AI

-Search:  
We must search a problem to solve (how to play a game)

-Knowledge:  
The AI must know the information (what to do)

-Uncertainty:  
What if a PC is not sure about a fact (I’m not sure the opponent is behind the tree)

-Optimization:  
Finding the best way (there are multiple way to solve a problem)

-Machine learning:  
Learning from experience (is that a spam email or a normal one)

-Neural Networks:  
How can a PC take inspiration by human brain?

-Language:  
Not programming language, but human language and how a PC can understand it

Search:  
Searching the problem, for example solving the 15 puzzles, figure out a way to escape from a maze. Last one example is something like google maps which return the best way to take from A to B.

Some terminology:  
-Agent: entity that acts depending the environment

-State: configuration of the agent (how the problem to solve is. In the fifteen puzzle we must find different moves depending of what is the starting configuration)

-Initial State: starting point to develop our algorithm

-Actions: Choices made from the AI so -> ACTIONS(s) = action take as input the state. This function returns the set of actions that can be executed in the given state. (In the 15 puzzle, most of the time, the output will be 4 moves: ←↑→↓  
We need a connection between actions and states, so here it comes

-Transition Model: how would be the given state after performing all the possible actions. The function is RESULT(s, a) that returns the state after performing actions(a) in the state(s). So if we have the fifteen puzzle game with the missing square on the bottom left corner we can run this function: RESULT(state where the table has got the empty square I the bottom left corner, moving on the right a square action)  
and we get as output the table with the squares moved.

-State Space all the possible moves available from the initial state performing any sequence of actions.  
This is represented by the classical images with dots and lines where all the dots are individual states reachable by other states connected to that state following different actions.

-Goal Test: giving a feedback to AI about its solution

-Path Cost: If we ask to google maps to brings us from A to B it should give just casual solution which takes us to the destination. This solution may be a long road. So with this thing we can give to a solution a path cost so when we ask to google to brings us to B we will also ask for the solution with the lowest path cost.

Initial state: the situation to solve the problem  
Actions: possibly actions to reach the solutions  
Transition model: the result of applying certain moves on certain states  
Goal Test: giving a feedback to the AI  
Path Cost: defining which solution is the fastest/convenient

After these steps we can finally find a

-solution which is the completed table of our fifteen puzzle game

-optimal solution is the solution with the lowest path cost

With all these datas we also need a

-Node: a data structure that keeps track of all datas: the state we are working now, the previous node that created this one (a parent), an action that created this node, the path cost (from initial state to this node).  
The parent data is important because allows us to see the previous action that the PC used to get to the result