**Turing Machines - a formal definition**

A Turing Machine is a 6-tuple

\[ M = (Q, \Sigma, I, q_0, \delta, F) \]

where

- **Q** finite, non-empty set of states
- **Σ** finite set of at least 2 symbols: the alphabet. \(^ \in \Sigma\)
- **I** non-empty subset of \(\Sigma\); \(^ \notin I\); input alphabet
- **q_0** \(q_0 \in Q\); starting or initial state
- **δ** \((Q \setminus F) \times \Sigma \Rightarrow Q \times \Sigma \times \{-1, 0, 1\}\), a partial function, the instruction table
- **F** \(F \subseteq Q\), the set of final or halting states
Some notation:

A word is a string of symbols eg. lecture^room

..for an alphabet (set of symbols) $\sum$

  a word of $\sum^*$ is a finite string of elements of $\sum$.

$\sum^*$ = all words of $\sum$

$\varepsilon$ is the empty word; $\varepsilon \in \sum^*$ for any $\sum$.

We represent concatenation of words $v$ and $w$ by $v.w$

The value computed by the TM, $M$ on input $w$ is $f_M(w)$. If $M$ does not Halt & Succeed on input $w$, then $f_M(w)$ is undefined.
The TM tape contents:

The input alphabet is $I$; $\not\in I$. TM Input is a word of $I^*$

| $w_0$ | $w_1$ | $w_2$ | $w_3$ | $w_4$ | $\not\in$ | $\not\in$ | $\not\in$ |

Eg. a tape with contents $w = w_0 w_1 w_2 w_3 w_4 \in I^*$ at start of TM run

There are no blanks ($\not\in$) in the input $w$..
the input word is terminated by the first $\not\in$ on the tape when
the TM starts.

If all the $w_i$ are blank, we have the empty input, $\varepsilon$. 
\( \Sigma \) is the whole or internal alphabet of the TM.

The input alphabet \( I \) is a subset of \( \Sigma \): \( I \subseteq \Sigma \) and \( ^\wedge \in \Sigma \).

The TM Output is a word \( w' \) of \( \Sigma^* \) without blanks (\(^\wedge\)).
It starts at square 0 and
ends just before the first blank, \(^\wedge\), on the tape
when the TM halts and succeeds.
(otherwise it is not defined).

Not all words of \( \Sigma^* \) are necessarily valid outputs.

Not all symbols of \( \Sigma^* \) are necessarily valid outputs.
a Turing Machine example..

\[ M = (\{q_0, q_1, q_2\}, \{1, \^\}, \{1\}, q_0, \delta, \{q_2\}) \]

\[ \delta \text{-function:} \]
\[ \delta (q_0, 1) = (q_1, \^, 1) \]
\[ \delta (q_0, \^) = (q_2, \^, 0) \]
\[ \delta (q_1, 1) = (q_0, 1, 1) \]

possible inputs: 11111\(^{\ldots}1\)
111\(^{\ldots}1\)
11\(^{\ldots}1\)
1\(^{\ldots}1\)

what does this TM do?
..a TM example

\[ \delta(q_0, 1) = (q_1, ^, 1) \]
\[ \delta(q_0, ^) = (q_2, ^, 0) \]
\[ \delta(q_1, 1) = (q_0, 1, 1) \]

the TM M starts in state \( q_0 \) with the head over square 0..
if M is in state \( q_0 \): if M reads a 1 it
  - writes a ^
  - moves right 1 square on the tape
  - goes into state \( q_1 \)
otherwise, M reads a ^: it
  - writes a ^
  - does not move along the tape
  - goes into state \( q_2 \)
if M is in state \( q_1 \): if M reads a 1 it
  - writes a 1
  - moves right 1 square along the tape
  - goes into state \( q_0 \)
otherwise M halts and fails: no applicable instruction
if M is in state \( q_2 \) it
  - halts and succeeds
another Turing Machine example..

design a TM, \( M = (Q, \Sigma, I, q_0, \delta, F) \) which evaluates the function \( \text{head} \):

- \( I = \{a, b\}, \Sigma = \{a, b, ^\} \)
- \( \text{head}(w) = s, \) where \( s \in I, w_1 \in I^* \) and \( w = s.w_1 \)
- \( \text{head}(\epsilon) \) is undefined

\[
\begin{align*}
\delta (q_0, a) &= (q_1, a, 1) \\
\delta (q_0, b) &= (q_1, b, 1) \\
\delta (q_1, a) &= (q_2, ^, 0) \\
\delta (q_1, b) &= (q_2, ^, 0) \\
\delta (q_1, ^) &= (q_2, ^, 0)
\end{align*}
\]

\[
\begin{array}{cccccccc}
\text{b} & \text{a} & \text{b} & ^ & ^ & ^ & ^ & \ldots \\
\hline
q_0 & \text{bab}^{^\ldots} \\
q_1 & \text{bab}^{^\ldots} \\
q_2 & \text{b}^{^b}^{^\ldots} \\
\end{array}
\]

Halt and Succeed
We have seen a formal definition of a Turing Machine:

\[ M = (Q, \sum, I, q_0, \delta, F) \]

where

- \( Q \) finite, non-empty set of states
- \( \sum \) finite set of at least 2 symbols: the alphabet. \( ^{\wedge} \in \sum \)
- \( I \) non-empty subset of \( \sum \); \( ^{\wedge} \notin I \); input alphabet
- \( q_0 \) starting or initial state \( q_0 \in Q \)
- \( \delta \) \( \delta: (Q\setminus F) \times \sum \Rightarrow Q \times \sum \times \{-1, 0, 1\} \), a partial function, the instruction table
- \( F \) \( F \subseteq Q \), the set of final or halting states

and examples of simple TMs to
- determine whether a number is odd/even
- return the Head of the input word.
Design a Turing Machine to implement the Tail function

\[ Q = \{ q_0, \ldots \}, \quad F = \{ \}, \quad \sum = \{ a, b, \text{\^}\}. \]

\[
\begin{array}{cccccc}
a & b & a & b & \text{\^} & \text{\^\text{\^\text{\^\text{\text{\^}}}}}
\end{array}
\]