Q1 (a) ((six equals (two plus one)) → one [] (three minus one)) plus two

Sol^n^

= ((six equals three) → one [] (three minus one)) plus two

[Applying choice]

= (three minus one) plus two

= (two) plus two

= Four [Ans]

(b) (two equals (true → one [] two)) and true

Sol^n^ = (two equals (one)) and true

= false and true

= false [Ans]

(c) not (false) → not(true) [] not(true)

sol:-

true → false [] false

= false [Ans]

Q2

(a) new_array $(m_0, m_1) = \lambda n_0 . m_0 \; \lambda n_1 \, m_1 . \; \lambda n_2$ Error λn_3 Error . . .

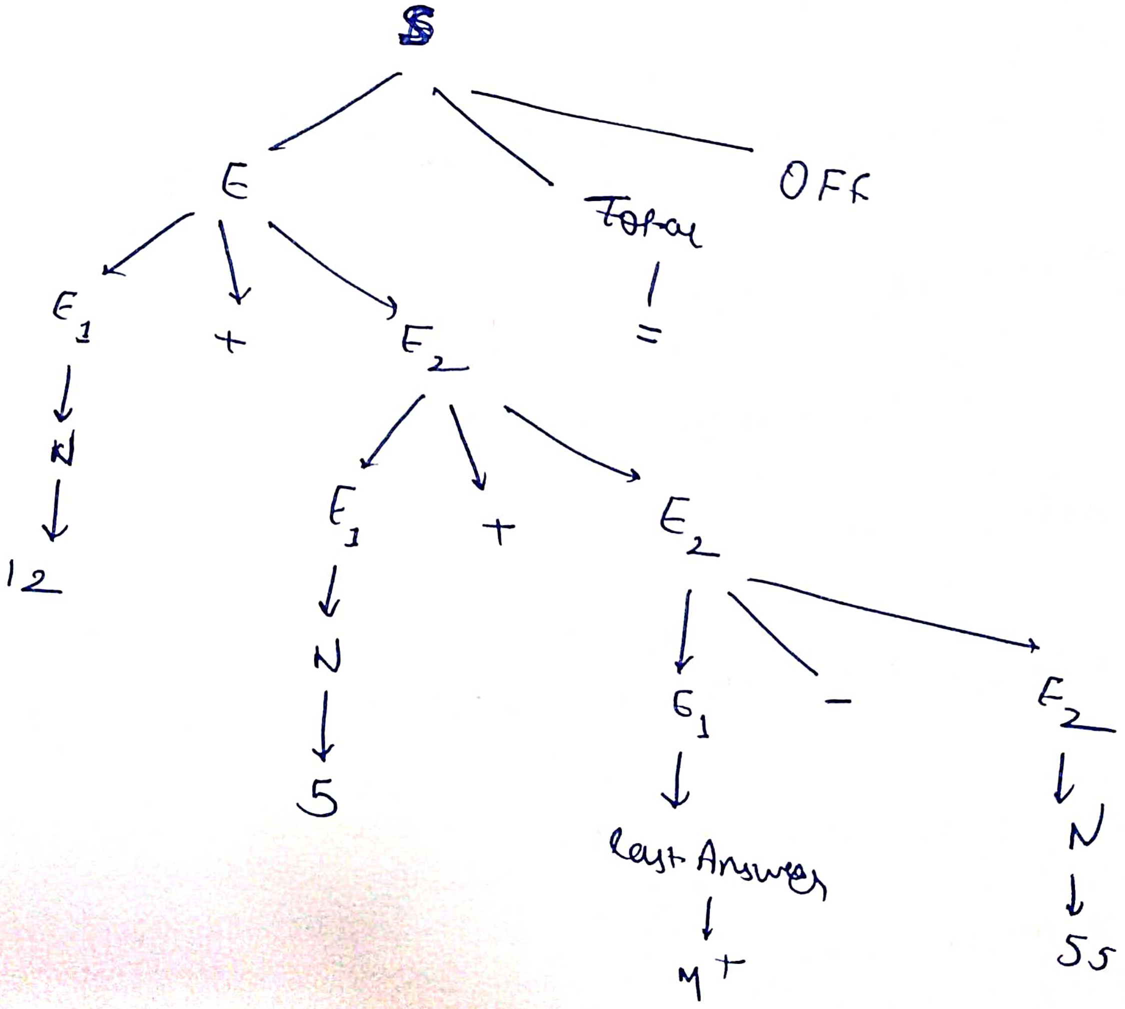
where m_0 & m_1 are lower & upper limit respectively

(b) access$(n, r) =$ if $r(n) \geq m_0$ and $r(n) \leq m_1 \rightarrow r[n]$ [] error

update $(n, v, r) =$ if $(r(n) > m_0$ and $r(n) < m_1 \rightarrow [n \mapsto v]\, r$

[] $r(n)$

Q3



Q4 (a)

(i) update-payrate(pay, employee) =

(employee ↓ 1, ((case (employee ↓ 2) of isDay(dwage)

→ isDay(pay) [] isNight (nwage) → isNight (pay) end)

employee ↓ 3)

(ii) update-hours = (employee ↓1, employee ↓2, ~~empto~~
(hours, employee)

hours addat employee ↓ 3)

(b)

(i) newemp (Jan_Doe) = (Jane_Doe, inDay (minimum_way), 0)

(ii) move-to-Nightshift(~~on~~ Jan_Doe)

= (Jan-Doe, (case (inDay(minimumWage) of isDay(dwage)

→ inNight(dwage) [] isNight (nwage) → inNight(nwage)

end), 0)

= (Jan-Doe , inNight (minimum_wage), 0) viii)

update_hours (makerat(38,1), Jan_Doe) =

(Jan_Doe, isNight(minimum_wage), makerat(38,1)
addrat 0)

= (Jan_Doe, isNight (minimum_wage), makerat (38,1))

ix) update_pay (~~Mo~~ 9.00, Jan_Doe)

= (Jan_Doe , isNight (9) , makerat(38,1))

⇒ Compute_pay (Jan_Doe)

= (cases (isNight(9.00) of is Day (dwage) → dwage multrat(makerat
(38,1))

[] is Night(nwage) → nwage(mult rate 1.5)
multrat (makerat(38,1) end)

= (9.00 multrat 1.5) multrat (makerat (38,1))end

= makerat (27, 2 mult-rat, makerat (38, 1)

= makerat(513, 1)