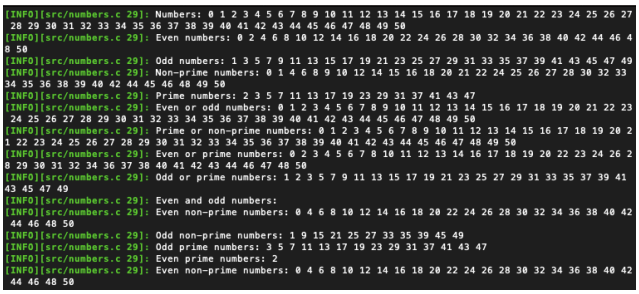
Status:



Har ikke klart og bli ferdig med oppgaven i tide!
Følgende fungerer:
Array-basert set-struct som fungerer på tallene, men ikke på strings.

Følgende fungerer ikke:

Linkedlist-basert set-struktur. Her har jeg prøvd og bruke node fra prekoden, blant annet for legge til elementer i midten av listen, men hjelpelærer anbefalte og heller bruke iteratoren. Denne er lakt på is til jeg får til det andre

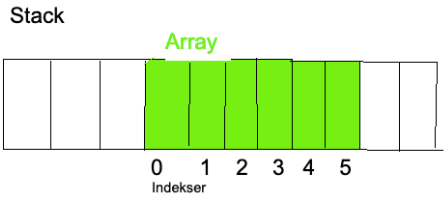
Har heller ikke gjort spamfilteret, men har forstått hvordan jeg skal gjøre dette.

Plan videre:
Kommer først til å prioritere og bli ferdig med den array-baserte implementasjon og lage spamfilter med denne. Deretter kan jeg fullføre den linked list-baserte og teste forskjellene.

Innledning:
I denne oppgaven

Implementasjon og problemer:
Som start har jeg implementert en array-basert datastruktur for mengder. Denne er utgangspunktet basert på en youtube-video[1], men har endret mye på kodet herfra i ettertid. I tillegg har jeg fylt på med resterende funksjoner for å tilfredstille set.h filen. Her er viktige punkter punkter fra implementasjonen slik jeg forstår den:

- Set -strukturen inneholder 3 relaterte variabler; Lengden på mengden(kardinalatet), en sammenlikningsfunksjon gitt i pre-koden, og et array med medlemmene i settet. Et array er en struktur hvor en lagrer elementer i påfølgende minneadresser. Dette gjør at elementer er lette og hente ut. Man kan angi indeksen til et element for å hente det ut, i motsetning til linkedlist hvor man må iterere fra start av listen for å finne et element. En ulempe er at om man skal endre på størrelsen til arrayet så må man lage et nytt array og kopiere alle elementene. Et annet problem er at om man vil sitte inn et element midt i arrayet, men samtidig ivareta rekkefølgen på andre elementer, så må man flytte veldig mange elementer nedover stacken. I en linked_list ville man kunne sette et element inn i midten mye enklerebare med enkel node-trikseri. Mer om det senere.



- En ineffektiv del av implementasjonen er hvordan elementer til. Hver gang et element blir lagt til utvider den arrayen med en adresse. Jeg har under arbeidet lært at dette koster veldig mange operasjoner, og at man heller burde doble størrelsen til arrayet hver gang det blir fult. Da slipper maskinen og lage et nytt array hver gang man legger til et element.

- En annet problem med måten elementer blir på er at de blir lagt inn sortert. Dette innebærer at mange elementer blir flyttet nedover stacken hver gang et element blir lagt til. Dette fører til mange operasjoner for datamaskinen og treg kode. Årsaken til at jeg implementerte det sånn er at mange av funksjonene i implementasjonen min baserer seg på at arrayene er sortert. (kommer tilbake til dette). For å finne riktig index til element som blir lagt til kjøres et binærsøk med cmpfunc fra pre-koden
  Løsning: Det er tenkelig at når arrayet blir brukt så vil mange elementer bli lagt til før andre funksjoner blir brukt. Derfor er en løsning og ha en variabel for om arrayet er sortert eller usotert. Deretter kan en sortere arrayet i det den blir brukt av andre funksjoner, men kun dersom den er usortert.
- Implementasjon av set-funksjonalitetene: union, snitt og differanse-funksjonene fungerer ved å iterere gjennom to sett parallelt og sammenligne verdier. Dette fungerer fordi arrayet i utgangspunktet er sortert.

Plan videre:

Kommer først til å prioritere og bli ferdig med den array-baserte implementasjon og lage spamfilter med denne. Deretter kan jeg fullføre den linked list-baserte og teste forskjellene.

Referanser:

[1] https://www.youtube.com/watch?v=RVqdK6WAjUI&t=308s