

Navn: Sigurd Munk Brekke

Gruppenummer: 8A

Emnekode:	<i>UIA IS-114</i>
Emnenavn:	<i>Samskaping, kommunikasjon og Prosjektarbeid</i>
Emneansvarlig (faglærer):	<i>Janis Gaillis og Niels F. Garmann-Johnsen</i>
Eventuell veileder:	<i>Sudeepika Wajirakumari Samarat Liyanapathirana</i>
Innleveringsfrist/tidspunkt:	<i>23.10.2023</i>
Antall ark inkl. denne forside:	<i>12</i>
Merknader/Tittel på oppgaven:	<i>Obligatorisk Innlevering 1 - individuell</i>

Jeg/vi bekrefter at jeg/vi ikke siterer eller på annen måte bruker andres arbeider uten at dette er oppgitt, og at alle referanser er oppgitt i litteraturlisten.

Ja ☒

Nei ☐

kildehenvisning,

Kopiering av andres tekster eller annen bruk av andres arbeider uten kan bli betraktet som fusk.

Kan besvarelsen gjenbrukes til forskning og undervisningsformål, og i evt. publiseringer?

Ja ☒

Nei ☐

Innhold

Figurliste	3
Prosjektblig 01 – individuell del 1.....	4
Prosjektblig 01 – individuell del 2.....	8
GitHub link til Repository med kode.	11
Referanseliste.....	12

Figurliste

Figur 1, fra eget arbeid i code.pyret.org	6
Figur 2: Eget arbeid med JSF på jsconsole.com.....	7
Figur 3: Eget arbeid med CPO, fra code.pyret.org.....	8
Figur 4: Eget arbeid i pyret på code.pyret.org.....	9
Figur 5: Eget arbeid med JSF på jsconsole.com.....	10
Figur 6: Eget arbeid med CPO, på code.pyret.org.....	10

Prosjektoblig 01 – individuell del 1

a) *Finn operatører og operander i uttrykket $3 + 5$*

Operatøren er multiplikasjonstegnet og operandene er verdiene 3 og 5.

b) *Nevn 4 grunnleggende operatører*

Addering (+), subtraksjon (-) multiplisering (*) og dividering (/)

c) *Hvordan kan uttrykket $3 + 4 * 5$ tolkes av en kompilator på en datamaskin?*

Uten parentes klarer ikke CPO å lese det, på utviklerværktøy vil vet ikke maskinen hva som skal svares fordi operatørene ikke er gruppert, dette gjøres med parentes.

d) *Forklar resultatet som CPO returnerer*

Som nevnt, vil ikke CPO klare å lese dette uten parentes, og må da skrives $(3 + 4) * 5$. Så lenge operatørene er på samme linje og uorganisert vil ikke CPO-fortolkeren lese – i motsetning til en kalkulator, som først adderer, så multipliserer. Dette skjønner ikke fortolkeren den skal gjøre, derfor er det viktig å presentere operatører i grupperinger, i dette tilfellet ved hjelp av parentes.

e) *Forklar resultatet av evalueringen av uttrykket på punkt b) med en JSF,*

Uten parentes fortolker JSF (jsconsole.com) å fortolke det til et svar som tilsvarer 23, hvor det rett og slett har blir $3 + 20$. Legger en til parentes blir svaret 35, slik det også blir på CPO.

f) *Hva er forskjeller på hvordan $3 - 4 + 1$ blir fortolket i CPO og JSF, og hvorfor?*

Svaret blir 0 JSF. JSF klarer dette uten noe problem, også når det er skrevet rett frem. CPO derimot, trenger ennå grupperinger, for å fungere og for i det hele tatt få et svar av fortolkeren, og må da skrives $(3 - 4) + 1$.

g) Hvordan må man endre uttrykket fra punkt e) for at resultatet skal være -2 i både CPO og JSF?

I både CPO og JSF må en endre uttrykket til $3 - (4 + 1)$ for å få svaret -2.

h) Hva kan man si om hvordan operatørene + og – blir behandlet i JSF?

JSF tillater en forenklet (i mange tilfeller feil) metode for å finne svaret, eksempelvis hvor svaret i punkt e) blir 23 uten parentes. Den leser altså verdiene rett frem, og finner svaret deretter.

i) Hva kan man si om hvordan * og / operatørene blir behandlet i JSF?

Som nevnt i h) vil også JSF lese disse verdiene rett frem, men behandler addering og subtraksjon som en ettertanke.

j) Lag et uttrykk som inneholder alle 4 aritmetiske operatørene og forklar hvordan JSF takler det. Valgt ligning: $15 - 2 * 3 / 5 + 2$

Uten parentes vil svaret bli 15,8 – og med vil svaret bli 9,8. Det blir da skrevet $(15 - 2) * (3 / 5) + 2$. Uansett om en inkluderer en parentes til i ligningen vil svaret bli 9,8. JSF følger de vanlige mattereglene, løse parentesene, dividere, subtrahere og addere. Uten parentes vil den først multiplisere, dividere, addere og subtrahere. Siden Adderingen ikke er i en parentes, vil da altså +2 ikke påvirke ligningene innenfor parentesene før de er løst.

k) Hva er fordelene og ulempene med måten CPO fortolker aritmetiske uttrykk med alle de 4 grunnleggende aritmetiske operatørene?

Ulempe: Mens JSF fortolker i en annen rekkefølge (løse parenteser, multiplisere, dividere, addere og subtrahere, i den rekkefølgen) gjør ikke CPO det. Om en bruker ligningen valgt i j), vil en først få en feilkode hvor en blir bedt om å spesifisere parenteser på operasjonene, for å gjøre det klart for CPO hvilken rekkefølge det skal gjøres. Om en da skriver $(15 - 2) * (3 / 5) + 2$ vil den ennå ikke løse mattestykket, da «*» og «+» er utenfor parentesene. I CPO må en da gjøre slik, for å finne svaret:

```

1 use context essentials2021
2
3 15 - 2
4 3 / 5
5 13 * 0.6
6 7.8 + 2

```

Results:

- 13
- 0.6
- 7.8
- 9.8
- »»

Figur 1, fra eget arbeid i code.pyret.org

Fordel: CPO vil alltid forlange parentes for å gruppere, noe som lar en løse avanserte ligninger så lenge en har læret seg hvordan programmet skal brukes.

l) *Hvordan kan man få regnemaskinen til å gjøre andre beregninger enn de grunnleggende aritmetiske operatørene i CPO?*

Ved å skrive eksempelvis «num-max(1,7)» eller «num-min(-4,4)» vil både CPO gjøre funksjonene gitt ved disse, i dette tilfellet vil num-max gi verdien 7 i svar, og num-min gi -4. Fordi disse er de høyeste og laveste gitte verdiene i beregningen.

m) *Sorter følgende eksempler i forhold om de er uttrykk, en verdi eller program*

$3 + 4 - \text{num-min}(7,4)$	uttrykk
$3 + 4$	uttrykk
$3 - 4$	uttrykk
$\text{Num-max}(-4,4)$	Utrykk
«IS-114»	verdi
3.141592653	verdi
«Hei på deg»	Verdi
«håper det går bra med deg»	Verdi
'3'	Verdi
$2 + 3$	uttrykk

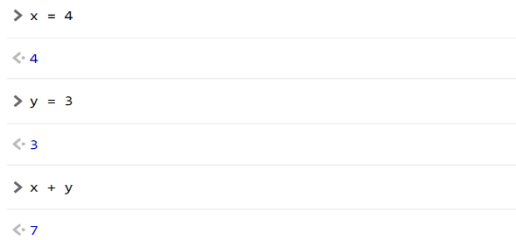
n) Kan JSF fortolke et uttrykk? En Verdi? Et Program? Forklar og vis med eksempler.

JSF kan fortolke et uttrykk, som $3 + 5$ lett, verdier kan den lese ved å definere verdiene. Som å skrive:

$X = 4$

$Y = 3$

$X + Y = 7$



```
> x = 4
< 4
> y = 3
< 3
> x + y
< 7
```

Figur 2: Eget arbeid med JSF på jsconsole.com

Nå har JSF fortolket uttrykket mitt og gjort det om til en verdi, og vil kunne gi et svar når en skriver $x + y$.

o) Sammenligne hvordan JSF og CPO tolker uttrykket $0.1 + 0.2$.

CPO presenterer svaret uten flere desimaltall enn nødvendig, og blir derfor 0.3. JSF derimot gir svaret 0.30000000000000004. Schwarzmüller skriver i sin artikkel om dette problemet at dette skyldes at datasystemer bruker binært tallsystem, som skyldes at desimaltallene lagres som binære tall. Grunnen til dette er fordi noen tall som kan være nøyaktige i noen tallsystemer, ikke nødvendigvis blir presentert like presist i andre systemer som i dette tilfellet er binærtall «oversatt» til desimaltall. (academind, 2021).

CPO ser ut til å være programmert til å vise færrest mulig desimaltall, eller runde opp, i motsetning til JSF.

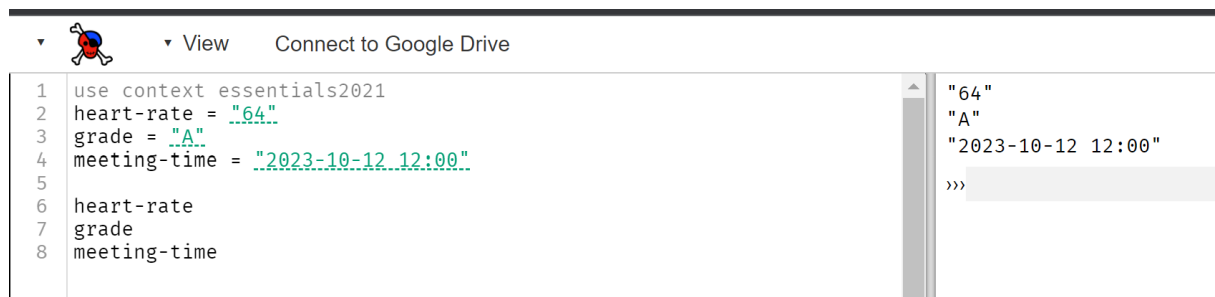
Prosjektoblig 01 – individuell del 2

- a) *Gitt følgende verdier: 64, «A», «2023-10-12 12:00». Navngi variabel for å representere hjerterytme eller puls, en eksamenskarakter, tidspunkt for møte.*

heart-rate = "64"

grade = "A"

meeting-time "2023-10-12 12:00"



Figur 3: Eget arbeid med CPO, fra code.pyret.org

Ved å skrive inn eksempelvis «heart-rate» i CPO nå vil en få i resultat "64" når man trykker på «Run» på CPO. Viktig å merke at en ikke må bruke bindestrek. Dette er valgt fordi det er mer oversiktlig.

- b) *Hvilke av disse er definisjonene og hvilke er uttrykk (tall foran hver linje er ikke en del av koden) i CPO. Begrunn*

1. $5 + 8$
2. $x = 14 + 16$
3. `triangle(20, "solid", "purple")`
4. `blue-circ = circle(x, "solid", "blue")`

1. Er et uttrykk, fordi $5 + 8$ skaper et resultat uten videre definisjoner
2. dette er en definisjon. Fordi en definerer X for CPO.
3. Dette er et uttrykk, som vil skape en trekant i CPO.

4. Dette vil være en definisjon, da den ikke vil skape en sirkel før X er en definert verdi. På samme måte kan en argumentere for at det er et uttrykk, når X er blitt definert.

c) *Hva skjer når man skriver inn to påfølgende definisjoner med det samme navnet i CPO? Forklar*

World-record-length-jump-women = “7.52 m”

World-record-length-jump-women = “24 ft 8 in”

Det som skjer er at blir 2 programmer med samme navn, og om en prøver å trykke «run» får man en feilmelding. CPO kaller dette en konflikt blant deklarerte navn, og vil ikke printe et svar.

d) *Kunne man skrive definisjonene om, slik at CPO responderte annerledes enn i punkt c)?*

Siden disse er presentert i to forskjellige metriske system, kan en ta utgangspunkt i. Min måte å løse dette på ville være å skille disse basert på de presenterte metriske systemene ved å gjøre:

World-record-length-jump-women-meters = “7.52”

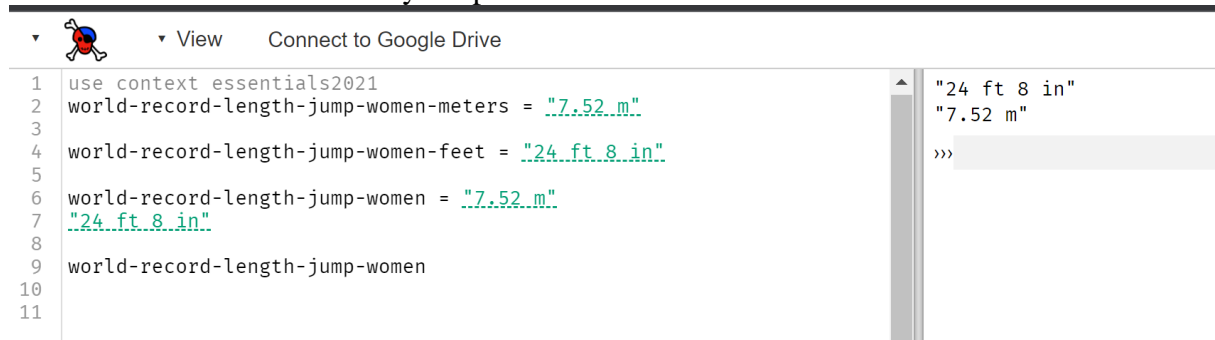
World-record-length-jump-women-feet = “24 ft 8 in”

Ennå vil ikke disse vises om man skriver begge inn samtidig, fordi de nå er programmert til å reagere på spesifikke definisjoner. Dette kan løses ved å skrive dette på CPO:

World-record-length-jump-women = “7.52 m”

“24 ft 8 in”

Resultatet blir da dette ved å trykke på “Run”:



```
1 use context essentials2021
2 world-record-length-jump-women-meters = "7.52 m"
3
4 world-record-length-jump-women-feet = "24 ft 8 in"
5
6 world-record-length-jump-women = "7.52 m"
7 "24 ft 8 in"
8
9 world-record-length-jump-women
10
11
```

The right-hand pane shows the output of the code, displaying the two conflicting definitions: "24 ft 8 in" and "7.52 m".

Figur 4: Eget arbeid i pyret på code.pyret.org

e) Utføre definisjoner i JSF. Sammenligne resultater med resultater i CPO. Forklar

Første merkbare forskjell er at JSF ikke tillater bindestrek for å definere, men dette løses med å heller ha store bokstaver mellom hvert ord, og at det heller ikke kan være mellomrom. Det blir da seende slik ut:

```
WorldRecordLenghtJumpWomenMeters = "7.52"
```

```
WorldRecordLenghtJumpWomenFeet = "24ft 8in".
```

Disse blir da også vist frem når man gir en definisjon til et uttrykk. Det problemet som forekommer, er at i motsetning til CPO er det noe vanskeligere å få vist disse på en felles liste. Min løsning er å bruke array, for å få opp en liste, ved å skrive [0] eller [1] etter WorldRecordLenghtJumpWomen vil en få opp enten i meter, eller i feet.

```
WorldRecordLenghtJumpWomen = ["7.52m", "24ft 8in"]
```

Dette er slik det ser ut på JSF:

> WorldRecordLenghtJumpWomen

< Array (2) ["7.52m", "24ft 8in"]

> WorldRecordLenghtJumpWomen[0]

< "7.52m"

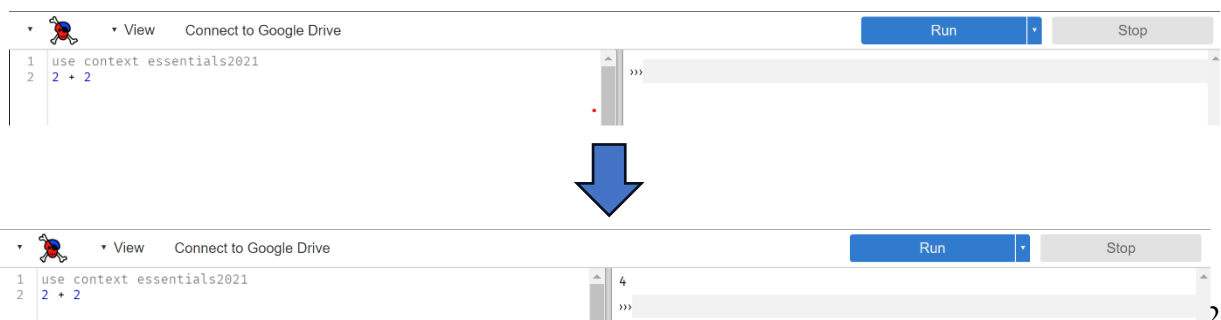
> WorldRecordLenghtJumpWomen[1]

< "24ft 8in"

Figur 5: Eget arbeid med JSF på jsconsole.com

f) Forklar hvordan definisjons- og interaksjons-vinduene fungerer i CPO.

På CPO, når man skriver en linje med kode vil den ikke automatisk dukke opp etter å ha gått inn i en ny linje på definisjonsvinduet. Derfor må man trykke på «Run» før man eventuelt får funksjonen over på den neste fanen. Det vil si, skriver en 2+2 i definisjonsvinduet, vil det ikke dukke opp i interaksjonsvinduet før man trykker run.



Figur 6: Eget arbeid med CPO, på code.pyret.org

g) Skriv og forklar koden som produserer Norges handels- og nasjonalflagg i interaksjonsvinduet til CPO når man trykker «Run».

I henhold til norske Flaggløver om proporsjoner må en ha høydeforholdene 6:1:2:1:6 og breddeforholdene 6:1:2:1:6:11, til sammen blir dette 16:22 i størrelsesforhold. Jeg valgte høyden 200, og bredden ble dermed 275 ($200/16 * 22$). Dette er hele og enkle tall og jobbe med, og vil gjøre prosessen lettere.

Koden er i stor grad skrevet baklengs, da jeg har tatt i bruk «overlay-xy» funksjonen til CPO. Overlay-xy funksjonen fungerer slik hvor et bilde vil legge seg over et annet. Ved å bruke «rectangle(275, 200, «solid», «red»)» som første linje i koden ville ikke noen andre farger vise seg. Dette er fordi overlay-xy leser det første rektangelet som skrives inn som det som skal ligge øverst. Derfor måtte koden til dette flagget skrives baklengs, hvor rød fungerer som grunnfargen, og dermed nevnes sist.

Det er også gjort relativt enkelt ved å ha det hvite korset i 4:16 i høyde og 4:22 i bredde. Flaggløvene forlanger egentlig at den hvite delen av korset skal være 1:16 og 1:22, men siden dette er funksjonen overlay-xy vil det være bedre å lage to hvite rektangel som er dobbelt så store som de blå rektanglene, slik det hvite ser ut som om et er 1:16 / 1:22, og ikke 4:16 / 4:22. Siden det blå korset er nevnt først, vil det legge seg over både det hvite korset og den røde grunnfargen, og siden det legger seg over det hvite korset, vil dette resultere i et proporsjonalt og symmetrisk riktig flagg, i henhold til norske flaggregler.

Overlay-xy lar også en bestemme hvor posisjonen til (i dette tilfellet) rektanglene skal være. Dette gjøres ved å bestemme koordinatene via X og Y aksen. Verdien på hver enhet ved kartet er gjort ved å dele med den valgte bredde med breddeproporsjonen altså $275 / 22 = 12,5$. Med dette, kan jeg nå plassere rektanglene riktig, og CPO vil også forstå hvor skal være, da jeg har tatt i bruk overlay-xy funksjonen.

GitHub link til Repository med kode.

<https://github.com/sigurdbrekke/prosjektoblig01/tree/main>

Referanseliste

Jsconsole.com (2023). *Jsconsole.com*. Hentet 16. Oktober 2023 fra <https://jsconsole.com/>

The Pyret Crew (2023) *code.pyret.org*. Hentet 16. oktober 2023 fra <https://code.pyret.org/editor>

The Pyret Crew, (uten år). The *Image Library* på pyret.org. Hentet 16. Oktober 2023 fra <https://pyret.org/docs/latest/image.html#%28part..Overlaying..Images%29>

Roede, L. (2023). *Norges flagg i Store norske leksikon*. Hentet 21. oktober, 2023 fra https://snl.no/Norges_flagg

Schwarz Müller, M. (2021) *Floating Point Arithmetic & Precision*. Hentet 17. oktober 2023 fra <https://academind.com/tutorials/floating-point-precision-and-arithmetic>