**Radix sort:**

* Facut folosind buckets, functioneaza in orice baza, insa baza aleasa este importanta – cele mai bune rezultate sunt cand baza aleasa este intr-o proportie cu numarul de elemente din vector
* Spre exemplu, pentru 10^4 elemente, radix cu baza 2^8 sorteaza mult mai repede decat radix cu baza 2^16, insa pentru n > 10^6, bazele 2^12 pana la 2^16 sorteaza mult mai repede decat bazele mici
* O(n + b) pe caz general, unde b reprezinta numarul de bucketuri

**Merge sort:**

* O(n log n)
* Pentru un anumit motiv, posibil din cauza limitarilor calculatorului meu, nu merge pe inputuri cu n > ~2\*10^7
* Insa pe caz general, timpul in care sorteaza un vector o anumita marime este aproximativ la fel indiferent de maximul pe care il poate avea un element, spre exemplu, pentru n = 10^7, merge sort sorteaza vectorul in 8-9 secunde indiferent daca max = 10^4 sau 10^8

**Count sort:**

* O(n + max)
* Este complet superior oricarui alt algoritm dpdv al timpului in caz general, singura exceptie fiind atunci cand n este mult mai mic decat max. Spre exemplu, pentru n = 10^4 si max = 10^8, count sort termina in 0.49 secunde, insa toti ceilalti agloritmi sunt mai rapizi
* De asemenea, pentru maxim foarte mare, ocupa mult spatiu, iar in cazul meu, nu functioneaza pentru max > 10^9.
* Pe langa aceste doua exceptii, count sort este detasat cel mai rapid algoritm de sortare dintre cele implementate de mine

**Shell sort:**

* O(n ^ (3/2)) folosind secventa lui Knuth, (k\*3 - 1)/2
* Pentru inputuri mici este foarte bun, cand diferenta dintre acesta si un algoritm cu o complexitate mai buna este neglijabila
* Insa pentru inputuri mari, timpul explodeaza, fiind pana la urma aproape n^2, fiind nevoit sa limitez n-ul maxim pentru shell sort la 2\*10^7

**Comb sort:**

* O(n^2 / 2^k) complexitate average, unde k este numarul de decrementari, are worst case in O(n^2) si best case in O(n log n)
* Este o imbunatatire a algoritmului bubble sort
* Functioneaza foarte bine pentru inputuri cu n mic, indiferent de max
* Pentru n foarte mare, functioneaza mai bine decat shell sort, nu imi ingheata calculatorul pentru n = 10^8, insa tot se resimte factorul de n^2, fiind cel mai lent in acest caz pe langa radix sort in bazele 10 si 16

**Niste observatii pentru doua cazuri speciale:**

* In ultimul tabel am luat 3 vectori pentru n = max = 10^7, unul sortat crescator, altul descrescator si unul random pentru a observa comportamentul agloritmilor pentru vectori deja sortati
* Se observa ca pentru algoritmii bazati pe numarare, timpii de executie sunt aproape identici pentru cei 3 vectori prin faptul ca ordinea elementelor nu este luata in considerare
* Insa pentru ceilalti, anume STD, merge, shell si comb se observa niste diferente
* Pentru Merge sort timpii sunt aproape la fel, insa pentru un vector descrescator, sorteaza cu 2 secunde mai lent decat unul aleator
* Ceilalti algoritmi au insa imbunatatiri mari, in special shell sort, care a trecut de la 21 de secunde pe vector random la 3-5 secunde pe cei sortati