1. Construct a Turing machine that adds two unary numbers, as in Example 8–2, except that it erases the first 1 and changes the separating + to a 1.

Diagram

Description automatically generated

1. Draw a Turing machine that takes a string representing two unary numbers, x and y, separated by a 0, and determines whether x ≥ y. For example, the input for x = 3, y = 4 would be 11101111. Use two halt states: one for yes and one for no.

Shape

Description automatically generated with low confidence

1. Give the trace of your machine in the previous problem processing the strings 11101111 and 11110111.

(q0)11101111-\_(q1)1101111-\_1(q1)101111-\_11(q1)01111-\_110(q2)1111-\_1101(q2)111….\_1101111\_(q2)-\_1101111(q4)-\_110111(q3)\_-\_11011(q3)1\_...-(q3)\_110111\_-\_(q0)110111\_.....-\_\_110111\_(q2)\_...\_\_\_1011\_(q2)\_\_...\_\_\_\_01\_(q2)\_\_\_\_... halt on no

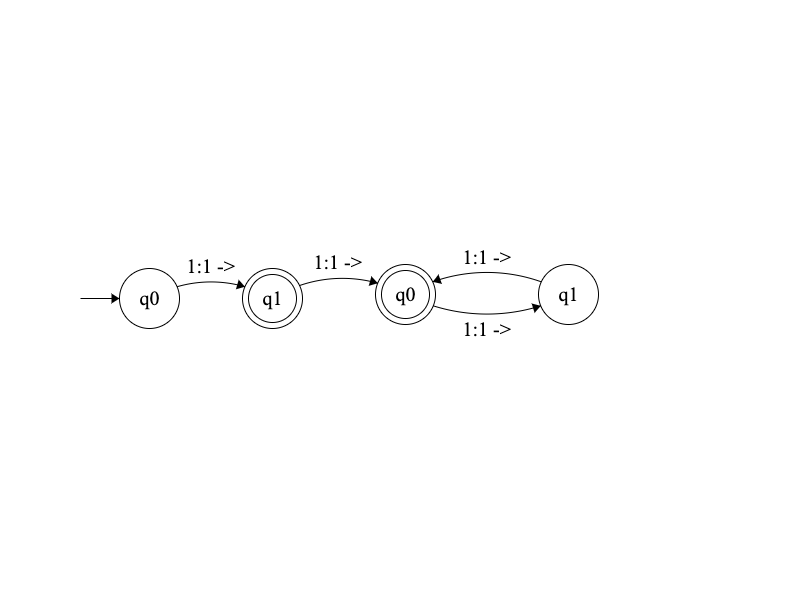
11110111…\_111011\_….\_\_1101\_\_...\_\_\_10\_\_\_... \_\_\_\_0\_\_\_... halt on yes

1. Draw a TM that computes f(w) = wR | w ∈ (a + b)∗. For simplicity, you can use a delimiter and just copy the letters backwards at the right, so, for example, abb# becomes abb#bba.

Shape

Description automatically generated with low confidence

1. Draw a TM that accepts unary numbers representing powers of 2. (Hint: Except for 1 and 2, dividing powers of 2 by 2 always gives an even number.)



1. Draw a Turing machine that computes ⌈n/2⌉ for unary numbers. Shape

   Description automatically generated with low confidence