Oblig 6 Alternativ 1 - Eksamensoppgave

Teori

Oppgave 1.1

Hva er funksjonell programmering og hvordan skiller det seg fra imperativ programering?

Funksjonell programmering og imperativ programmering er programmeringsparadigmer. Funksjonell programmering er programmering basert på funksjoner. Alle funksjoner i funksjonell programmeriner er verdier akkurat som alle andre typer objekter. Det kan lagres, sendes som argument og de kan retuneres. Man kan ha lokal og anonyme funkjsoner som kan legges i hvernadre og kan settes sammen modulært. I dette er det rekurskjon og ikke løkker.

alle funksjonene i ren funksjonell programmering er deterministiske matematiske definisjoner så vi får en input av en eller annen type og vi får en Output basert på den typen helt deterministisk. Det kan ikke påvirkes av mutable states eller sideffekter. Det vil si at vi har ikke noe tilstandsvariabler eller tilstands avhengige handlinger som påvirker koden, er ren inn og ut.

imperativ programmering har statements eller kommandoer som som endrer på en program tilstand så vi så gjør oprasjoner.

funksjonell programmering da er i andre enden av den skalaen her er så den endrer sjeldent på program tilstand, stort sett bare input og basert på input så gjør vi returnerer vi noe. imperativ programmering bruker løker så har funksjonell programmering primært rekursjon men også høyere ordens funksjoner.

Oppgave 1.2

Hva er currying?

Er at man går fra å ha en parameter liste til flere parameter lister. Du går fra en parameterliste til å ha flere parametre lister du får flere parentes grupper da parenteser så kan vi gir oss mulighet dele opp en parameterliste.

Eksamensoppgave

Oppgave 1.1

a) Polymorfisme
   Polymorfisme er en måte å få objekter til og utgi seg for å være fra klasser over. Om man skal sammenligne to objekter, må man ha en klasse over som de arver fra, sånn at de objektene brit til samme objekt «type».

b) Interface
   Interface ligner litt som abstract, forsjellen er at interface ikke har noe kodekropp({}).
   Har Returntype, metode navn og eventuele parametere.

Oppgave 1.2

a) Final

Final er at man setter noe konstant og skrives alltid med store bokstaver.
Når man har satt en variabel final, så man ikke endre den gitte verdien. Final metode kan ikke overloades det vil si at man ikke kan endre metodens funskjon av under klasser. Final klasse kan man ikke lage under klasser til.

b) Static

Static betyr det til hører til selve klassen og ikke instansen objektet er i. Om man lager en static variabel, vil alle instansene an den klassen ha tilgang til variabelen. Om man lager en static metode kan den bli brukt uten og initialisere et objekt av klassen.

c) Override

Er når man lager en fucksjon i en under klasse (en klasse bygget på en annen klasse) og funksjinen allerede eksisterer. Det vil si at når man lager en ny funkjson, vil java erstatte den gamle funskjonen med den nye, Sånn man har overskrevet den andre variablen/metoden.

Oppgave 1.3

Gjør rede for begrepene aggregering og komposisjon. Underbygg utredningen ved å lage et enkelt UML-diagram og eksempel kode.

Oppgave 2.1

```
public class A {
      private String kode;

      public A(String kode) {
            this.kode = kode;
      }
}

public class B extends A{
      private int verdi;

      public B(String kode, int verdi) {
            this.kode = kode;
            this.verdi = verdi;
      }
}
```

Feil med dobbel semkolon i String kode. Andre feilen er at man ikke får akksresert kode i B sin kostruktør om man vil at B sin konstruktør skal lese kode må man hente fra A sin konstruktør ved å sette som super

Oppgave 2.1.2

```
public interface  A {
      void printTekst();
}

public class B extends A{
      private String navn;
      private String tekst;

      public B(String tekst, String navn) {
            this.tekst = tekst;
            this.navn = navn;
      }

      @Override
      public void printTekst() {
            System.out.println(navn + ": " + tekst);
      }
}
```

implements

B kan ikke utvide klassen til A fordi det er et interface og ikke en klasse. For og bruke inholdet til A må man implementere(implements) ikke utvide(extends).

Oppgave 2.2.1

Volvo XC90

Volvo XC60

Oppgave 2.2.2

Ford Mondeo

Volvo XC90

BMW X5

Oppgave 2.2.3

Volvo XC90

Ford Mondeo

BMW X5

Lada Niva