Øvingsforelesning 4

#School/TA/TDT4100/Lectures

Læringsmål:

- Lære å lage Java programmer hvor flere klasser jobber sammen

Oppgaver

I denne oppgaven skal vi lage en kalkulator som kan utføre diverse kalkulasjoner som består av operasjonene addisjon, subtraksjon, multiplikasjon, division og potensregning. Vi kommer til å representere disse fem operasjonene respektivt med symbolene `+-*/^`.

a)

En kalkulasjon består av en start verdi og en rekke operasjoner, hver bestående av et symbol som representerer operasjonen og operanden på høyre side i operasjonen. For eksempel vil den enkle kalkulasjonen **1 + 2** bli representert med start verdi **1** og operasjonen **(+, 2)**. På samme måte blir kalkulasjonen **1 * 2 + 3 - 4 / 5** representert med start verdi **1** og operasjonene **(*, 2)**, **(+, 3)**, **(-, 4)** og **(/, 5)**.

Lag en klasse "Operation" som kan representere en operasjon på formen gitt over. Det burde være mulig å hente ut både tegnet som representerer operasjonen og tallet når som helst, men kun kunne sette disse i konstruktøren.

b)

Nå som vi har en måte å representere operasjoner på, kan vi lage en klasse som representerer en hel kalkulasjon. Altså start verdien og alle operasjonen.

Lag en klasse "Calculation" som representerer en kalkulasjon på måten gitt over. Det burde være mulig å hente ut både initial verdien og operasjonene. Det burde være mulig å både kunne gi inn en liste med operasjoner i konstruktøren, samt å kunne gi disse inn én etter én i etterkant.

c)

Vi representerer kalkulasjonen **1 * 2 + 3 - 4 / 5** som start verdi **1** og operasjonene **(*, 2)**,

(+, 3), **(-, 4)** og **(/, 5)**. Det er ikke nødvendigvis så lett å se hva kalkulasjonen vi skal utføre egentlig er, når den står på denne formen.

Lag en toString metode i "Calculation" klassen, som gir ut en fin representasjon av kalkulasjonen

Kjøring av følgende **main**-metode skal skrive ut: `10.00 + 10.00 * 3.14 / 12.00 ^ 13.12 - 14.14 - 12.12` eller noe lignende (avhengig av hvilken presisjon som velges for flyttall)

```
public static void main(String[] args) {
  Calculation calculation = new Calculation(10, Arrays.asList(new
Operation('+', 10), new Operation('*', 3.14)));
  calculation.addOperation(new Operation('/', 12));
  calculation.addOperation(new Operation('^', 13.12));
  calculation.addOperation(new Operation('-', 14.14));
  calculation.addOperation(new Operation('-', 12.12));
  System.out.println(calculation);
}
```

d)

Til nå har vi kun laget klasser for representasjon, og ikke skrevet noe kode for utregning av kalkulasjonene. Hver operasjon burde kunne regne ut resultatet sitt hvis det får oppgitt operanden som skal stå på venstre side i operasjon.

Lag en metode i "Operation" klassen som tar inn operanden på venstre side i operasjonen og gir ut resultatet av denne operasjonen. Pass også på at det ikke er mulig å bruke noen andre operasjoner enn de fem som er definert over.

Kjøring av følgende **main**-metode skal gi resultatet: `2.0`

```
public static void main(String[] args) {
  Operation operation = new Operation(‘/‘, 5);
  System.out.println(operation.perform(10));
}
```

e)

Vi har nå alt som skal til for å regne ut resultatet av en kalkulasjon.

Lag en klasse "Calculator" med en metode for å regne ut resultatet av en kalkulasjon. Du kan anta at alle operasjoner har like presedens. Det vil si at du kan utføre de fra venstre til høyre.

Kjøring av følgende **main**-metode skal gi resultatet: `2.693844814120271E9`

```
public static void main(String[] args) {
  Calculator calculator = new Calculator();
  Calculation calculation = new Calculation(10);
  calculation.addOperation(new Operation('+', 10));
  calculation.addOperation(new Operation('*', 3.14));
  calculation.addOperation(new Operation('/', 12));
  calculation.addOperation(new Operation('^', 13.12));
  calculation.addOperation(new Operation('-', 14.14));
  calculation.addOperation(new Operation('-', 12.12));

  System.out.println(calculator.calculate(calculation));
}
```

f)

Det er ofte slik at forskjellige operasjoner har ulik presedens. Det vil si at noen operasjoner skal utføres før andre.

Endre "Calculator" klassen slik at den i konstruktøren tar inn en liste av strenger som representerer presedensen til operatorene. Jo tidligere i listen en streng kommer, desto tidligere skal operasjonene i strengen utføres. Operasjoner i samme streng har samme presedens. Endre kalkulasjonsmetoden til å følge presedensen gitt av listen.

Kjøring av følgende **main**-metode skal gi resultatet: `-16.25999999999978`

```
public static void main(String[] args) {
  Calculator calculator = new Calculator(Arrays.asList("^", "*/", "+-"));
  Calculation calculation = new Calculation(10);
  calculation.addOperation(new Operation('+', 10));
  calculation.addOperation(new Operation('*', 3.14));
  calculation.addOperation(new Operation('/', 12));
  calculation.addOperation(new Operation('^', 13.12));
  calculation.addOperation(new Operation('-', 14.14));
  calculation.addOperation(new Operation('-', 12.12));

  System.out.println(calculator.calculate(calculation));
}
```