FIT1045 Algorithmic Problem Solving – Tutorial 9.

Objectives

The **objectives of this tutorial** are:

- To become familiar with backtracking.
- To understand how to apply backtracking to specific problems.

Task 1

Part A

Consider the backtracking approach to generating all bitstrings of length N. As a class, step through the case for $N = 2$.

What is a partial solution in this case?
What is a complete solution?
What is the list of next possible items in this case?

Part B

Now use the backtracking approach to find all bitstrings of length N with exactly 2 zero bits. Step through the case for $N = 3$.

What is a partial solution in this case?
What is a complete solution?
Is the list of next possible items always the same at each step?

Task 2

The Knapsack Problem In groups of 3–4, use backtracking to find the subset of items with the most value that can be packed into a Knapsack that can hold a maximum weight of 10kg.

	Item 1	Item 2	Item 3	Item 4	Item 5
Weight (kg)	5	4	6	1	7
Value ($)	3	2	3	2	7

Task 3

Create an algorithm that uses backtracking to solve the n-Queens problem. Each person in your group should be responsible for a different function (or functions) of the algorithm. Step through the algorithm for the $n = 4$ case.

Task 4

Write an algorithm that uses backtracking to generate all permutations of the numbers 1 to n.

How would you modify this algorithm to solve the Travelling Salesperson problem?

A *Hamiltonian Cycle* is a cycle that visits every vertex in the graph exactly once. How would you modify your algorithm to find a Hamiltonian cycle in a graph?

Puzzle of the week

Arithmetic Progression An arithmetic progression (AP) or arithmetic sequence is a sequence of numbers such that the difference between the consecutive terms is constant. For instance, the sequence 5, 7, 9, 11, 13, 15 is an arithmetic progression with common difference of 2.
Given a list that represents elements of arithmetic progression in order. One element is missing in the progression, find the missing number.

Examples:

Input: $arr[] = \{2, 4, 8, 10, 12, 14\}$
Output: 6

Input: $arr[] = \{1, 6, 11, 16, 21, 31\}$
Output: 26