Teori spørsmål – Obligatorisk innlevering 4

Teorioppgave 1 - Exception

«Forklar hva en exception er, hvordan vi kan håndtere dem og i hvilke tilfeller slik håndtering kan være relevant.»

Når man koder skjer det ofte feil. Noen ganger planlagt, ofte ikke. Å legge til en Try/Exception statement kan fange opp feilmeldinger og foretelle deg hva som skjedde feil ved å gi printe en feilmelding. I disse tilfellene vil da også ikke programmet krasje, men helle fortsette videre.

Den vil «prøve» å gjøre det som er it Try-blokken, men hvis den møter på en error så vil den stoppe og gå til Exception-blokken i stedet. Etterpå vil den fortsette vanlig videre i koden.

Et eksempel på dette er en kalkulator. Du kan ha regnestykket i en Try/exception-block, som vil fange opp for eksempel en «ZeroDivisionError».

Teorioppgave 2 - Klasse

«Forklar med egne ord hva en klasse er. Gi gjerne et enkelt eksempel eller to»

Klasser er en ferdiglaget oppskrift for å lage objekter, og dens variabler og funksjoner. Den gjør det enklere å dele opp koden din i biter (Object Oriented Programming), som gjør det lettere å holde styr på koden. Et eksempel er et enkelt skyte spill. Spilleren vil være en egen klasse, fiender vil være en egen klasse. Skuddene du skyter vil være en egen klasse. Hver fiende vil da også arve noen generell fiende klasser, men også ha sin egen klasse basert på typen fiende de er.

Teorioppgave 3 - Objekt

«Forklar med egne ord hva et objekt er, og hva dets relasjon til klasser er.»

Når du oppretter en instanse av en klasse, lager du et objekt av klassen. Klassen er oppskriften vi bruker til å bygge objektet. Alle variabler og funksjoner inne i objekter kommer fra klassen som ble brukt til å bygge den.

Programmeringsoppgave 2 - Dokumentasjon

«I programmeringsoppgave 2 skal du skrive et relativt omfattende program. Underveis/etter at du har skrevet dette programmet skal du her skrive litt om hvordan du har jobbet med denne oppgaven og kommet frem til din løsning. Dette gjelder spesielt hvis du sliter med noe underveis, hvor du burde dokumentere hva du slet med og hva du forsøkte å gjøre for å komme videre. Reflekter også hva som kunne vært bedre med din løsning og fremgangsmetode. Du trenger ikke skrive en bok i lengde, men det er nok naturlig å ende opp med i alle fall en half side tekst for denne dokumentasjonen.»

Introduksjon

Til å begynne med vil jeg si dette var en morsom oppgave. Den var åpen nok i oppgaveteksten til at man kunne gjøre mye forskjellig i løsningen sin. Først tenkte jeg å lage en tekstbasert løsning i Python terminalen, men etter litt tenking synes jeg det ble litt kjedelig, så jeg bestemte med for å lage en enkel GUI med Pygame.

GUI ble utviklet med et Pygame bibliotek jeg begynte å utvikle for et par uker siden. Biblioteket ble laget slik at jeg lett kunne lage en GUI, men knapper, sliders, inputs, og mer, slik at jeg kun kunne fokusere på logikken til programmene jeg ville lage. Dette biblioteket er fortsatt i Alpha fasen, for det er mye jeg ikke er fornøyd med der ennå.

Planlegging

Måten jeg jobbet med dette programmet var at jeg først begynte å planlegge logikken til spille Blackjack, og etter jeg følte jeg hadde en god plan på det begynte jeg å planlegge logikken på spillets gang.

Jeg kom fram til at hver runde har essensielt tre faser.

1. Get Cards Phase
2. Player's Turn Phase
3. Dealers Turn Phase

Mens Blackjack logikken var ganske rett frem. Hver gang noen får et kort så sjekker du om samlet verdi er over 21. Hvis den er det så sjekker du om du kan gjøre om en 11er til en 1er. Utenom det var det bare å legge inn et par if-statement for å sjekke åssen utfall spillet fikk. Fikk noen 21? Ble fikk dealeren høyre enn deg? Fikk du høyre en dealeren, osv.

Problemer underveis

Jeg slet ikke med noe særlig underveis, men jeg møtte på et liste-comprehension problem som tok nesten en time å finne ut av.

```
class Flexbox():
  def __init__(self,
               x = 0, y = 0, width = 300, height = 300,
               border_radius = 0, background_color = (50,50,50),
               direction = 1, id = None,
               border_color = (0,0,0), border_width = 0,
               contents = [], disabled=False):
```

Dette en konsktruktøren til Flexboksene jeg brukte i kodingen. Problemet er oppsto hvis jeg lagde flere instanser(objekter) omgangen og ikke fylte ut contents med en gang. Da ville de forskjellige objektene dele en liste, fordi Python kobler de opp til hverandre. Så alt jeg la til i et objekt's content liste, ble lagt til i alle objekter sine lister. Noe som førte til problemer med å legge ut kortene, siden det var her kortene ble plassert. Etter en del troubleshooting så kom jeg opp med denne løsningen:

```
class Flexbox():
  def __init__(self,
               x = 0, y = 0, width = 300, height = 300,
               border_radius = 0, background_color = (50,50,50),
               direction = 1, id = None,
               border_color = (0,0,0), border_width = 0,
               contents = None, disabled=False):
  # The things we have inside the box
  self.contents = contents
  if self.contents == None:
    self.contents = []
```

Hva ville jeg gjort annerledes?

Hvis jeg skulle gjort dette annerledes, ville jeg lagt til grafikk, som bilder av kort og Blackjack bord og chips. Jeg ville også ha lagt til animasjoner for å dele ut kort og muligheten for flere spillere om gangen. I tillegg ville jeg lagt til muligheten for å splitte kort og doble penge innskuddet ditt.

Dette var ting jeg vurderte, men fordi dette bare en innleveringsoppgave, og å legge til dette hadde vært tidskrevende så valgte jeg å ikke fokusere på det nå.