A finite state machine($m$) can be determined as a 5 tuple (Q,Î£,Î´,q0,F) Where the symbols can be defined as, Q as set of states,$\Sigma$ as the input alphabet, Î´ being the transition function,q0 as the start state,F as the states of accepting states, a basic overview in how this works can be said as this reads the input string from the left to right in a single pass. When it encounters a symbol it moves to a new state based on the transition function. After finishing reading the string We are in accepting state, where we can either accept the string or reject it. The language that a finite state machine may accept are all R.L's(Regular Languages). The memory states are finite as well.

A deterministic Turing machine is 6 tuple (Q,Î£,Î´,q0,q accept,q reject) where the symbols can be defined as follows.Q being the set of states $\Sigma$ is the tape alphabet, which include the input symbol or symbols you can write on the tape, and also it includes the blank symbol.Î´ being the transition function,q0 as the start state,q accept is the accepting state and q reject is the rejecting state. A Turing machine works on the assumption that we have an infinitively long tape. when the machine starts it has the input reading in the portion of the tape followed by trailing blanks. It can be noted that the Turing machine is more powerful than the finite state machine because it can read or write a symbol and go back and forth as many as times it wants. Another big advantage it has is that it can decide on any computable languages hence making its total configurations arbitrary large.

Whereas a modern day computer doesn't have unbounded storage but Turing machines do. So in technical speaking modern day computers are closer to finite state machines but again the present states of an 8 GB ram machine will be around 2^3936*2^2^36 which is technically huge, but again Input Output (I/O) are not specified in Turing machines. Input is whatever is inserted in the tape and output is whatever is written on the tape when the machine halts or stops, on the contrary Modern Day machines accept I/O all the time during execution.

We were unable to transcribe this image
We were unable to transcribe this image
We were unable to transcribe this image