FIT1008 – Intro to Computer Science
Tutorial 10

The **objectives of this tutorial** are:

- To understand Dynamic Programming.

Task 1

Write down an algorithm to compute the n-th Fibonacci number using Dynamic Programming. How does this compare in terms of time complexity to the recursive and tail recursive implementations done in the class?

Task 2

Pick up the largest amount of money from the coins below, with the constraint that you cannot pick up any two adjacent coins.

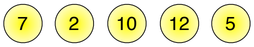

Figure 1: A row of coins.

How can you design an algorithm to solve the problem for n coins of arbitrary values?

Task 3

Type	1	2	3	4	5
Value	$4	$5	$10	$11	$13
Weight	3kg	4kg	7kg	8kg	9kg

Using dynamic programming, solve the knapsack problem given by the table above. The knapsack capacity is 17 kg.

Task 4

In solving the knapsack, assume that you are now allowed to take as many items as you can of each type as long as the knapsack capacity is not exceeded. Discuss how you would modify the Dynamic Programming algorithm given in the lecture to solve the problem when duplicate types are allowed.

Task 5*

- Write a **recursive** algorithm to compute the binomial coefficients.[1]

[1] The binomial coefficients are given by the following formula

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Assume $n \geq k$, with $n > 0$, $k > 0$, $n, k \in \mathbb{Z}$. Note that $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$

- Write a **Dynamic Programming** algorithm to compute the binomial coefficients.

Compare the complexity of the two implementations.