TDT4102 - Procedural and Object-Oriented Programming

Uke 9 2024

0.1 Generell Intro

Les gjennom oppgavetekstene nøye. Noen av oppgavene har lengre tekst, men dette er for å gi kontekst, introduksjon og eksempler til oppgavene.

Når det står "implementer", "definer" eller "lag" skal du skrive en fungerende implementasjon: hvis det handler om en funksjon skal du skrive deklarasjonen med returtype og parametertype(r) og hele funksjons-kroppen.

Når det står "deklarer" er vi kun interessert i funksjons- eller klassedeklarasjonen. Typisk vil dette være deklarasjoner du vanligvis finner i header-filer.

Hvis det står "forklar" står du fritt i hvordan du svarer, men bruk enkle kodelinjer og/eller korte tekstforklaringer og vær kort og presis.

Dersom du mener at opplysninger mangler i en oppgaveformulering, gjør kort rede for de antagelser og forutsetninger du finner nødvendig.

Hvis du oppdager at informasjon er feil eller mangler i en oppgave, forklar hvilke antagelser du må gjøre, og hvordan du tolker oppgaven. Legg forklaringene til som kommentarlinjer i koden din.

Hver enkelt oppgave er ikke ment å være mer omfattende enn det som er beskrevet. Noen oppgaver fokuserer bare på enkeltfunksjoner og da er det utelukkende denne funksjonen som er tema. Andre oppgaver er "oppskriftsbasert" og vi spør etter funksjoner som utgjør deler i et program, eller forskjellige deler av en eller flere klasser. Du kan velge selv om du vil løse dette trinnvis, eller om du vil lage en samlet implementasjon, men sørg for at det går tydelig frem hvilke spørsmål du har svart på hvor i koden din.

Husk at funksjonene du lager i en deloppgave ofte er ment å skulle brukes i andre deloppgaver. Selv om du står helt fast på en deloppgave bør du likevel prøve å løse alle eller noen av de etterfølgende oppgavene ved å anta at funksjoner fra tidligere deloppgave er riktig implementert.

All kode skal være i C++. Du kan bruke VS-code, eller andre utviklingsmiljøer om du ønsker. Oppgaven krever ikke kjennskap til andre klasser og funksjoner enn de du har blitt godt kjent med i øvingsopplegget, eller som står beskrevet i læreboka. Hvis du mener det er nødvendig kan du fritt inkludere flere klasser og funksjoner.

"include"-setninger og organisering av koden i flere filer er helt nødvendig i siste del av denne eksamen.

Hele oppgavesettet er arbeidskrevende, og det er ikke forventet at alle skal klare alt. Tenk strategisk i forhold til ditt nivå og løs oppgavene som er enklest for deg først.

Deloppgavene i "tematiske" oppgaver er organisert i en logisk rekkefølge, men det betyr ikke at det er en økende vanskelighetsgrad utover i deloppgavene. Hoveddelene av eksamensoppgaven teller i utgangspunktet med den andelen som er angitt i prosent, men den prosentvise uttellingen for hver oppgave kan likevel bli justert ved sensur, basert på hvordan oppgavene har fungert. De enkelte deloppgaver kan også bli tillagt forskjellig vekt.

Før den detaljerte evalueringen av eksamen din, vil vi foreta automatisk testing og plagiatkontroll av all koden du har levert. Basert på resultatene kan det hende vi ber deg om en forklaring, og dette kan

påvirke eksamensresultatet ditt. Du må huske å legge til kommentarlinjer i koden din, siden det også kan hjelpe oss å forstå koden din bedre.

Helt på slutten av eksamen ber vi deg laste opp en .zip-fil som inneholder alle de komplette kodefilene tilhørende den siste (største) delen av eksamen. Koden din trenger IKKE å være feilfri eller å kjøre uten problemer for at du skal få bestått, men det er en fordel å sende inn kjørende kode. Vi vil evaluere hvert spørsmål i detalj for å se hvor mye du har lært av dette kurset. For å få bestått på denne eksamen er det HELT AVGJØRENDE AT DU LASTER OPP ZIP-FILEN, innen de 4 timene til rådighet. Prøv å last opp filen en gang midt i eksamenstiden, for å se at du klarer det, og hvor lang tid du bruker på det.

0.2 Eksamen - Step by Step

Denne listen vil lede deg trinn for trinn igjennom hva du skal gjøre på denne eksamen.

1. Les nøye igjennom disse **introduksjons-sidene**.
2. **Last ned** medfølgende .zip-fil med en gang eksamen starter. I .zip-filen finner du en mappe med .cpp- og .h-filer, og en undermappe med datafiler.
3. Les **forklaringen** på problemet gitt i begynnelsen av hver seksjon.
4. I .h og .cpp-filene finner du både **grunnleggende kode** og **oppgavene** du må fullføre. Oppgavene står også i del 3 på Inspera.
5. Du kan bruke VS Code (eller et hvilket som helst annet utviklingsmiljø du foretrekker) til å åpne og jobbe med den oppgitte koden (som i Øving 0).
6. Du kan bruke boken eller andre online/offline ressurser, men du kan **IKKE** samarbeide med andre på noen måte, eller direkte kopiere og lime inn online kode som om det er din egen.
7. Etter å ha fullført hver enkelt kodeoppgave, bør du sende inn svaret ditt gjennom Inspera.
    - Last opp all den komplette koden som en .zip-fil. Ikke endre den opprinnelige mappestrukturen. For å få bestått på denne eksamen er det HELT AVGJØRENDE AT DU LASTER OPP ZIP-FILEN, innen de 4 timene til rådighet.
    - Det er mulig å oppdatere både enkelt-svarene og filopplastningen flere ganger, i tilfelle du retter på noe etter første innlevering.
    - Prøv å last opp filen en gang midt i eksamenstiden, for å se at du klarer det, og hvor lang tid du bruker på det.
8. Send inn koden din selv om den ikke kan kompileres og / eller ikke fungerer riktig. Fungerende kode er **IKKE** et krav for at du skal stå, men det er en fordel.

0.3 Nedlasting av start-fil

Vi gir dere en .zip-fil med noen forhånds-programmerte deler av programmene og en datafil som du trenger å bruke i programmet ditt.

- Last ned og lagre .zip-filen (lenken er på Inspera). Gjør dette med en gang eksamen starter... i tilfelle noe er galt, eller internett detter ut av og til underveis.
- Husk å lagre den på et sted på datamaskinen du husker og kan finne igjen.
- Pakk ut (unzip) filen. Du finner de forskjellige oppgavene i forskjellige mapper, inkludert oppgaveforklaringer som kommentarer. Du kan også finne disse forklaringene i Inspera, som forskjellige del-spørsmål.
- Begynn å jobbe med oppgavene i VS Code (eller et hvilket som helst annet utviklingsmiljø du foretrekker). Vi forventer at du vet hvordan du sammenstiller og kjører kode, slik det ble forklart i øving 0 på begynnelsen av semesteret.

- Filene vi leverer ut kompilerer til et kjørende program, men du må selv skrive resten av koden for alle oppgavene og prøve å få hver programmet til å kjøre som et eget prosjekt. Det er ikke et krav at all den innleverte koden kan kjøres, men det er en fordel.
- Etter at du er ferdig med kodingen av alle del-spørsmål, må du laste opp alle kodefilene dine, etter at de på nytt er pakket sammen til en lignende .zip-fil som den du begynte med. Ikke endre noe på de opprinnelige mappe/fil-navnene før du zipper filen og laster den opp til Inspera igjen. Last gjerne opp på nytt hver time!