Safe packing **PD 2019/20** Declarative Programming

You need to pack several items into your shopping bag without squashing anything. The items are to be placed one on top of the other. Each item has a weight and a strength, defined as the maximum weight that can be placed above that item without it being squashed. A packing order is safe if no item in the bag is squashed, that is, if, for each item, that item's strength is at least the combined weight of what's placed above that item. For example, here are three items and a packing order:

Position	**Item**	**Weight**	**Strength**
Top (0)	Apples	5	6
Middle (1)	Bread	4	4
Bottom (2)	Carrots	10	10

(A) This packing is **not safe**. The bread is squashed because the weight above it, 5, is greater than its strength, 4. Swapping the apples and the bread, however, gives a safe packing.

(A.1) Construct a CSP model for this problem, i.e. one which finds safe packings. In constructing the model, consider the need to place **N** items, where item **i** is placed in position $\mathbf{P_i}$ (0 means "at the top"), has weight $\mathbf{W_i}$ and strength $\mathbf{S_i}$. *[formal problem, not coding!!]*

You may (and should!) use indexed variables, which you will denote with a subscript, for example $\mathbf{X_i}$. Note that i may also be a constraint variable.

(A.2) Prepare a **Choco model** or a **CLP(FD)** program. Note that here, indexed variables will likely be an array of constraint variables, e.g. $\mathbf{X_i}$ will be seen as **varX[i]** (in Java) or X will be a list (in Prolog).

To test your model and program, use the following cases:

1. N=3, WS = { (5,6), (4,4), (10,10) } (example above)
2. N=5, WS = { (1,1), (2,1), (7,3), (4,4), (5,8) }

(B) Consider a variant of this problem, where we are given a **set of possible items** and the **maximum weight capacity, M,** of the backpack and wish to fill it up as much as possible (i.e. not exceeding its maximum weight) but don't know beforehand how many items (value of **N**) to include.

(B.1) Provide a **CSP formulation** and apply to the following example:

- M=12, N≤8, WS = { (2,20), (3, 9), (3,6), (1,8), (4,14), (5,4), (1,2), (1,7) }

(B.2) Present a **program** to solve this.

(C) Consider another variant of problem (A) where we wish to pack the most robust sets of items: the robustness is understood to be the sum of the margins for each item, i.e. the difference between the weight on top of it and its strength. We wish to find **maximum robustness**. Just indicate its CSP formulation.

Problem adapted from the Oxford University admission examination, as reported in The Guardian, 2019.11.04 edition.