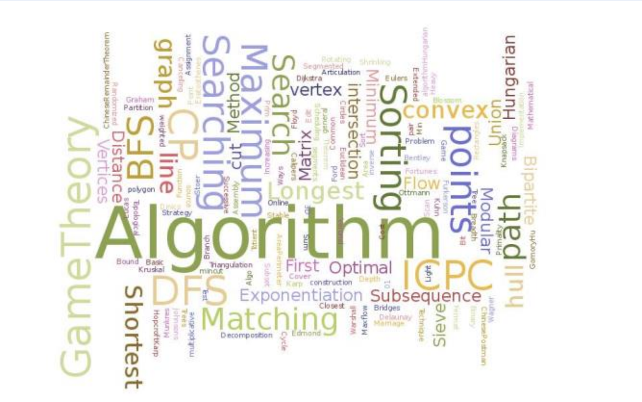
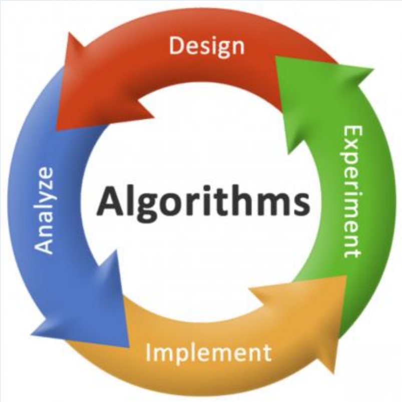
Dynamic Programming
0-1 Knapsack

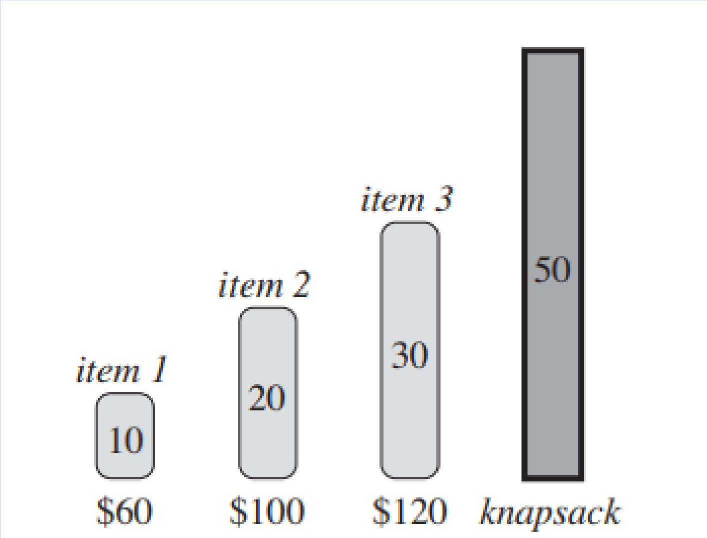
COP 3503
Fall 2021
Department of Computer Science
University of Central Florida
Dr. Steinberg





The Knapsack Problem

A thief robbing a store finds n items. The i^{th} item is with v_i dollars and weighs w_i pounds, where v_i and w_i are integers. The thief wants to take as valuable a load as possible, but he can carry at most W pounds in his knapsack, for some integer W. Which items should the thief take?



Source: Cormen Intro to Algorithms 3rd Edition

0-1 Problem

- In this problem we can only take the whole object or none of it.
- Input
  - n objects
  - i = 1, …, n object that has a weight w_i and value v_i (always a positive value)
  - A knapsack that can only carry a limit weight ($weight \leq W$)
- Objective
  - To fill the knapsack such that to maximize the value of the included objects, while keeping the knapsack to weight capacity limits.

Remember the Greedy Approach?

```
FractionalKnapsack(W, n) //weight and max weight W of the knapsack, n is size
Apply sorting the array I based on v_i/w_i
for i = 1 to n
    x_i = 0
load = 0
value = 0
i = 1
while load < W and i <= n
    if w_i <= W – load
        x_i = 1
    else
        x_i = (W – load)/ w_i of item i
    load = load + x_i*w_i
    value = value + x_i*v_i
    i = i + 1
```

How can Dynamic Programming be utilized?

- We can create a 2D array V[1…n,0…W]
  - Row for each object that is available
  - Column for each weight
- V[i,j] can represent the maximum value of objects we can take if the weight limit between 0 and W
- The solution to the 0-1 knapsack problem can be given by V[n,W].

Setting up V

- $V[i,0] = 0$ for all i values
- Fill table row by row or column by column
- Computing the value in $V[i,j]$. There are two options
  - Do not add object i to the knapsack, $V[i,j] = V[i-1, j]$
  - Add object i to the knapsack, $V[i,j] = v_i + V[i-1, j-w_i]$
  - Overall $V[i,j] = \max(V[i-1, j], v_i + V[i-1, j-w_i])$
- Out of bounds table entires
  - $V[0,j] = 0$ when $j >= 0$
  - $V[i,j] = -\infty$ for all i when $j < 0$

Let's Derive the Dynamic Programming Solution