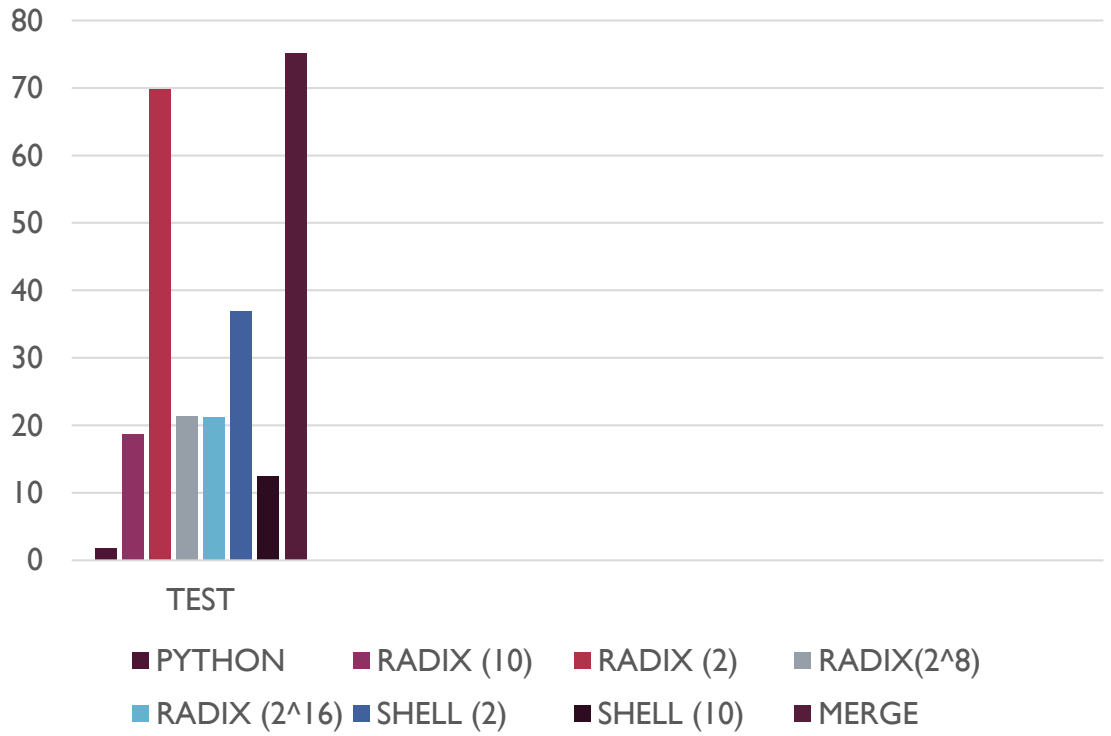
	10	10^2	10^3	10^4	10^5	10^6	10^7	10^8
10	0.0 s	0.0 s	0.002 s	-	-	-	-	-
10^2	0.0 s	0.0 s	0.001001 s	-	-	-	-	-
10^3	0.0 s	0.0 s	0.002 s	-	-	-	-	-
10^4	0.0 s	0.0 s	0.000999 s	-	-	-	-	-
10^5	0.0 s	0.0 s	0.001 s	-	-	-	-	-
10^6	0.0 s	0.0 s	0.002 s	-	-	-	-	-
10^7	0.0 s	0.0 s	0.001004 s	-	-	-	-	-
10^8	0.0 s	0.0 s	0.001997 s	-	-	-	-	-

TEST 1

- Pe acest test pot rula toti algoritmii.
- Observam ca programul care ruleaza cel mai greu este Counting Sort.
- Pentru a face o comparatie corecta intre ceilalti algoritmi, rulam un test fara Counting Sort si Quick Sort.

```
Numarul de elemente = 1000
Valoarea maxima a unui element = 10000000
✓ PYTHON SORT                0.000998 s
✓ COUNTING SORT              138.67437 s
✓ RADIX SORT (baza 10)       0.003003 s
✓ RADIX SORT (baza 2)        0.01 s
✓ RADIX SORT (baza 2^8)      0.002982 s
✓ RADIX SORT (baza 2^16)     0.002017 s
✓ SHELL SORT (2)             0.001 s
✓ SHELL SORT (10)            0.0 s
✓ QUICK SORT (U)             0.002001 s
✓ QUICK SORT (MED)           0.001 s
✓ MERGE SORT                 0.003 s
================================================
```

TEST 2




```
Numarul de elemente = 10000000
Valoarea maxima a unui element = 1000
✓ PYTHON SORT                 1.743017 s
✓ RADIX SORT (baza 10)        18.730745 s
✓ RADIX SORT (baza 2)         69.80427 s
✓ RADIX SORT (baza 2^8)       21.347054 s
✓ RADIX SORT (baza 2^16)      21.207942 s
✓ SHELL SORT (2)              36.934995 s
✓ SHELL SORT (10)             12.415205 s
✓ MERGE SORT                  75.135043 s
================================================
```

CLASAMENT (CEI MAI RAPIZI … CEI MAI LENTI)

1. PYTHON SORT
2. SHELL SORT (10)
3. RADIX SORT (10)
4. RADIX SORT (2^16)
5. RADIX SORT (2^8)
6. SHELL SORT (2)
7. RADIX SORT (2)
8. MERGE SORT
9. COUNTING SORT