Oppgavesettet består av 2 sider

HØYSKOLEN KRISTIANIA
PG3302 Software design
Mappeeksamen

Gjennomføring

- **Mappeeksamen, skal leveres i gruppe** (3-5 personer).
- **Arbeidsmapper** kan vises foreleser i øvingstimer for **tilbakemelding underveis**.
- **Presentasjonsmappe** skal leveres via **WISEflow.**
- Den endelige presentasjonsmappen skal inneholde:
    1. Dokument som beskriver prosessen og evt. innhold/bruk av løsningen. (pdf)
    2. Kildekode og evt. andre nødvendige filer i denne sammenheng. (zip)
    3. Kjørbar (.exe-)versjon, *som kan startes via Windows!* (zip)

Leveranseinfo, 3 deler

Her følger litt mer info om de 3 delene i leveransen:

1. Dokument som beskriver prosessen og evt. innhold/bruk av løsningen. (pdf)

Denne delen går ut på å lage en kort, skriftlig rapport som dokumenterer prosjektet dere har jobbet på. Foreslått lengde: minimum 1500 til 2000 ord (3 til 4 sider ved vanlig størrelse og linjeavstand).

Elementer som bør være med:

- **Prosessen**: Skriv noen få avsnitt om hvordan dere har jobbet med besvarelsen.
- **Elementer fra pensum:** En kort liste med hvilke elementer fra pensum dere har med.
- **Spesielle utfordringer & fancy features:** Si litt om hva dere slet med, og hva dere er spesielt fornøyde med.
- **Kjente bugs:** Er det noen feil i programmet som dere kjenner til? Nevn de i så fall her.
- **Kildehenvisning – VIKTIG:** Om det oppdages at dere har benyttet ferdig materiell som dere *ikke* opplyser om (ikke nevner her), kan det regnes som juks! Husk derfor å oppgi kilder, vi ønsker å unngå plagiatsaker!

2. Kildekode og evt. andre nødvendige filer i denne sammenheng. (zip)

Dette er kodeimplementasjonen av programmet. Lever kildekoden (og tilhørende filer), slik at sensor kan se programmeringen og kodestrukturen! Merk: Bør kunne åpnes i Visual Studio 2019.

Husk: Vi har lært om modellering av softwareløsninger i dette emnet: klasse- og sekvensdiagrammer i UML. Figurer og annet innhold dere produserer i den sammenheng bør også leveres her. Det kan være screenshots fra digitale verktøy, mobilfoto av papirbaserte tegninger, etc.

3. Kjørbar (.exe-)versjon, *som kan startes via Windows!* (zip)

Kjørbar versjon skal kunne benyttes på Windows 10. Sørg for at både eventuell .exe-fil og nødvendige mapper og filer blir med.

Utforming av løsning

Oppgaven går ut på å **designe en softwareløsning**, deretter **programmere løsningen i C#**.

"Softwareløsning" er et vidt begrep. Emnet går for både 2. klasse Programmering og 2. klasse Spillprogrammering, samt som valgemne for 3. klasse Intelligente systemer. Innenfor begrepet "softwareløsning" godkjennes i denne sammenheng:

- Standard C# Console prosjekt.
- C# WPF prosjekt (altså en vindusbasert løsning).
- Unity prosjekt. (Merk: Fokuser i så fall på C# koden! *Ikke på grafikk eller gameplay delene*)

Designe en softwareløsning

Når det gjelder "designe en softwareløsning", menes det med fokus på kodestrukturen. Ikke funksjonalitet/brukeropplevelse, ikke visuell still, ikke markedsaspektet eller annet som ikke hører inn under pensum i emnet.

Programmere løsningen i C#

Dere står rimelig fritt til å velge type prosjekt/program, f.eks.:

- Standard 3-lags løsning med database i bunn (integrasjon av database forklares i forelesning 10), domenelogikk i midten og Console eller vindusbasert (WPF) frontend. (*Ikke* fokuser på UI, det er ikke del av pensum.)
- Dataspill, enten som spilles i Console, i et OS-vindu (WPF) eller med grafikkbasert UI (Unity). NB: Husk i så fall på et det er *kodestruktur som er fokus!* Ikke gameplay, art, sound, e.l.

Pass uansett på at dere får med et godt utvalg elementer fra pensum! (Se under.)

Pensum, hva bør med

Elementer fra pensum: (ikke en uttømmende liste, vurderer din gruppe flere/andre elementer fra pensum er det bare fint)

- UML diagrammer
- C# .NET (og en del finesser i språket: properties, operator overloading, …)
- SOLID (beskriv hvilke prinsipper dere har hatt fokus på i dokumentasjonen)
- Design patterns (7 stk., men plukk noen få: *prøver man å benytte alle blir det "lapskaus"*)
- Refactoring (sørg for at koden holdes oversiktlig og lett å sette seg inn i, beskriv refactoring prosessene deres, de kan være vanskelige å se ut fra endelig kode)
- Lagdeling (evt. MVx DP, men ikke forventet: det kommer helt på slutten av undervisningen)
- Unit testing (omfattende å skrive, så ok å bare benytte det på deler av løsningen: så lenge dere viser at dere mestrer unit testing i noen utvalgte kodefiler får dere full uttelling)
- Multithreading / event-basert kode (pass i så fall på at dere skriver trådsikker kode)
- Parprogrammering (passer fint for eksamensoppgaver i gruppe, beskriv i dokumentasjonen hvordan dere i så fall benyttet dette, samt erfaringer dere gjorde dere)
- Bruk av versjonskontroll

Merk: Tenk god struktur, ikke nødvendigvis hva som hadde vært praktisk for et så lite prosjekt utenfor skolesammenheng. Få med mange elementer fra pensum. Gjør dere noe som ikke virker logisk (fordi dere vil vise at dere behersker en gitt del av pensum, men finner ikke en perfekt plass å vise den på), forklar hvorfor dere har det med, *både* som kodekommentar og i deres vedlagte pdf dokument.

- Slutt på oppgavesettet -