# UE19CS251
# Design and Analysis of Algorithms
# Unit 5: Limitations of Algorithmic Power and Coping with the Limitations

The Knapsack Problem

PES University

## Outline

**Concepts covered**

- The Knapsack Problem
  - Introduction
  - Recurrence
  - Example

## 1 Problem Definition

- Given
  - $n$ items of integer weights : $w_1$ $w_2$ ... $w_n$; values : $v_1$ $v_2$ ... $v_n$
  - knapsack of capacity $W$ (integer $W > 0$)
- Find the most valuable subset of items such that sum of their weights does not exceed $W$

## 2 Knapsack Recurrence

- To design a dynamic programming algorithm, we need to derive a recurrence relation that expresses a solution to an instance of the knapsack problem in terms of solutions to its smaller subinstances

- Consider the smaller knapsack problem where number of items is $i$ ($i \leq n$) and the knapsack capacity is $j$ ($j \leq W$)
- <2-> Then

$$F(i,j) = \begin{cases} \max(F(i-1,j), \quad v_i + F(i-1, j-w_i)) & \text{if } j - w_i \geq 0 \\ F(i-1,j) & \text{if } j - w_i < 0 \end{cases}$$

# 3 Example

$$F(i,j) = \begin{cases} \max(F(i-1,j), \quad v_i + F(i-1, j-w_i)) & \text{if } j - w_i \geq 0 \\ F(i-1,j) & \text{if } j - w_i < 0 \end{cases}$$

**Dynamic Programming Example**

| item $i$ | weight $w_i$ | value $v_i$ |
|---|---|---|
| 1 | 2 | 12 |
| 2 | 1 | 10 |
| 3 | 3 | 20 |
| 4 | 2 | 15 |

What is the maximum value that can be stored in a knapsack of capacity 5?

| | capacity $j$ | | | | |
|---|---|---|---|---|---|
| $i$ | 1 | 2 | 3 | 4 | 5 |
| 1 | 0 | 12 | 12 | 12 | 12 |
| 2 | 10 | 12 | 22 | 22 | 22 |
| 3 | 10 | 12 | 22 | 30 | 32 |
| 4 | 10 | 15 | 25 | 30 | 37 |

Given above 6 items, maximum value that can be stored in a knapsack of capacity 5 is **37**

# 4 Complexity

- Space complexity: $\Theta(nW)$
- Time complexity: $\Theta(nW)$
- Time to compose optimal solution: $O(n)$

# 5 Think About It

- <2-> Write pseudocode of the bottom-up dynamic programming algorithm for the knapsack problem
- <3-> True or False:
  1. <3-> A sequence of values in a row of the dynamic programming table for the knapsack problem is always nondecreasing?
  2. <4-> A sequence of values in a column of the dynamic programming table for the knapsack problem is always nondecreasing?