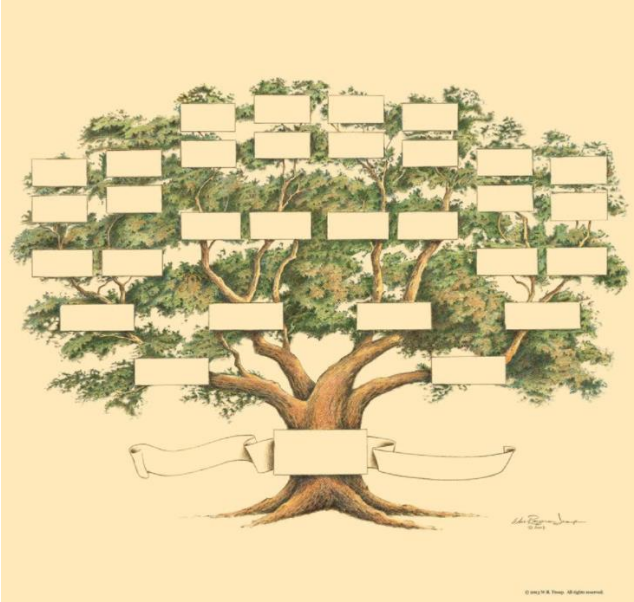
Mappe AIS1002 vår 2022

Prosjektoppgave: Forfedrediagram (ancestor chart)

Kravspesifikasjon:

Et program skal designes som kan brukes til å sette sammen et digitalt forfedrediagram.

Et forfedrediagram viser forfedrene til en *person*. Diagrammet starter fra bunnen og oppover til to foreldre, fire besteforeldre og så videre. Denne strukturen er den samme for hver person.



En *directed acyclic graph* er en naturlig datastruktur for å modellere dette.

Det skal være mulig å traversere grafen gjennom dybde-først-traversering (depth-first-traversal). En god løsning vil også ha mulighet for bredde-først-traversering (breadth-first traversal).

En person bør ha **minst** følgende attributter:

Fornavn, etternavn, evt. fødselsår, evt. dødsår, kjønn.

Legg til andre om dere føler det kan løfte bruken av applikasjonen.

Gruppe/samarbeid

Det blir lagt opp til at man skal samarbeide i grupper på to. Det er viktig at man raskt finner ut om man er på en gruppe der begge er aktive og samarbeidsvillige.

En del av vurderingsgrunnlaget blir om man kan dokumentere godt samarbeid bl.a ved bruk av Git og prosjekthjelpemidlene som finnes på GitHub.com slik som issues og pull requests. Lær av hverandre!

Alenegruppe kan vurderes. Git og prosjekthjelpemiddel skal fortsatt brukes underveis.

Brukergrensesnitt

Det skal lages brukergrensesnitt som kan brukes via kommandolinjen.

Som **minimum** skal det være mulig å:

- Legge til personer i grafen (minst 3 lag).
- Printe informasjon om personer i grafen på forskjellig hvis:
    - Her kan man demonstrere traversering, sortering, etc.

For å løfte oppgaven bør dere legge til flere menyvalg. Noen eksempel:

- Hente ut et utvalg av personer basert på en eller flere søketermer.
- Modifisering av eksisterende person.
- Fjerne personer i grafen.
- Lagring og gjenoppretting av datastrukturen til/fra fil.
- Etc.

Brukere kan gjøre feil, og dette må applikasjonen ta hensyn til og gi tilbakemelding på helst uten at programmet krasjer.

Logikken trenger ikke være 100%. Man kan f.eks. gå ut ifra at alle navn er unike, noe som kan gjøre søking lettere. Slike forenklinger skal dokumenteres i koden.

Enhetstester

Prosjektet skal inneholde enhetstester (unit-test) som kan teste ulike deler av koden deres. Automatisk testing ved bruk av github workflows vil trekke positivt opp og gjør det mulig å sjekke kryssplatforms-kompabilitet.

Bruk av tredjeparts bibliotek

Dere står fritt til å bruke tredjeparts bibliotek, men bruk i så fall header-only bibliotek slik at det blir enklest mulig for utenforstående å kompilere koden dere skriver. Dette gjelder også enhetstesting rammeverk. Catch2 demonstrert i klassen er header-only og kan fint brukes.

Kildehenvisninger

Dersom du benytter deg av kode som du selv ikke har skrevet, skal dette dokumenteres i koden med henvisning til kilden, og en begrunnelse for hvorfor og hvordan dere mener benyttet kode bidrar til deres løsning.

Vurderingskriterier

Din besvarelse vil bli vurdert basert på følgende kriterier:

- Om dere har fulgt (og holdt deg til) kravspesifikasjonen
- Om du viser god forståelse for grunnleggende prinsipper i programmering som variabler, datatyper, metoder, betingelser, løkker osv.
- Om du har fulgt prinsippene for god design (abstraction, encapsulation, coupling, cohesion, responsibility driven design osv.).
- Om du har valgt gode, selvforklarende navn på klasser, metoder og variabler/felt/parametere.
- Bruk av konsekvent og rydding kode design (formatering, konvensjoner etc.)
- Om du har dokumentert koden din godt.
- Om du har god og fornuftig samhandling med bruker (et godt brukergrensesnitt).
- Om du har skrevet gode og dekkende enhetstester.
- Om programmet kan håndtere feil (ugyldig input fra bruker osv.).
- Evne til å reflektere over løsningen.

Besvare og levere

Hver gruppe skal levere en versjon av prosjektet sitt i Blackboard (kun kildekode) **innen torsdag 7. april**. Dette for at det skal være mulig å få *tilbakemelding* på arbeidet underveis. Dette teller altså som en nødvendig del av mappen (gis ikke direkte karakter) og det vil ikke gis utsettelse på denne.

Om ikke annet blir oppgitt, så skal den endelige mappen leveres i Inspera. Denne skal bestå av:

- Prosjektfilene dere tidligere leverte (se ovenfor) på blackboard i zippet mappe (.zip).
- Endelige prosjektfiler i zippet mappe (.zip). Denne skal IKKE inneholde auto-genererte byggfiler.
    - Det forventes at prosjektet har en beskrivende README fil som kort beskriver prosjektet samt et klassediagram i henhold til UML-standarden over løsningen dere har implementert. Det er tilstrekkelig at dere tegner klassene kun med navn (ikke felt og metoder). Få med hvilke avhengigheter (avhengigheter, assosiasjoner osv.) som finnes mellom klassene. Få med kandidatnummer i README.
- Et kort **individuelt** dokument (maks 1 side) som oppsummerer:
    - Hva **du** har bidratt med, hvordan gruppen har samarbeidet og hvordan versjoneringsverktøy og prosjektsplanleggningsverktøy har blitt brukt gjennom prosjektet.
    - Reflekter over hvordan dere har valgt å løse prosjektet. Hva var den største utfordringen? Hva vil du fremheve som du synes dere har løst på en god måte (hva har dere gjort for å sikre høy kvalitet og en robust løsning)? Basert på hvordan dere har valgt å løse prosjektet i denne oppgaven, hvordan vil du vurdere løsningen din i henhold til design-prinsippene coupling, cohesion og responsibility driven design? Er det noen av klassene dere f.eks. tenker kunne vært bedre designet iht prinsippet om cohesion? Dersom du hadde mer tid for å løse oppgave, hvilken deler av programmet vil du ha endret (refactoring), og hvorfor?

Endelig karakter settes basert på en helhetlig vurdering av mappen.

Info om endelig leveringsdato kommer (engang i mai).