

IDAT1001 Programmering 1 – H2020 – 4t- Hjemmeeksamen

Løsningsforslag/sensorveiledning

Fra emnebeskrivelsen:

Kunnskap. Kandidaten skal:

- [1] kunne oppdatere sin kunnskap innen fagområdet
- [2] kunne forklare hva et program er
- [3] har kunnskap om enkle prinsipper innen objektorientert programmering som innkapsling, modularisering og samhandlende objekter
- [4] kunne forklare hva som menes med en lagdelt arkitektur og hvorfor det er viktig i programvaredesign
- [5] kunne forklare prinsipper for god kodedesign (kobling/ kohesjon), kodekvalitet og god kodelstil
- [6] kunne forklare typiske disipliner i programvareutvikling (livssyklusen til et program) på et overordnet nivå
- [7] kunne modellere klassediagram som består av flere klasser med aggregering, komposisjon gitt en problemstilling
- [8] kunne gjøre rede for grunnleggende diagrammer i UML (aktivitetsdiagram, klassediagram, sekvensdiagram, pakkeprogrammer)
- [9] har kunnskap om de viktigste komponentene i datamaskinen og hvordan de samvirker
- [10] kunne forklare representasjon av de viktigste datatypene og hvordan datamaskinen prosesserer dem
- [11] kunne forklare hvordan hovedkomponentene samspiller og hvordan de konfigureres for best mulig pris/ytelses-forhold (systemarkitektur).
- [12] har kunnskap om oppbygging av moderne prosessorer: pipeline, dynamisk utføring, flerkjerneprosessorer, parallellprosessering på ulike nivå (prosessarkitektur).

Datamaskinarkitektur

Ferdigheter. Kandidaten skal:

- [13] - kunne sette opp programmiljø for å utvikle, teste og kjøre objektorienterte programmer (D)
- [14] - kunne lage strukturert, oversiktlig og godt dokumentert programkode basert på prinsipper for god kodedesign
- [15] - kunne anvende klasser, kontrollstrukturer og samarbeid mellom objekter
- [16] - kunne anvende klasser fra biblioteker og finne fram i API-dokumentasjon (D)
- [17] - kunne anvende grunnleggende objektorientert tankegang til å analysere og løse enklere problemer

Generell kompetanse. Kandidaten skal:

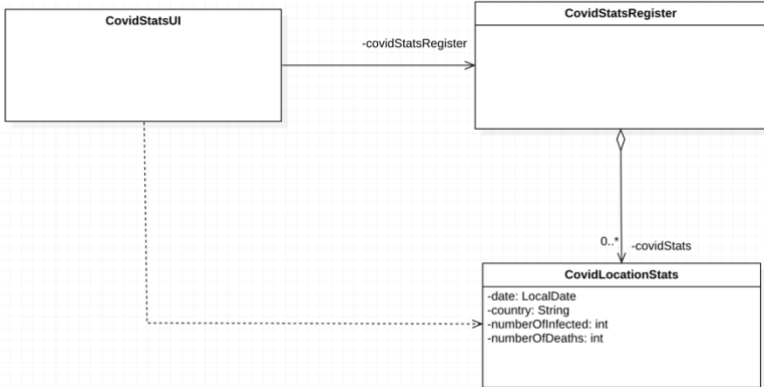
- [18] - kan forklare og gjøre bruk av sin kunnskap både innen hvert enkelt tema i faget og på tvers av temaene.

Java/Programmeringsspråk

- [a] Variabler og datatyper
- [b] Klasser og Objekter
- [c] Metoder med og uten parametre og retur
- [d] Betingelser (IF)
- [e] Løkker (for, for-each og while)

[f] Samlinger (ArrayList, HashMap og HashSet)

[g] Iteratorer

				Poeng
Oppg	Løsningsforslag	Vurderingskriterier	LUB nr	
	OPPGAVE 1			[11p]
1 a)	Klassediagram  <pre> classDiagram class CovidStatsUI class CovidStatsRegister class CovidLocationStats { -date: LocalDate -country: String -numberOfInfected: int -numberOfDeaths: int } CovidStatsUI --> CovidStatsRegister : -covidStatsRegister CovidStatsRegister o-- CovidLocationStats : -covidStats CovidStatsUI ..> CovidLocationStats </pre>	Tester kandidaten i: <ul style="list-style-type: none"> UML Klassediagram Vurder om kandidaten har: Laget en håndtegnet <i>skisse</i> . Ikke viktig å vise <i>felt</i> og <i>metoder</i> . Her er fokus å identifisere <i>relasjonen</i> mellom klassene og fornuftige klasser. Dette er en skisse tidlig i besvarelsen før detaljer er på plass.	3,7, 8, 17	4
		Kandidaten har tegnet klassene, og vist avhengighet (stiplede piler)		(3)
		Tegnet assosiasjoner gjerne med Kardinalitet/Multiplicity og type relasjon (komposisjon, består av)		(1)

				Poeng
Oppg	Løsningsforslag	Vurderingskriterier	LUB nr	
2 c)	<p>Metode for å sjekke om en gitt dato er tidligere enn en annen dato er isBefore med følgende signatur:</p> <pre>boolean isBefore(ChronoLocalDate other)</pre>	<p>Tester kandidaten i:</p> <ul style="list-style-type: none"> • Å finne frem i dokumentasjon • Å forstå hvordan metoder er dokumentert i Javadoc • Hva som menes med signatur <p>Vurder om kandidaten har:</p> <ul style="list-style-type: none"> • Funnet riktig metode. isAfter() kan til nød godkjennes (kommer an på hvordan man sammenligner), men gi trekk. • Har skrevet fullstendig signatur (retur type, navn, parametre). 		<p>4</p> <p>(2)</p> <p>(2)</p>
2 d)	<p>Metoden <i>parse(CharSequence text)</i> tar en tekst som input parameter og prøver å tolke teksten som en dato. Teksten må være på formen "2007-12-03" el.l. (avhenger av nasjonalitet på operativsystem)</p>	<p>Tester kandidaten i:</p> <ul style="list-style-type: none"> • Å finne frem i dokumentasjon • Å forstå dokumentasjon til en gitt metode. <p>Vurder om kandidaten har:</p> <ul style="list-style-type: none"> • Forstått hva metoden gjør. Svar på formen "gjør om/konverterer" en dato gitt som text/streng til et object/instans av LocalDate" er det vi ser etter. • Nevner spesifikt at metoden returnerer et objekt/en instans av LocalDate 		<p>4</p> <p>(3)</p> <p>(1)</p>
OPPGAVE 3				[40p]

				Poeng
Oppg	Løsningsforslag	Vurderingskriterier	LUB nr	
3 a)	<p>Klassen for en COVID registrering: CovidLocationsStats (el.l.)</p> <p>Forslag til felt og datatyper:</p> <pre>private LocalDate date; private String country; private int numberOfInfected; private int numberOfDeaths;</pre> <p>Det skal lages aksessormetoder/tilgangsmetoder til samtlige felt.</p> <p>I utgangspunktet skal det her ikke være nødvendig med mutatormetoder. Får poeng dersom ingen muttormetoder lages ;-)</p>	<p>Tester kandidaten i:</p> <ul style="list-style-type: none"> • Å lage en klasse • Aksessor/tilgangs og mutator metoder • Konstruktør <p>Vurdere om kandidaten har:</p> <ul style="list-style-type: none"> • Valgt fornuftige datatyper for feltene. (2) • Valgt gode fornuftige engelske navn for felt (2) • Implementert en konstruktør, som validerer en eller flere av parametrene (2) • Implementert aksessormetoder/tilgangsmetoder til alle felt (2) • Sett at det ikke er nødvendig med mutatormetoder. (1) <p>Klassen og metodene er dokumentert iht JavaDoc standard (3)</p>		10

				Poeng
Oppg	Løsningsforslag	Vurderingskriterier	LUB nr	
3 b)	<p>Følgende parametre i konstruktøren (og eventuelle set-metoder) bør/skal testes for gyldighet:</p> <ul style="list-style-type: none"> • date – bør ikke være <i>null</i> • country – bør ikke være <i>null</i> • numberOfInfected – bør ikke være <i>negativ</i> • numberOfDeaths – bør ikke være <i>negativ</i> <p>Tanker om hvordan håndtere:</p> <ul style="list-style-type: none"> • sjekke verdier med <i>if-setning</i> • når feil verdi oppdages, velge en fornuftig strategi. Siden kandidatene ikke har lært unntakshåndtering ennå, er en fornuftig strategi å gi feltene en default/forhåndsbestemt gyldig verdi. F.eks. dersom numberOfInfected er negativ, settes den til 0. Dersom <i>country</i> er null, sett til en gyldig streng som f.eks. "INVALID COUNTRY" el.l. 	<p>Tester kandidaten i</p> <ul style="list-style-type: none"> • Å lage robust kode ved å være bevist hva som er gyldige verdier til parametere. • Hvordan håndtere feil verdier i parametere på en trygg måte. <p>Vurder om kandidaten har:</p> <ul style="list-style-type: none"> • Har påpekt minst 2 parametere som bør sjekkes, og hva som er ugyldige verdier • Har gitt en forklart greit hvordan håndtere feil verdi. 		<p>5</p> <p>(3)</p> <p>(2)</p>

3 c)	<p>En forkortet utgave av klassen:</p> <pre>import java.time.LocalDate; import java.util.ArrayList; import java.util.Iterator; import java.util.List; public class CovidStatsRegister { // BEGRUNNELSE: Jeg har valgt å bruke ArrayList // siden vi ikke har noen unik ID knyttet til // hver COVID registrering, så da er HashMap // ikke nyttig. Jeg kunne også valgt HashSet private ArrayList<CovidLocationStats> covidStats; public CovidStatsRegister() { this.covidStats = new ArrayList<>(); } public void addCovidStats(CovidLocationStats covidLocationStats) { ... } public CovidLocationStats findCovidStatsByDate(LocalDate date) { CovidLocationStats foundCovidLocStats = null; ... return foundCovidLocStats; } public ArrayList<CovidLocationStats> findCovidStatsAfterDate(LocalDate date) { ArrayList<CovidLocationStats> foundCovidStats = new ArrayList<>(); ... return foundCovidStats; } }</pre>	<p>Tester kandidaten i:</p> <ul style="list-style-type: none"> • Bruk av tabeller/samlinger • Løkker (for/while) ifm samlinger • Betingelse (if-setning) • Enkel aritmetikk/enkle beregninger <p>Vurdere om kandidaten har:</p> <ul style="list-style-type: none"> • Implementert samling som ArrayList, HashSet, HashMap el.l.. (3) • Begrunnet valget av samling/tabell (2p) (Siden vi ikke har en unik ID for hver registrering, er HashSet og ArrayList gode valg.) (2) • God dokumentasjon av klassen og metodene (JavaDoc) (3) <p>Implementert metoder for å:</p> <ul style="list-style-type: none"> • legge til COVID-registrering. Parameter skal være objekt av CovidLocationStats, ikke data for å lage en COVID-registrering (loose coupling). (3) • Finne en registrering på en gitt dato. Må fremgå av dok hva som blir returnert om ingen COVID-reg blir funnet. Ekstra poeng for å bruke while og IKKE for-each. (4) • Finne samtlige COVID-registreringer etter en gitt dato. Ekstra poeng for å bruke for-each og IKKE while. (1) 	14, 15, 16	25
------	--	---	------------	----

a)	<p>Brukeren må kunne</p> <ul style="list-style-type: none"> registrere/legge inn COVID-19 tilfeller, hver registrering er pr dag. skrive ut liste over alle registreringer søke etter en registrering basert på dato søke etter registreringer etter en gitt dato regne ut samlet antall døde for et gitt land <p>Forslag til løsning: Se kode i prosjekt.</p>	<p>Tester kandidaten i:</p> <ul style="list-style-type: none"> Lage et brukervennlig tekstbasert brukergrensesnitt. Brukerinteraksjon. Å hente data fra bruker og presentere data til bruker. Validere input fra bruker. Håndtere feil input fra bruker. Lage en komplett applikasjon fra presentasjon via forretningslogikk til data (entity) Kan å tolke en kravspesifikasjon og omsette i ferdig løsning. Koble brukergrensesnitt mot forretningslogikk (lagdelt arkitektur) <p>Vurdere om kandidaten har:</p> <p>Implementert funksjonalitetskrav:</p> <ul style="list-style-type: none"> Implementert å registrere et COVID-19 tilfelle. Implementert utskrift av alle registreringer Implementert søk etter registrering basert på en gitt dato. Bør gjøre bruk av <code>LocalDate.parse()</code>-metoden. God tilbakemelding til bruker dersom ingen tilfeller blir funnet. Implementert søk etter tilfeller <i>etter</i> en gitt dato. Bør bruke <code>isBefore()/isAfter()</code>-metode i <code>LocalDate</code>. God tilbakemelding til bruker dersom ingen tilfeller blir funnet. 	3, 4, 5	<p>25</p> <p>(4)</p> <p>(3)</p> <p>(4)</p> <p>(4)</p>
----	---	---	---------	--

				Poeng
Oppg	Løsningsforslag	Vurderingskriterier	LUB nr	
		<ul style="list-style-type: none"> Implementert utregning av antall døde i et gitt land. God håndtering dersom oppgitt land ikke har noen registreringer. <p>Øvrig:</p> <ul style="list-style-type: none"> Laget egen metode for å skrive ut detaljer om ett COVID-tilfelle (cohesion) Laget egne metoder for hver funksjon (ikke skrevet kode direkte i switch-case) Godt dokumentert kode (samtlige metoder har Javadoc) 		<p>(2)</p> <p>(2)</p> <p>(3)</p> <p>(3)</p>

				Poeng
Oppg	Løsningsforslag	Vurderingskriterier	LUB nr	
b)	<p>Fra oppgaveteksten:</p> <p>Med bakgrunn i din endelige løsning av kravspesifikasjonen, hvordan vil du si at designet og implementasjonen din er utført i henhold til design-prinsippene kobling (eng: coupling), og samstemthet (eng: cohesion) ? Gi gjerne konkrete eksempler fra din egen kode.</p> <p>Eksempel på mulig besvarelse:</p> <p>Coupling:</p> <ul style="list-style-type: none"> • Jeg har kun private felt i mine klasser. • Jeg har designet løsningen slik at klassene trenger å vite minst mulig om hvordan klassene internt er strukturert/oppbygget. For eksempel har jeg en metode som returnerer en iterator for å gi andre objekter tilgang til å iterere over alle COVID-registreringene, istedenfor å returnere ArrayList. På denne måten står jeg fritt til å endre fra ArrayList til f.eks. HashSet internt i metoden, uten at dette får følger for resten av koden. <p>Cohesion: Alle klassene mine har en klart definert rolle/ansvar, og de fleste metodene mine utfører kun en oppgave.</p>	<p>Tester kandidaten i:</p> <ul style="list-style-type: none"> • Forståelsen for god design, samt evnen til å implementere godt designet kode. • Forståelsen av designprinsippene Coupling og Cohesion • Evne til å reflektere over egen løsning i forhold til god design <p>Vurdere om kandidaten har:</p> <ul style="list-style-type: none"> • Vist at han/hun har forstått Coupling (4) • Vist at han/hun har forstått Cohesion (4) • Har gitt konkrete eksempler i egen kode som understøtter at god design er brukt (2) 		10
			SUM	100