ABSTRACTION AND AUTOMATION

Logical reasoning: process of using a given set of facts to determine whether new facts are true

Problem solving: process of finding a solution to real life problems

Algorithm: a sequence of instructions

Abstraction

Concept of picking out common concepts in order to reduce the problem to its essential defining features. There are two main types of abstraction:

- Representational abstraction: process of removing unnecessary details so that only information that is required to solve the problems remains
- Abstraction by generalization/categorisation: concept of reducing problems by putting similar aspects into hierarchical categories
    - Procedural abstraction: breaking down solutions into a series of procedures or subroutines, is the basis for top-down design
    - Functional abstraction: similar to the procedural abstraction, focuses on common functions that can be used to solve problems. Using functions reduces complexity as the functions only needs to be written once.
    - Data abstraction: hiding how data is represented so that it is easier to build a new kind of data object
    - Problem abstraction: removing unnecessary details from a problem until the underlying problem to see if this problem has already been solved.

Information hiding: process of hiding all details of an object that do not contribute to its essential characteristics. E.g. a satnav interface hides the complexity of calculating the route

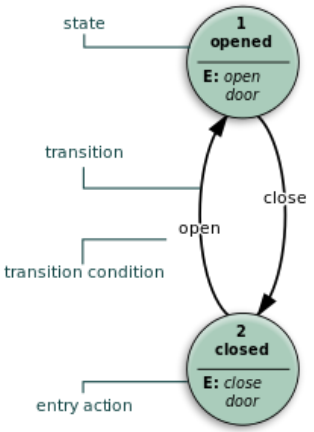
Decomposition: breaking down a large complex task into a series of manageable subtasks.

Composition: building up a whole system from smaller units. The opposite of decomposition.

Automation: process of creating a computer model of a real-life situation and putting it into action

FINITE STATE MACHINES AND THE TURING MACHINE

A finite state machine (FSM) is any device that stores its current value and whose status can change as the result of an input. There are finite number of transition that can take place.



- Mealy machine: type of finite state machine with outputs

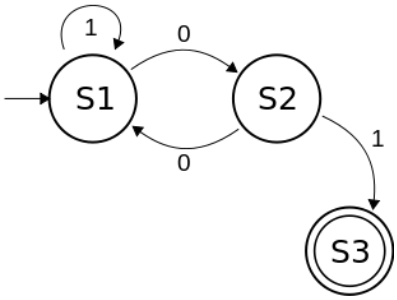
FSMs are the basis for programs for spell checking, indexing or searching text, recognising speech and network protocols.

There are two main ways to represent an FSM: a state transition diagram or a state transition table:

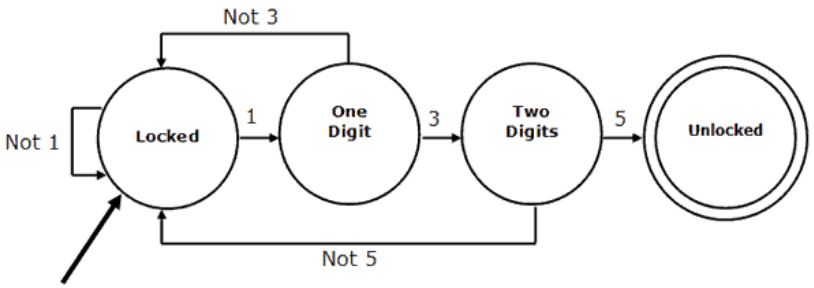
State transition diagrams

Are a visual representation of an FSM using circles to represent each state and arrows to represent the transition that occur as the result of an input.

- *Accepting state*: the state that identifies whether an input has been accepted, it is represented with a double circle.
- *Default state*: the starting state, it is indicated with an arrow



State transition diagram of a FSM to show the process of unlocking a computer with password 135.



State transition table

A tabular representation of a FSM showing inputs, current state and next state.

State transition table for the FSM represented in the diagram above.

Input	Current state	Next state
0	S1	S2
1	S1	S1
0	S2	S1
1	S2	S3

The Turing machine

The Turing machine is a theoretical model developed by Alan Turing in 1936 as a way of trying to solve what was called the decision problem. The problem was whether it was theoretically possible to solve any mathematical problem within a finite number of steps given particular inputs. Turing developed a theoretical machine that was able to carry out any algorithm and in doing so essentially produced a model of what is computable: a problem is computable iff it can be computed by a Turing Machine.

A Turing machine is a finite state machine with the ability to read and write data to an unlimited tape. Turing machine can be viewed as a computer with a single fixed program, expressed using:

- a finite set of states in a state transition diagram
- a finite alphabet of symbols
- an infinite tape with marked-off squares
- a sensing read-write head that can travel along the tape, one square at a time.

A program in a Turing machine must have: a starting state, a halting state, an alphabet and a transition function (method of notating how the machine moves from one state to another and how data on the tape changes). This can be represented using a state transition diagram very similar to the ones used for FSM (few extra symbols shown in the table), an instruction table or a list of transition rules in the form:
$\delta(Current\ State,\ Input\ Symbol) = (Next\ State,\ Output\ Symbol,\ Movement)$.

Symbol	*Meaning*
|	Separates input from output
□	Blank square
#	Delimiting symbol
→	Move the observation window 1 square to the right
←	Move the observation window 1 square to the left

Universal Turing machine

A machine that can simulate any Turing machine by reading the description the individual Turing machine and the inputs required.

To a UTM the program and the data are the same thing → earliest form of the stored program concept.