Det er **testet** forskjellige inputfiler, med variasjoner av tall tallsekvenser, slik at programmet både har forsøkt å sortere allerede sorterte sekvenser, fullstendig usorterte sekvenser, samt delvis sorterte sekvenser. Alt ser ut til å stemme.

Sammenlikninger er telt kun ved sammenlikning av elementer i tall- sekvensen. Altså er det ikke telt sammenlikninger ved for eksempel sammenlikning av en teller i en while- loop, med størrelsen på en tallsekvens.

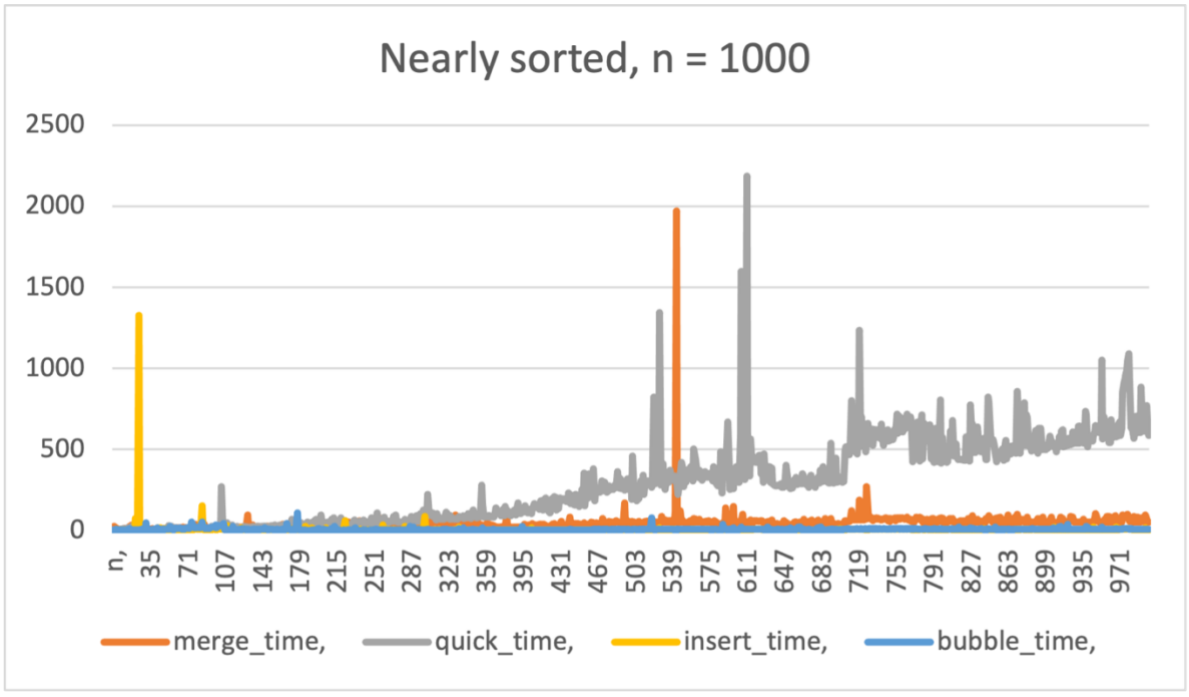
Bytter er telt alle steder der det byttes om to elementer (*swap()*), samt i algoritmer der det settes inn elementer fra input- tallene i en ny sekvens.

På **nesten sorterte sekvenser** er *Insertionsort*, *Bubblesort* og *Mergesort* de algoritmene som kjører raskest. Dette til tross for at de to førstnevnte er $O(n^2)$. *Quicksort* derimot, gjør det betydelig dårligere (*figur 1*).

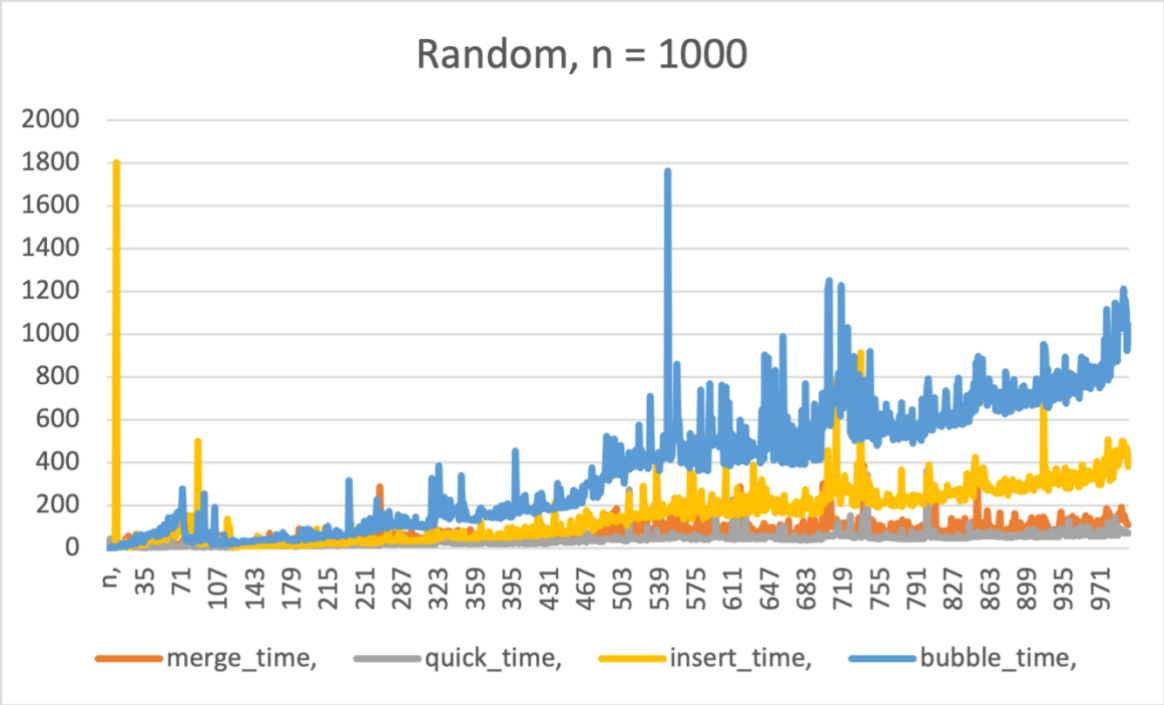
På **usorterte sekvense**r er algoritmene med O(nlog(n)), *Mergesort* og *Quicksort*, de klare vinnerne (*figur 2*).

Teorien stemmer godt med praksis på usorterte sekvenser, hvor det er lett å se at *Insertionsort og Bubblesort* nærmer seg en kvadratisk kurve når **n blir stor**, og *Mergesort* og *Quicksort* virker nokså **upåvirket av stor n**. På delvis sorterte sekvenser stemmer derimot ikke teoretisk kjøretidskompleksitet like godt overens med testen.

Det kan se ut som et **økende antall bytter** impliserer drøyere kjøretid. På nesten sorterte sekvenser ligger de raskeste (Insertionsort og Bubblesort) godt under *Quicksort*, som bruker lengst tid. Samtidig er antallet bytter *Insertionsort* og *Bubblesort* utfører på en usortert sekvens godt over det Quicksort utfører.



Figur 1



figur 2

Tatt både graden av sortering på input, samt størrelsen på input i betraktning, er *Mergesort* den algoritmen som fungerer best. Denne holder seg stabilt rask.

Videre er det interessant at det plutselig er kraftige utslag i grafene, på tilfeldige steder. Har ingen forklaring på dette. Kan ha noe med maskinen det kjøres på? Da det kjøres andre ting samtidig.

Skulle gjerne lagt ved figurer for andre størrelser på input også, men fikk ikke formatert grafene sånn det er tydelig hva som skjer. Ingen av algoritmene ble ferdige med n = 100 000 og større innen 15 min.

Kjøring av program:

Har laget én fil per algoritme. Disse kjøres fra Oblig3.java ved å endre variablene alg1, alg2, alg3 og alg4. Innholdet i loopen lenger ned er ganske uoversiktlig, og kunne vært løst på en bedre måte enn å hardkode så mye.