

Innhold

Innhold	0
Del 1: Flervalgsoppgaver	1
1.1 Visualisering	1
1.2 Diagram	1
1.3 Det tolvte smidige prinsipp	1
1.4 Trusselmodellering	2
1.5 Android - layout	2
1.6 MVVM	2
1.7 Kotlin.fold	3
1.8 Kotlin operasjoner på list	3
Del 2: Spørsmål fra pensum	4
2.1 Diverse kort svar fra pensum (10p)	4
2.2 Git (3p)	5
2.3 finnSummer (12p)	5
2.4 Refleksjon - teamarbeid (15p)	6
Del 3: Caseoppgave	8
3.1 Prosjektoppstart (14p)	8
3.2 Funksjonelle og ikke-funksjonelle krav (9p)	8
3.3 Definer en klasse i Kotlin slik at JSON-objektet kan deserialiseres. (4p)	9
3.4 App som fryser (8p)	9
3.5 Refleksjon UU (10p)	9

Del 1: Flervalgsoppgaver

Denne delen består av **åtte** deloppgaver og gir maks 17 poeng. Se Inspira for hva som er riktig svar.

1.1 Visualisering

Visualisering, (dvs. bruk av "tavle" eller "board") er viktig i...

Velg ett alternativ:

- ☒ ...både Scrum og Kanban
- ☐ ...Kanban, men ikke Scrum
- ☐ ...verken Scrum eller Kanban
- ☐ ...Scrum, men ikke Kanban



1.2 Diagram

Du jobber med et system for en online bestillingsprosess i en restaurant. Programmet skal kunne håndtere følgende aktiviteter: kunder kan velge retter fra menyen, legge dem i en handlekurv, betale for bestillingen og få en bekreftelse. Hver av disse aktivitetene kan skjule komplekse underliggende prosesser og tilstander, og det er viktig å illustrere flyten av informasjon og interaksjoner gjennom hele prosessen.

Hvilket av følgende diagram egner seg best for å illustrere denne prosessen?

Velg ett alternativ:

- ☒ Aktivitetsdiagram
- ☐ Klassediagram
- ☐ Tilstandsdiagram
- ☐ Sekvensdiagram



1.3 Det tolvte smidige prinsipp

Det 12. smidige prinsippet lyder som følger:

“Med jevne mellomrom reflekterer teamet over hvordan det kan bli mer effektivt og justerer adferden sin deretter.”

Basert på dette, hva skal teamet gjøre regelmessig?

Velg ett alternativ:

- ☐ Feire oppnådde milepæler
- ☐ Utføre refaktoring av kode
- ☒ Avholde retrospektiv-møter
- ☐ Jobbe saktere for å sende signaler til ledelsen



1.4 Trusselmodellering

Hvilket av følgende alternativer er **ikke** et mål for trusselmodellering?

Velg ett alternativ:

- ☐ Identifisere nøkkelsvakheter ved designet av applikasjonen
- ☐ Forbedre designet av sikkerheten
- ☐ Identifisere relevante trusler for ditt applikasjonsscenario
- ☒ Inkorporere trusselpoker i Scrum-sprinten



1.5 Android - layout

Du ønsker å vise en **vertikal liste** med **fire elementer**.

Hvilken av følgende komponenter bør du benytte deg av?

Velg ett alternativ:

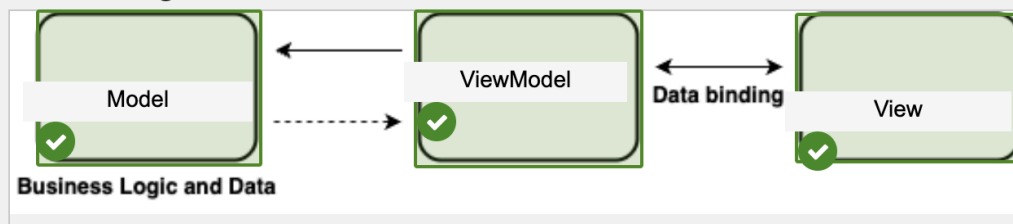
- ☐ LazyRow
- ☐ LazyColumn
- ☐ Row
- ☒ Column



1.6 MVVM

Dra riktig del av MVVM til riktig boks.

Dra boksene til sine riktige felt:



1.7 Kotlin.fold

```
fun main() {  
    val list = listOf(1, 2, 3, 4, 5)  
    val result = list.filter { it % 2 == 0 }.fold(0) { acc, x -> acc + x }  
    println(result)  
}
```

Hva blir skrevet ut i terminalen?

Skriv ditt svar her: 

1.8 Kotlin operasjoner på list

Gitt følgende liste i Kotlin:

```
val list: List<Int> = listOf(1, 2, 3, 4, 5, 6, 7, 8, 9)
```

Hvilke av følgende uttrykk skriver ut **2** i terminalen?

3p for riktige alternativer svar, 0p for feil.

Velg ett eller flere alternativer

☐ list.map { it * 2 }.count { it == 4 }

☒ list.map { it / 3 }.count { it == 0 } 

☐ list.count { it > 5 }

☐ list.filter { it % 2 == 0 }.count()

☒ list.count { it < 3 } 

Del 2: Spørsmål fra pensum

Denne delen består av fire deloppgaver og dekker diverse deler fra pensum. På denne delen er det mulig å oppnå totalt 40 poeng.

2.1 Diverse kort svar fra pensum (10p)

Denne oppgaven består av små oppgaver der kandidaten **maks skal bruke fire setninger** på å besvare hver av dem. Dersom kandidatene bruker mer enn fire setninger, eller svarer feil gis det 0 poeng. Her skal det mye til å få 1 av 2, siden som regel har kandidaten riktig eller feil - lite i mellom.

a) Forklar forskjellen på smidig og fossefall innen systemutvikling.

Eksempel på svar:

Smidig fokuserer på iterativ/inkrementell utvikling, der teamet jobber tett sammen for å kontinuerlig forbedre produktet. Fossefall har en mer lineær tilnærming med ulike faser, der hver fase må ferdigstilles før en går videre til neste.

b) Hva er teknisk gjeld?

Eksempel på svar:

Teknisk gjeld er kompromissene og snarveiene som tas for å oppnå raske resultater/gevinst. Dette kan resultere i økte kostnader og kompleksitet senere, når de ufullstendige eller suboptimale løsningene må rettes opp i.

c) Hva er trusselmodellering?

Eksempel på svar:

Trusselmodellering involverer identifisering og analyse av angrepsscenarioer for nye funksjoner i smidig utvikling, med mål om å redusere sårbarheter i hver sprint.

d) Forklar begrepet "state hoisting".

Eksempel på svar:

State hoisting handler om å flytte tilstand eller data opp til en forelderkomponent, slik at den styrer tilstanden og sender den ned til barnekomponentene. Dette gjør komponentene lettere å gjenbruke og å teste.

e) Hva menes med "Sprint 0"?

Eksempel på svar:

Sprint 0 er en innledende fase i prosjektet der teamet samarbeider for å identifisere problemer, definere mål og planlegge hvordan arbeidet skal utføres, samtidig som de bygger felles forståelse og kommunikasjonslinjer før den faktiske utviklingen starter.

2.2 Git (3p)

Nevn seks ulike git-kommandoer, og for hver av dem beskriv kort hva de gjør.

Her gis det 0,5p per riktige git-kommando **med** tilhørende beskrivelse. Poenggivning som følger:

0 og 1 kommando m/beskrivelse	0 poeng
2 og 3 kommandoer m/beskrivelse	1 poeng
4 og 5 kommandoer m/beskrivelse	2 poeng
6 kommandoer m/beskrivelse	3 poeng

Alle kommandoer gir poeng så lenge de er beskrevet riktig. Det gir ikke poeng å oppgi den samme kommandoen flere ganger.

2.3 finnSummer (12p)

a) Implementer prosedyren finnSummer(L, x) ved bruk av Kotlin (...) - 8p
se oppgavetekst for fullstendig beskrivelse

- Løsningsforslag #1: <https://pl.kotl.in/tovAkomA6>
- Løsningsforslag #2: <https://pl.kotl.in/KchdZslRe>
- Løsningsforslag #3: <https://pl.kotl.in/5NOtxMm2b> (ikke full pott)

Det er mulig å få full pott på denne oppgaven selv om den ikke er løst på måte #1 eller #2 over. Det bedømmes på:

- Riktig deklarasjon av funksjon
- Riktig logikk, dvs. at funksjonen gjør det den skal
- “Kotlin”-idiomati som gjennomgått i forelesning

Det er mulig å få inntil 2 poeng på hele deloppgaven dersom metoden er implementert som pseudokode / i et annet programmeringsspråk.

Rekkefølgen det skrives ut på har ingenting å si, så lenge den ikke blir repetert.

b) Skriv en enhetstest til finnSummer ved bruk av Kotlin (...) - 4p
se oppgavetekst for fullstendig beskrivelse

I denne oppgaven ble det gitt en beskjed under eksamen grunnet utydelig oppgavetekst. Kandidatene kan derfor anta at finnSummer fra forrige deloppgave ikke bare printer til terminal, men faktisk returnerer en liste med par (tupler). Andre antagelser kandidater har gjort er innafor, så lenge de beskrives godt.

Det vi er ute etter å teste her er at kandidaten kan skrive en test-funksjon, dvs.:

- Funksjonen deklarerer med @Test

- Navnet på funksjonen bør beskrive hva som testes
- En eller annen form for Assert-benyttes
- Testmetoden bør ikke være for lang eller teste for mye

2.4 Refleksjon - teamarbeid (15p)

a) Beskriv tre praksiser for godt teamarbeid og begrunn hva som gjør dem til gode praksiser. Ta utgangspunkt i pensum, men relater gjerne beskrivelsene til eget teamarbeid fra prosjektarbeidet i IN2000. NB: Ikke nevne navn/teamnummer i besvarelsen! (10p)

Dette er en refleksjonsoppgave, dermed er det viktigste at studentene klarer å faktisk reflektere, fremfor å kun beskrive. En god besvarelse vil klare å henvise både til pensum (primært Sommerville kap 3 og forelesningen om teamarbeid), og relatere til eksempler, for alle tre praksiser som beskrives. Det MÅ ikke være egne erfaringer fra prosjektarbeidet, men det må være eksempler og begrunnelser i noen form.

Mulige praksiser som kan trekkes inn.

- Fasilitering av arbeidsprosessen
- Fokus på gode og effektive møter
- Bruk av digitale kanaler for skriftlig kommunikasjon
- Parprogrammering som samarbeidsverktøy
- Bruk av product backlog (gir oversikt)
- Bruk av flytverktøy (e.g. ulike boards)
- Bruk av teamkontrakt

Andre praksiser som ikke listes her kan også være aktuelle, det viktigste er god refleksjon rundt **hvorfor** det er gode praksiser.

b) Nevn noen årsaker til psykologisk trygghet. (5p)

NB! Her var det flere studenter som stilte spørsmål ved formuleringen “årsaker til”, noe vi ser at kan være en litt forvirrende formulering. Det vi her var ute etter er hva er det som kan skape psykologisk trygghet, hvilke faktorer bør være til stede for å legge til rette for psykologisk trygghet.

Mange vil nok tolke dette til å være "psykologisk trygghet i team", men ettersom vi faktisk ikke presiserer det, så vil vi også gi poeng hvis det beskrives psykologisk trygghet på arbeidsplassen. Andre sosiale sammenhenger anses som mindre relevante.

Psykologisk trygghet kan defineres som graden av tillit og komfort teammedlemmer opplever når det gjelder å uttrykke meninger, stille spørsmål og innrømme feil uten frykt for negative konsekvenser. I utvikling er dette særlig viktig, ettersom det ofte er komplekse tekniske problemstillinger som løses i team. En høy grad av psykologisk trygghet fremmer også innovasjon og kvalitet.

Her følger en del faktorer som kan relateres til psykologisk trygghet. En god besvarelse bør nevne minst tre av disse og bør også klare å utdype kort rundt hvorfor. *Hvis det kun listes opp årsaker kan dette også gis poeng, men ikke mer enn 3 poeng.*

- Ikke-dømmende kultur rundt feil hvor man tør og eksperimentere, der feil betraktes som læringsmuligheter, gjør det lettere å si fra hvis man ser noe som er feil
- Tydelige roller og forventninger - Hvis alle vet hva som forventes av dem, og hvordan deres arbeid henger sammen med resten av teamet, kan man unngå misforståelser som kan skape usikkerhet.
- Åpen og hyppig kommunikasjon, feks regelmessige møter og arenaer for spontan kommunikasjon (både fysisk og Slack, Teams osv.) hvor spørsmål kan stilles og besvares uten tidspress.
- Gjensidig respekt og tillit mellom kolleger, for eksempel ved å lytte aktivt, anerkjenne andres innsats og behandle hverandre likt uavhengig av stilling eller erfaring.
- Fokus på felles mål og deltakelse i beslutninger rundt mål, estimerer og tekniske valg, viser at hver rolle (UX, utvikling, test osv) er viktig.
- Aksept for ulikhet og mangfold av alle slag
- Regelmessig refleksjon og kontinuerlig forbedring, kan gi bonus om det relateres til de smidige prinsippene, feks retrospektiver eller lignende møter hvor teamet snakker om hva som gikk bra og hva som kan forbedres
- Viktigheten av teamledelsen og seniore teammedlemmer som kulturbærere, som aktivt viser at det er greit å stille spørsmål, be om hjelp eller innrømme feil, signaliserer at alle kan delta uten frykt for negative konsekvenser.

Del 3: Caseoppgave

Denne delen inneholder caseoppgaven. For hver oppgave i delen er det spesifisert hvor mye poeng det er mulig å oppnå.

Totalt er det mulig å oppnå 44 poeng på denne delen.

3.1 Prosjektoppstart (14p)

Beskriv de mest sentrale aktørene for dette systemet og hvilke som er primære aktører og hvilke som er sekundære aktører. Begrunn svaret. (4p)

Eksempler:

- Ansatte som foretar vareopptelling (primær)
- Ledere (sekundær)

Her er det ingen fasit, men kandidatene bør begrunne valg av aktører og hvorfor de er primære/sekundære.

Teamet bestemmer seg for å gjennomføre "etnografi". Beskriv denne forskningsmetoden, samt hvordan du ville gjennomført etnografi i kontekst av casebeskrivelsen. (4p)

Her skal kandidaten beskrive hva etnografi er, f.eks.: *etnografi er observasjon over lengre tid*

I tillegg forklares hvordan det kan gjennomføres i kontekst av caset, f.eks. ved å observere hvordan vareopptelling foregår.

Reflekter rundt universell utforming og WCAG-krav til et system som ikke skal brukes av allmennheten, men internt i en bedrift. (4p)

Her er det ingen fasit, men følgende er punkter som kan nevnes i refleksjonen:

- Viktig å lage et system som mange kan bruke
- Kan bli vanskelig for enkelte å utføre arbeidsoppgavene sine
- Uheldig dersom en ansatt skulle få en nedsettelse som gjør at vedkommende ikke kan gjøre jobben lenger pga. "dårlig" app.
- "Nødvendig for noen, men bra for alle"

3.2 Funksjonelle og ikke-funksjonelle krav (9p)

a) Definer tre ikke-funksjonelle krav for appen til butikken. (3p)

Her skal kandidaten definere tre ikke-funksjonelle krav. 1 poeng per krav, og kravene bør være avgrenset av dette caset og ikke for generelle. Her er det lov å trekke dersom kravene ikke er målbare.

b) Definer tre funksjonelle krav for appen til butikken. (3p)

Her skal kandidaten definere tre ikke-funksjonelle krav. 1 poeng per krav, og kravene bør være avgrenset av dette caset og ikke for generelle.

c) Det oppstår uenigheter i teamet om hvilket krav som er viktigst og dere klarer ikke å bli enige om hva dere skal jobbe med først. Reflekter kort hva du ville gjort for å løse dette. (3p)

Her er det flere riktige svar, men følgende kan nevnes i en besvarelse:

- Produkteier bestemmer
- Snakke med bruker av systemet
- Teamet blir enige seg i mellom ved å anvende

3.3 Definer en klasse i Kotlin slik at JSON-objektet kan deserialiseres. (4p)

```
data class Product(  
    val productId: Int,  
    val name: String,  
    val producer: String,  
    val lastScanned: LocalDateTime,  
    val relatedProductIds: List<Int>  
)
```

Merk at det vi er ute etter her er at kandidaten bruker **data class**, at datatyper som Int/String og Dato er satt riktig, og at relatedProductsId er ei liste og ikke en annen data class.

3.4 App som fryser (8p)

Flere viktige ting å trekke frem som forbedringspunkter/ting en kunne gjort annerledes, her er noen av de:

- Unngå runBlocking: runBlocking bør ikke brukes i UI-tråden. Dette blokkerer tråden til den nye Couroutinen fullfører, som kan føre til frysing.
- Bruk av CoroutineScope: I stedet for å bruke runBlocking, kan du bruke en CoroutineScope som er tilknyttet en ViewModel.
- Tilbakemelding til bruker: Det kan være nyttig å gi tilbakemelding til brukeren hvis API-anropet feiler, for eksempel ved å vise en feilmelding.
- Feilhåndtering: Bør kaste Exception fremfor å logge feil.

3.5 Refleksjon UU (10p)

a) For hver av kravene; hva ville du gjort for å ettergå kravet? (6p)

Her skal kandidatene komme med forslag til hvordan de kan ettergå kravene, noen eksempler kan være:

Typiske virkemidler for suksesskriterium 1.4.4:

- Bruk relative enheter
 - Prosent eller .sp (her hadde jeg skrevet .dp, men .sp er riktig) istedenfor absolutte piksler, da skaleres de i forhold til brukerens preferanser
- Unngå faste tekststørrelser
 - Ikke lås tekststørrelsen til en bestemt størrelse, da dette kan hindre brukere fra å endre den etter behov
- Tilrettelegg for tekstforstørring
 - Appen burde fungere godt når tekststørrelsen øker
 - Sørg for at ingen innholdselementer blir beskåret eller overlapper over hverandre ved større tekst
- *Typiske virkemidler for suksesskriterium 3.1.3:*
 - Påse at definisjonen for ordet i den gitte konteksten er tilgjengelig
 - Ha informasjonsknapper som forklarer begrepet eller punktet
 - Ha mulighet til å trykke på ordet for å få forklaringer
 - Ikke bruk kompliserte ord unødvendig

Kandidatens svar bør ikke være en omformulering av suksesskriteriet. Det viktigste er at kandidatene viser forståelse for hva kravene innebærer, og dette kan besvares både med både teknisk og ikke-teknisk tilnærming.

b) Hvorfor er begge disse kravene viktig å oppfylle? (4p)

Suksesskriterium 1.4.4:

- Hensikten med kravet er at personer med moderat nedsatt syn skal kunne forstørre og lese tekst, uten bruk av hjelpemiddelteknologi. Kravet er ment å ivareta personer med nedsatt syn.

Suksesskriterium 3.1.3:

- Brukere kan identifisere og lære hva de uvanlige ordene betyr. Kravet er ment til å gi definisjoner for teknisk sjargong og uvanlige ord. Kravet ivaretar de fleste, men særlig personer med kognitiv nedsettelse for å gi forståelse for innhold.

Kandidatens svar bør reflektere noe fra punktene over.