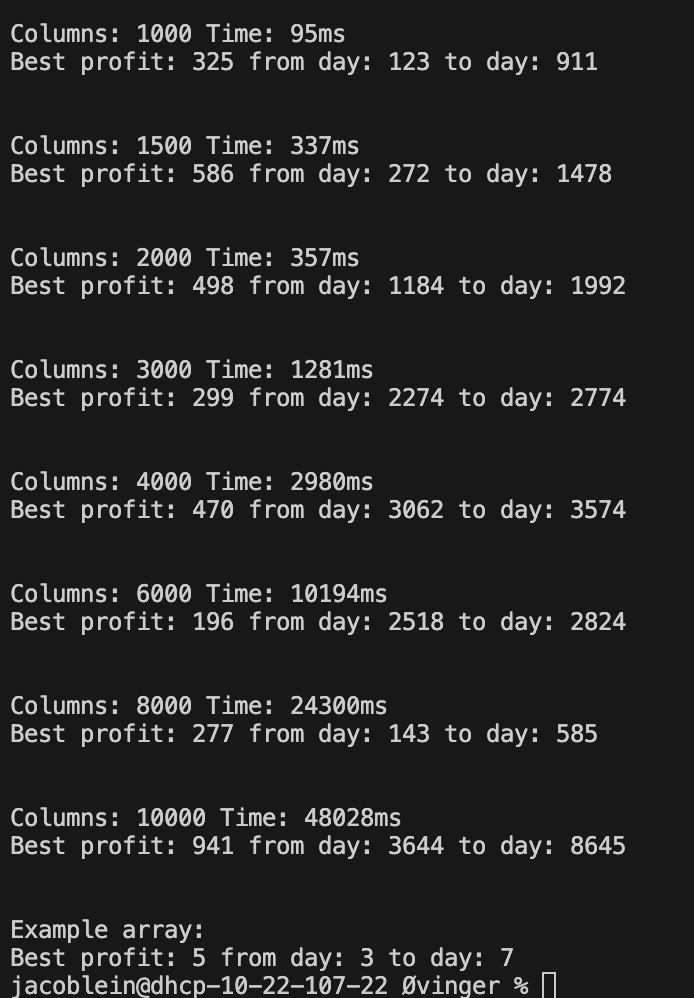
A screen shot of a computer code

Description automatically generatedA computer screen shot of a program code

Description automatically generatedØving 1 Algoritmer og datastrukturer Jacob Lein



* 1. Algoritmen søker etter beste mulige profitt for hver kjøpsdag, der hvis profitten er høyere enn tidligere profitter (maxProfit) så lagres datoene for dette kjøpet. Koden regner med at kjøp og salg blir gjort på slutten av dager. Altså at hvis beste kjøps- og salgdato er fra dag 5 til 7 så tilsier det at man kjøpte etter kursendring på kjøpsdato og solgte etter kursendring på salgsdato.
  2. Tidskompleksiteten til algoritmen vil være O(n^3) siden findBestProfit() metoden inneholder to for-løkker med n-antall iterasjoner, der innerste løkke kaller på metoden sum() som igjen inneholder en løkke med n-antall iterasjoner. De røde pilene i bildet peker til løkkene, mens metode-kallet er understreket. Det finnes nok algoritmer med A computer screen shot of a program code

     Description automatically generatedbedre effektivitet.
  3. Her vises det som kjøres i rekkefølge. Tiden startes rett før algoritmen finner beste kjøp og salg og alt printes etter det igjen. Til høyre er utskriften ved en kjøring. Siden tidskompleksiteten kan antas å kunne beskrives ved O(n^3) så kan man dobbeltsjekke dette med utskriften. 4000 dager tar 2980ms mens dobbelt mengde, 8000 dager, tar 24300ms.

Dette forholdet blir omtrent det samme som 8, som er det forventes ved dobling av data til en algoritme med kubisk kompleksitet. Hvis man øker datamengden med 2.5 altså til 10 000 dager, får man en responstid på 48028ms.

Selv om forholdene vist over har en differanse på 0.5 så stemmer det omtrentlig som forventet i forhold til O(n^3). Vi kan til slutt også se på doblingen av data mellom 3000 og 6000 dager. () Tidsmålingene ser ut til å stemme i forhold til analysen.