INF2810 OBLIG 1

1

a) (* (+ 2 2) 5) evalueres til 20
b) (* (+ 2 2) (5)) gir feilmelding ettersom vi ikke gir noen opperasjon til (5).
c) (* (2 + 2) 5) gir og feilmelding ettersom vi prøver å beskrive operasjonen direkte, men i scheme angir vi hvilke operasjon vi øsnker å kjøre førtst, deretter angir vi argumentene.
d) (define bar (/ 4 2)) bar evalueres til 2 og definerer “bar” til verdien 2
e) (- bar 2) evalueres til 0 etter som “bar” har verdien 2
f) (/ (* bar 3 4 1) bar) evalueres til 12

2

a)

“(or (= 1 2)
"piff!"
"paff!"
(zero? (1 - 1)))” Evalueres til “piff!”, vi ber om at det er piff som skal printes dersom 1 = 2 eller paff er sant. Dermed får vi ut piff. Den syntaktisike feilen ligger i (zero? (1 - 1)) som skulle vært definert som følger (zero? (- 1 1)).

“(and (= 1 2)
"piff!"
"paff!"
(zero? (1 - 1)))” evalueres til false, vi ber om at både 1 = 2 er sant, samtidig som “paff!” er sann. Ettersom 1 ≠ 2 så stoppes vi allerede her og det blir printet #f for false.

“(if (positive? 42)
"poff!"
(i-am-undefined))” Evalueres til “poff!”, vi ber om å få “poff” dersom 42 er et positivt tall. Ellers vil vi kjøre “i-am-undefined” som vi desverre ikke har definert. Så dersom tallet vi mater inn ikke er positivt vil vi få en feilmelding på at funksjonen vi ønsker å kjøre ikke er definert.

if, and, og or er special forms for det de bryter med standard scheme evaluering. Normalt regner vi ut alle under uttrykkene før vi regner ut det lverste uttrykket som er gitt. Men når vi bruker and, or og if sjekker vi det aller første vi gir inn først og tar en beslutning basert på det.