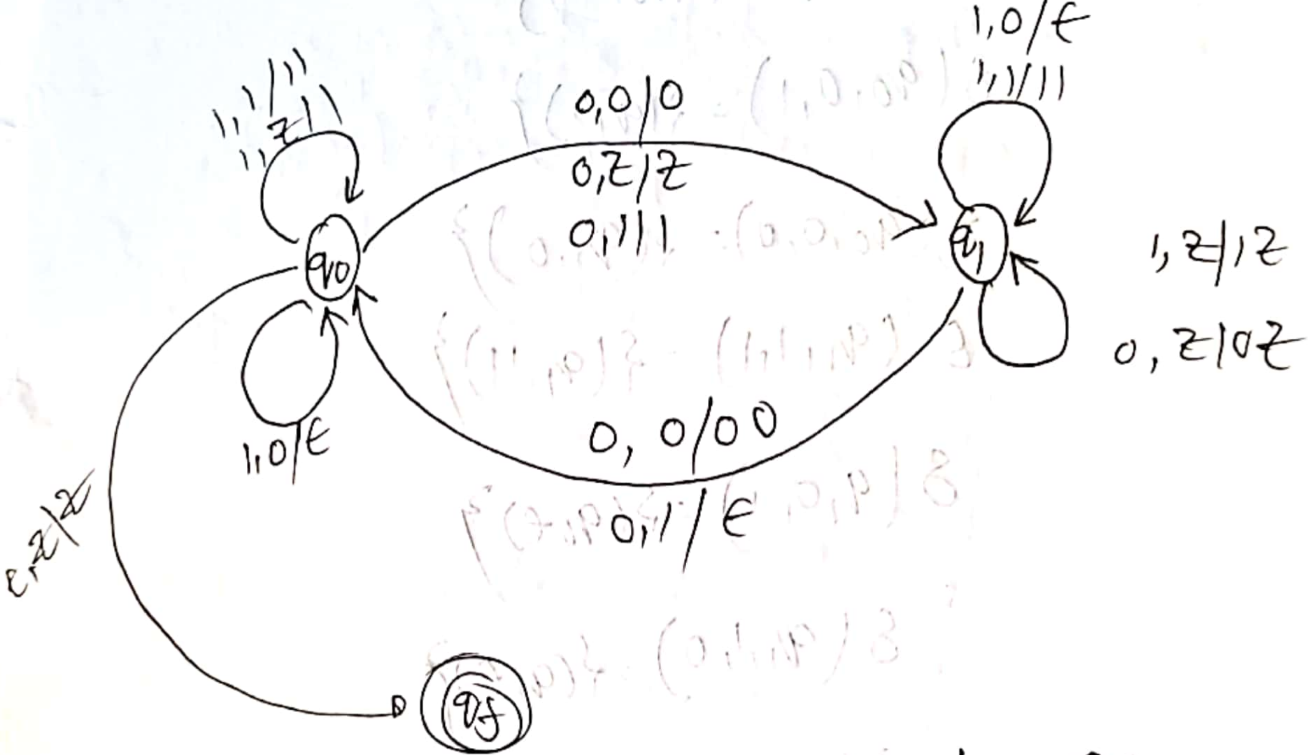
(a)

1. The set of all strings with twice as many 0's as 1's.

So, it can receive any string having double zeros than ones.



$q_0 \to$ state having even no. of zeros

$q_1 \to$ state receiving & odd no. of zeros let the PDA be

$$P(\{q_0, q_1\}, \{0, 1\}, \{0, 1, Z\}, \delta, q_0, Z)$$

q_f

$\delta(q_0, \overset{1}{\cancel{\epsilon}}, Z) = \{(q_0, \overset{1}{\cancel{0}}Z)\}$

$\delta(q_0, 0, Z) = \{(q_1, Z)\}$

$\delta(q_0, 1, 1) = \{(q_0, 11)\}$

$\delta(q_0, 0, 1) = \{(q_1, 1)\}$

$\delta(q_0, 0, 0) = \{(q_1, 0)\}$

$\delta(q_1, 1, 1) = \{(q_1, 11)\}$

$\delta(q_1, 0, 1) = \{(q_0, \epsilon)\}$

$\delta(q_1, 1, 0) = \{(q_1, \epsilon)\}$

$\delta(q_1, 0, 0) = \{(q_0, 100)\}$

$\delta(q_1, 1, Z) = \{(q_1, 1Z)\}$

$\delta(q_1, 0, Z) = \{(q_1, 1Z)\}$

$\delta(q_1, 0, Z) = \{(q_1, 0Z)\}$

$\delta(q_0, \epsilon, Z) = \{\cancel{(q_f, \epsilon)}\}$

2 (c)

$H(m)$ ::= M to be the set of inputs w such that M halts given input w, regardless of whether or not M accepts w. The halting problem is the set of pairs (M, w) such that w is in $H(M)$

we can construct a Turing Machine U_H similar to the Universal Turing Machine. U_H takes as input a Turing machine M and a binary string v. U_H simulates the actions of M on input w. If M halts, U_H accepts and halts (regardless of whether M accepts or rejects). But if M keeps running, U_H also keeps running

we know, L_U is not recursive. Suppose, for contradiction, the halting problem is recursive. If we want to determine whether a ~~to~~ turing machine M

accepts w. we can do the following:

→ determine whether M halts on input w
(we've assumed that it is decidable)

→ If it does execute M on w and see

whether M halts, it does so in an

accepting state. if so, accept (M, w).

→ If M does not halt on w, reject

(M, w))

thus we could test whether M accepts w by using our assumed algorithm for determining halting. we have reduced Lu to the halting problem.

If the halting problem is recursive then so is Lu. But we know that Lu is not recursive, as the assumption that halting problem was recursive must be false.