Om eksamen

Eksamen i IN2000 våren 2022

Eksamensdato: 10. juni 2022
Eksamenstid: 09.00–13.00 (4 timer)
Hjelpemidler: Ingen

Les denne forsiden nøye.

Oppgaven består av:

- Teorispørsmål, flervalgsoppgaver 10 poeng
- Teorispørsmål fritekstoppgaver 10 poeng
- Case-oppgave (modellering/håndtegning) 30 poeng
- Tekniske oppgaver 30 poeng
- Refleksjonsoppgave 20 poeng

Hvis oppgaven er uklar, kan du gjøre dine egne forutsetninger. Gjør i så fall rede for disse.

Digital håndtegning: I dette oppgavesettet skal du svare med digital håndtegning i Case-oppgaven (nr. 7), deloppgave 2 og 3. Du bruker skisseark du får utdelt. Det er anledning til å bruke flere ark per oppgave. Se instruksjon for utfylling av skisseark i vedlegg som du finner nederst på skjermen.

Det er **IKKE** anledning til å bruke digital håndtegning på andre oppgaver enn case-oppgaven. Det blir **IKKE gitt ekstratid** for å fylle ut informasjonsboksene på skisseark (engangskoder, kand.nr. o.l.)

Tekniske oppgaver: I de tre første oppgavene av de tekniske oppgavene er det viktig at du **kun** skriver det som **skrives ut i terminal** når koden kjøres. Der skal du skrive **kun et heltall** (ingen desimalkomma) eller **kun en tekststreng** (ingen anførselstegn).

PDF-dokument til pizzaoppgaven
Oppgaven om RandomPizza (nr. 12–15) er beskrevet i et lengre PDF-dokument. Dette dokumentet er også **tilgjengelig som vedlegg** nederst på skjermen slik at du kan få det i en egen fane.

Tallet i parentes bak en oppgave er maks poeng for den deloppgaven. F.eks. betyr (5p) maks 5 poeng. På hele eksamensoppgaven er det maks 100 poeng.

Lykke til!

1 Oppgave 1 (2p)

Handler universell utforming om både prosess og produkt?

Velg ett alternativ:

- Nei, bare produkt
- Nei, bare prosess
- Ja, begge deler
- Nei, ingen av delene

Maks poeng: 2

2 Oppgave 2 (2p)

Hva er **ikke** et prinsipp for god arkitektur?

Velg ett alternativ:

- Vedlikeholdbarhet
- Testbarhet
- Høy kobling
- Ytelse

Maks poeng: 2

3 Oppgave 3 (2p)

Hva er Gradle?

Velg ett alternativ:

- Et Rammeverk
- Pakkehåndteringssystem
- Versjonshåndtering
- Testebibliotek

Maks poeng: 2

4 Oppgave 4 (2p)

Hvilket av følgende krav for en printer er et ikke-funksjonelt krav?

Velg ett alternativ:

- Printeren skal kunne printe i farger og svart-hvitt
- Printeren skal støtte flere arkstørrelser
- Printeren skal kunne scanne dokumenter og sende dem til pc’er via et trådløst nettverk
- Printeren skal være stillegående

Maks poeng: 2

5 Oppgave 5 (2p)

Hvilke av disse er **ikke** et mål for trusselmodellering?

Velg ett alternativ:

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

Maks poeng: 2

6 Oppgave 6 (10p)

Nevn 5 eksempler på karakteristikker i kildekode som kan indikere et problem (såkalt «code smells») og forklar hva som kan gjøres for å unngå dem.

Skriv ditt svar her

Format

Words: 0

Maks poeng: 10

7 Case-oppgave (30p)

Case:
Selskapet RETT-i-NEBBET tilbyr tjenester for bestilling og frakt av mat i de største byene i Norge. Stort sett alle ”take away” restauranter er med i tillegg til en del andre restauranter.

Din oppgave er å modellere deler av et system for en mobilapp som gjør brukere i stand til å bestille mat, og få det levert fra en restaurant. Maten leveres med sykkel eller bil fra selskapet.

Appen skal støtte:

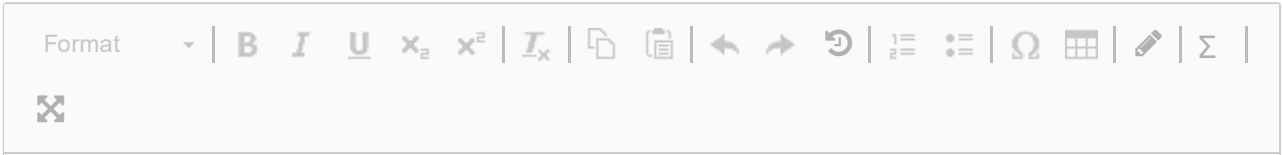
- Registrering av brukerkonto med tilhørende informasjon
- Registrering av bankkort/kredittkort for betaling, samt kunne koble opp med Vipps som betalingsmetode
- Vise oversikt over restauranter i nærheten av brukeren
- Vise meny for hver restaurant, samt allergener for de rettene som inneholder det
- Vise oversikt over restauranter i forskjellige kategorier, f.eks italiensk, indisk, meksikansk ol.
- Brukeren skal kunne lagre sine favorittrestauranter, og få forslag til andre restauranter basert på disse

Oppgave 1 (5p)—besvares i tekstfeltet nedenfor
Gi en tekstlig beskrivelse av use-caset “Bestill mat”, med minst en alternativ flyt. Mot slutten av bestillingen skal brukeren kunne velge betalingsløsning.

Oppgave 2 (13p)—besvares på skisseark, se informasjon om Scantron nederst på skjermen
Lag et sekvensdiagram basert på use-caset fra oppgave 1.

Oppgave 3 (12p)—besvares på skisseark, se informasjon om Scantron nederst på skjermen
Lag et klassediagram som reflekterer de større strukturene av systemet beskrevet i caset. Diagrammet bør også inkludere klasser, metoder og attributter du beskrev i deloppgave 2.

Skriv svaret på oppgave 1 her:



Words: 0

Maks poeng: 30

8 list.filter (2p)

Gitt følgende Kotlin-kode, hva skrives ut til terminalen?

```
fun main() {
    val list = listOf(0,1,2,3,4,5,6).filter { it % 2 == 0 }
    println(list.size)
}
```

Skriv svaret her slik det skrives ut til terminalen:

Maks poeng: 2

9 list.indices (2p)

Gitt følgende Kotlin-kode, hva skrives ut til terminalen?

```
fun main() {
    val list = listOf(1,2,3)
    for (i in list.indices) {
        list.find { it == i }?.dec()
    }

    println(list.sum())
}
```

Maks poeng: 2

10 mutableListOf (3p)

Følgende kode vil gi 2 errors og 1 warning. Velg de linjene som enten gir error eller warning (til sammen 3).

```
1   fun main() {
2       var list1 = listOf(1,2,3)
3       val list2 = mutableListOf(1,2,3)
4       val list3 = mutableListOf(1,"b",3)
5       list1.add(list2.first())
6       println(list2.add(list1.last()))
7       println(list3.add(list1[list1.size / 2]))
8       println(list2 += list1)
9       println(list3 + list1)
10  }
```

Velg 3 linjenummer som gir feilmelding:

- [ ] 1
- [ ] 2
- [ ] 3
- [ ] 4
- [ ] 5
- [ ] 6
- [ ] 7
- [ ] 8
- [ ] 9

Maks poeng: 3

11 XML (3p)

Det er to feil i denne XML-filen. Identifiser disse og beskriv hvordan de kunne blitt fikset.

```
<restaurant navn=”trattoria popolare”>
    <meny>
        <rett>
            <navn>Spaghetti Vongole</navn>
        </rett>
        <rett>
        <navn>Spaghetti al Pommodoro</navn>
    </meny>
</restaurant>
<restaurant navn=”mother india”>
    <meny>
        <rett>
        <navn>Chicken Tikka Masala</navn>
        </rett>
        <rett>
        <navn> Saag Aloo</navn>
        </rett>
    </meny>
</restaurant>
```

Skriv ditt svar her

Maks poeng: 3

Pizzaoppgaven PDF

Vedlagte pdf-dokument "RandomPizza" må leses før de neste oppgavene besvares.

12 Pizza 1 (4p)

```
fun getPizza(): LiveData<Pizza> {

}

fun loadPizza() {

}
```

Før du løser denne oppgaven, må du lese hele dokumentet RandomPizza.pdf.

I denne deloppgaven skal ViewModel kobles opp til DataSource. Definer innholdet av metodene i filen *MainActivityViewModel.kt*. Dette innebærer å fylle ut metodene *getPizza()* og *loadPizza()* som beskrevet i TODO-ene.

Du kan kopiere inn disse kodebitene for å få fullstendige metoder:

Skriv ditt svar her

```
1
```

Maks poeng: 4

13 Pizza 2 (7p)

```
class MainActivity : AppCompatActivity() {
    private lateinit var binding: ActivityMainBinding
    private val viewModel: MainActivityViewModel by viewModels()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
        val pizzaButton = binding.pizzabutton
        val pizzaText = binding.pizzatext
    }
}
```

Før du løser denne oppgaven, må du lese hele dokumentet RandomPizza.pdf.

I denne oppgaven skal du implementere at det hentes en tilfeldig pizza fra API-et når brukeren trykker på knappen, *pizzaButton*. Dette skal gjøres ved bruk av metodene og LiveData-objektet fra deloppgave 1. Når LiveDataobjektet endrer seg skal TextViewet, *pizzaText*, oppdateres.

Du kan kopiere inn denne for å få en komplett klasse:

Skriv ditt svar her

Maks poeng: 7

14 Pizza 3 (9p)

Oppgave a) (6 poeng)

Klassestrukturen som er beskrevet i filene i *RandomPizza.pdf* følger Androids anbefalte arkitektur, hvor man skiller mellom *presentasjonslaget* og *datalaget*. Beskriv hva som representerer presentasjonslaget og datalaget i denne oppgaven, samt hva som er ansvarsområdet til de to lagene.

Skriv ditt svar her

Oppgave b) (3 poeng)

Brukerne av RandomPizza har rapportert at de ønsker mer enn én pizza om gangen, da de ofte bestiller flere samtidig. Derfor skal RandomPizza videreutvikles til å vise en liste av fem pizzaer til brukerne når de trykker på knappen. Beskriv kort hvordan og hvor du ville endret applikasjonen for å tilfredsstille dette nye kravet. Du trenger ikke skrive kode her.

Skriv ditt svar her

Maks poeng: 9

15 Refleksjon (20p)

Oppgave a) Smidig metodikk (10 poeng)

I løpet av prosjektarbeidet har teamene arbeidet smidig. De fleste har benyttet enten Scrum, Scrumban eller Kanban.

Velg to smidige tilnærminger. Diskuter fordeler og ulemper ved bruk av de valgte tilnærmingene dersom de skal brukes i et prosjekt med et så kort tidsperspektiv som IN2000.

Skriv ditt svar her

Oppgave b) Forskningsmetoder og universell utforming (10 poeng)

Diskuter hvordan du ville gått frem for å evaluere kravet “Appen skal være universelt utformet” ved hjelp av en eller flere forskningsmetoder.

Velg tre målbare egenskaper ved systemet eller ved interaksjonen mellom bruker og systemet som støtter opp under universell utforming.

Skriv ditt svar her

Maks poeng: 20

Document 2
Attached

RandomPizza: Arkitektur-oppgave (20p)

Mange sliter med å bestemme seg for hvilken pizza de skal bestille, derfor har selskapet RandomPizza laget en app som fikser dette problemet. Appen generer en tilfeldig pizza-type når brukeren trykker på en knapp. RandomPizza har fått et endepunkt fra en pizza-leverandør som gir dem en tilfeldig pizza, men har ikke kommet helt i mål med utviklingen av appen.

Du har blitt gitt fire ulike filer, *PizzaDataSource.kt*, *Pizza.kt*, *MainActivityViewModel.kt* og *MainActivity.kt*. To av disse filene er ikke komplette for en fungerende app, henholdsvis *MainActivityViewModel.kt* og *MainActivity.kt*.

Du skal fylle ut koden som er i disse to filene for å lage en enkel app. Appen skal inneholde en knapp (Button). Når knappen trykkes på, skal appen gjøre et api-kall, og hente en tilfeldig type pizza, og dette skal oppdateres i et TextView. Du kan anta at Button og TextView er implementert i en fungerende XML-fil. Koblingen til disse er allerede gjort i *MainActivity.kt*.

Det er ikke forventet kjørbar kode i denne oppgaven.

Pizza.kt

```
data class Pizza (val id: String, val name: String)
```

PizzaDataSource.kt

```
class PizzaDataSource {
    suspend fun fetchRandomPizza (): Pizza ? {
        val path = "https://getrandomfood.com/pizza"
        val gson = Gson()

        try {
            Log.d( tag: "MAIN", Fuel.get(path).awaitString())
            return gson.fromJson(Fuel.get(path).awaitString(), Pizza::class.java)
        } catch(exception: Exception) {
            Log.e( tag: "MAIN", msg: "${exception.message}")
        }
        return null
    }
}
```

MainActivityViewModel.kt

```
class MainActivityViewModel: ViewModel() {

    private val pizzaDataSource = PizzaDataSource()

    private val pizza: MutableLiveData<Pizza> by lazy {
        MutableLiveData<Pizza>()
    }
    //TODO: Return the pizza-LiveData object.
    fun getPizza(): LiveData<Pizza> {

    }

    //TODO: Do an asynchronous operation to fetch a random pizza
    fun loadPizza() {

    }
}
```

MainActivity.kt

```kotlin
class MainActivity : AppCompatActivity() {
    private lateinit var binding: ActivityMainBinding
    private val viewModel: MainActivityViewModel by viewModels()


    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        val pizzaButton = binding.pizzabutton
        val pizzaText = binding.pizzatext

    }
}
```