Computational Methods

To be able to solve a problem using Computational Methods:

- Define the problem clearly: Identify the end goal and the possible obstacles
- What calculations are needed? Can they be done quickly with little processing?
- What data types are needed? How much storage is needed?
- Can the problem be approached using decomposition and abstraction?

Problem Recognition:

- Describe the problem thoroughly
- Describe the solution you want
- Reduce the problem to a more general case
- Makes it easier to find the optimal solution

Abstraction: The process of removing all unnecessary details from a problem, leaving only the aspects crucial to solving the problem; Simplifies the complexity of a problem, making it easier to write an algorithm to solve it
Decomposition: The process of breaking a problem down into smaller, more manageable sub-problems that are easier to write algorithms for; Simplifies the complexity of a problem, making it easier to write an algorithm to solve it
Divide and Conquer: recursively breaks a problem down into sub-problems until they become simple to solve. The solutions of the sub-problems are combined to form the solution to the original problem; makes large problems much easier to solve
Enumeration: The process of designing an algorithm that performs an exhaustive search and attempts all possible solutions until the correct one is found; Computers will be able to complete a complex task in a smaller amount of time
Automation: The process of building problem-solving models and putting them into action, Simplifies the complexity of a problem, making it easier to write an algorithm to solve it
Simulation: The process of designing a model of a real system in an attempt to understand its behaviour, Simplifies the complexity of a problem, making it easier to write an algorithm to solve it
Theoretical Approach: The process of breaking a problem down to pure theory and representing it using maths equations; computers are great at maths.
Computational Methods: Methods to find a way to move from a current situation to a desired outcome.

Backtracking: Incrementally building towards a solution; when you make a mistake, return to a previous successful match to find an alternative path.
Data Mining: Analysing vast amounts of data gathered from a variety of sources to discover new information and trends
Heuristics: Encourages us to make use of our experience and find a solution that can be considered "good enough", as long as it is found faster.
Performance Modelling: Carrying out mathematical approximations to test how efficient models are in different situations (stress testing)
Pipelining: When a task is divided into a series of subtasks, where each subtask leads to the next, the result of a process feeds into the next one, and this is repeated until the whole task is completed.
Visualisation: Turning data into a visual representation which is easier for humans to understand and work with