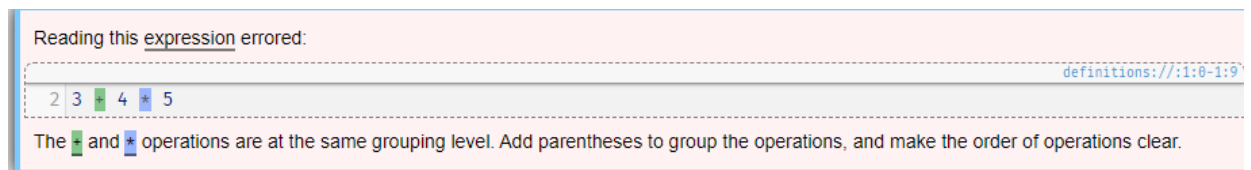


Progoblig 01 individuell del

Oppgave Ind01

- a) I uttrykket $3 + 5$ vil $+$ -tegnet (addisjon) være operatøren mer spesifikt skjent som en matematisk operatør. Operanden i tilfellet vil være tallene 3 og 5 dette er verdiene operasjonen utføres på.
- b) De mest grunnleggende aritmetiske operatørene er: Addisjon (+), Subtraksjon (-), Multiplikasjon (*) og Divisjon (/).
- c) Utrykket $3 + 4 * 5$ vil bli tolket slik av en kompilator: Multiplikasjonen blir utført først ($4 * 5 = 20$), deretter blir 3 lagt til resultatet av multiplikasjonen ($3 + 20 = 23$).
- d) Fortolkeren i Pyret (CPO) vil gi en feilmelding når du gir regnestykket $3 + 4 * 5$. Dette er på grunn av at fortolkeren ikke har definert en operatør-prioritet som for eksempel JavaScript har. Dette gjør at man må spesifisere hvilke operatører som skal utføres først. Dette kan gjøres ved hjelp av parenteser: $3 + (4 * 5)$.



- e) JavaScript fortolkeren JSF har allerede definert en prioriterings-rekkefølge av operatørene, i henhold til matematikk. Av den grunn vil man ikke trenge å spesifisere hvilke operatører som skal kjøre først.
- f) Forskjellen er som tidligere forklart at Pyret-fortolkeren har ikke definert operatør-prioriteten i programmeringsspråket noe JavaScript har.
- g) Både i JavaScript og Pyret kan man legge til parenteser for at fortolkeren i JavaScript skal overstyre operatør-prioriteringen og i fortolkeren til Pyret skal skjønne hvilke du vil regne først dersom den ikke har en prioritering. $3 - (4 + 1)$.
- h) De blir behandlet som aritmetiske operatører for verdier som for eksempel tall eller strenger
- i) De blir også behandlet som aritmetiske operatører på samme måte som addisjon og subtraksjon, men blir prioritert ovenfor disse på grunn av operatør-prioriteringen

- j) $5 + 1 - 1 * 10 / 2 = 1$. Her tolker JSF regnestykke i henhold til operatør-prioriteringen altså gange og dele først og så addisjon og subtraksjon.
- k) Fordelene er at brukeren har full kontroll og oversikt over hvilken rekkefølge operatører skal prioriteres. Ulemper er at det tar lenger tid på grunn av ekstra kode og uriktig rekkefølge ifølge matematiske standarder.
- l) Du kan bruke funksjoner som **num-min**, **num-max** eller **num-random** for å utføre andre typer beregninger. Typisk bruk av **num-min** eller **num-max** kan være å finne minste eller største verdi i en liste. Eksempel:
- ```
var numbers = [5, 10, 2, 8, 15]
var maxVal = num-max(numbers)
```
- maxVal vil nå inneholde 15, som er det største tallet i listen*
- m)

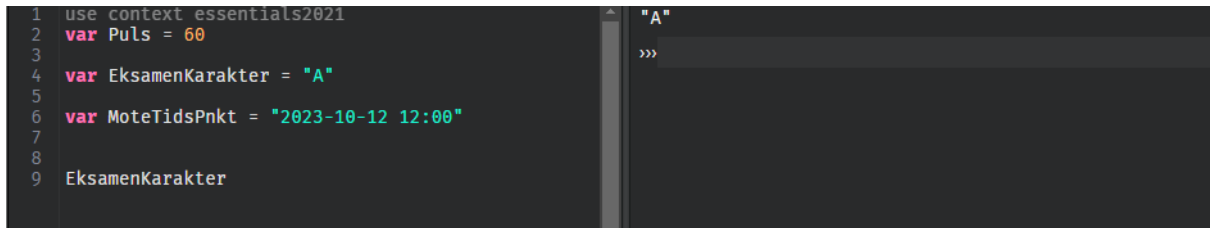
|                                               |         |
|-----------------------------------------------|---------|
| 3 + 4 - num-min(7, 4)                         | Uttrykk |
| 3 + 4<br>3 - 4<br>num-max(-4, 4)<br>"hei"     | Program |
| "IS-114"                                      | Verdi   |
| 3.141592653                                   | Verdi   |
| "hei på deg"                                  | Verdi   |
| "håper det går bra med deg"<br>'3'<br>2 + '3' | Program |

- n) JSF vil klare å tolke uttrykk som for eksempel  $5 + 5$ . JSF vil imidlertid ikke klare å tolke en verdi som en streng, da denne har en semantisk sammenheng. Et unntak er True og False verdier. JSF kan ikke tolke

programmer i tradisjonell forstand da den er laget for å håndtere brukergrensesnittlogikk.

- o) Noen tall kan ikke bli uttrykt 100% presist med en datamaskin da denne bruker binærsystemet for å bryte ned desimalsystemet mennesker bruker for å gjøre matematiske beregninger.

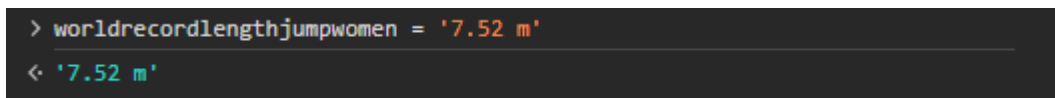
## Oppgave Ind02



```
1 use context essentials2021
2 var Puls = 60
3
4 var EksamenKarakter = "A"
5
6 var MoteTidsPnkt = "2023-10-12 12:00"
7
8 EksamenKarakter
9
```

The console on the right shows the output: "A"

- a)
- b)
1.  $5 + 8 =$  **uttrykk fordi det er en kombinasjon av operatør og operander som evalueres til en ny verdi**
  2.  $x = 14 + 16 =$  **Definisjon fordi den oppretter en navngitt indikator (variabel)**
  3. `triangle(20, "solid", "purple") =` **Utrykk fordi den buker en kombinasjon av funksjon, verdier og beskrivelser**
  4. `blue-circ = circle(x, "solid", "blue") =` **Definisjon fordi den navngir en blokk med kode uten at koden blir regnet ut eller brukt på noen måte**
- c) Du vil få en feilmelding på grunn av at du har to definisjoner med samme navn som har motstridene innhold.
- d) Man kunne lagd to definisjoner med forskjellige navn, en med det metriske systemet og en med det amerikanske
- e) I JSF får man en syntaks feilmelding. For å fikse dette må man fjerne alle bindestreker i navnet på definisjonen og man må bruke enkle 'hermetegn' i stedet for ``doble``.



```
> worldrecordlengthjumpwomen = '7.52 m'
```

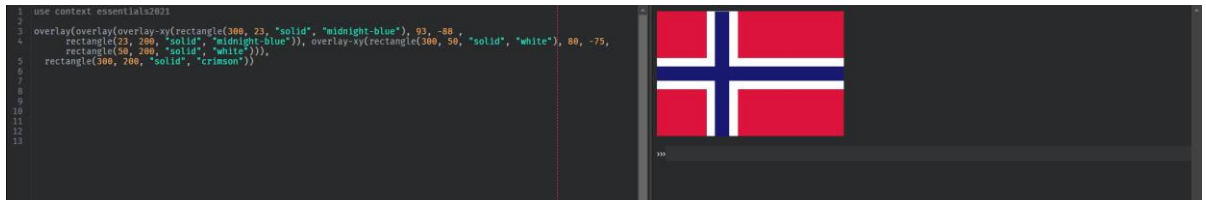
The console on the right shows the output: < '7.52 m'

- f) Definisjons-vinduet i CPO er der man skriver programmet eller definisjonene. Interaksjons-vinduet derimot er der koden eller

programmet blir kjørt og vist. Dette gjøres via «run-knappen» på toppen av interaksjons-vinduet.

- g) Det norske nasjonalflagget genereres ved hjelp av et par innebygde funksjoner i Pyret. For å skape alle elementene bruker jeg rektangler, dette skapes ved funksjonen: ***rectangle(width, height, mode, color)***. Videre er må man legge alle rektanglene på hverandre i en viss rekkefølge der det blå korset er øverst også videre. Funksjonen som hjalp med dette heter ***overlay()*** i tillegg til at ***overlay x-y()*** blir brukt for å plassere korsene på riktig plass. Her gjelder det å bruke flere overlay'er for å få ønsket resultat.

<https://github.com/SimonPortillo/ProgObligInd01> (link til github repository)



## Bibliography

**There are no sources in the current document.**