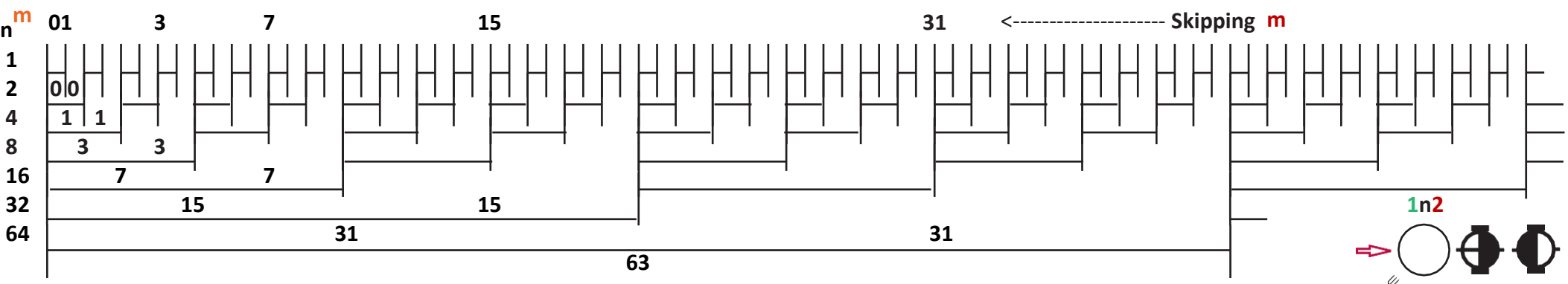
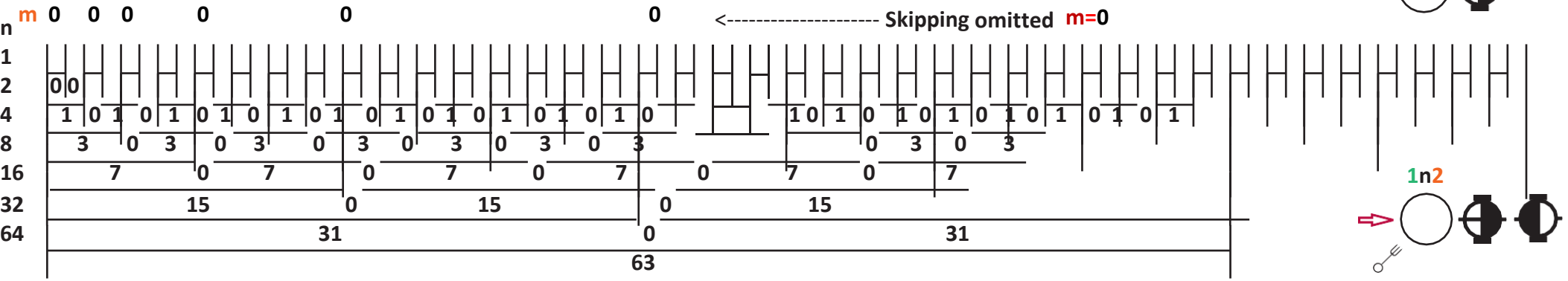


Example 1. The visualization, as an example “SumInPairsPathAlongMatrixTemplet” (Example 1-6), of the theory.



Example 2. The visualization, as an example “SumInPairsPathAlongMatrixTemplet”(Example 1-6), of the theory. The fragment to “Skip” omitted.

- 1-21

Who Where How	is going to address (precondition, postcondition-precondition,	“No elements to add up”. “Only one (no pair) element to add up - at the beginning”. “Only one (no pair) element to add up - the end of the iteration”. “Not enough elements to skip”. “What the precondition prevents divide by zero?”. “What the precondition prevents to have a negative index?”. “What the precondition prevents to have more indexes than elements.”.	What we do?

- 22

What the iterations are?

Example 3. The questions (to construct 22 of them)

Let's ask some questions and show **the difference** among the **Programming and Coding**, and then, illustrate what does **the notions** "Calculus for programming", "Calculus χπ" and Calculus "Core Pictograms" **mean**. The questions (to construct 22 of them) are:

- 1-21 (Example 3):

		"No elements to add up".	
		"Only one (no pair) element to add up - at the beginning".	
Who		"Only one (no pair) element to add up - the end of the iteration".	
Where	Is going to address (precondition, postcondition-precondition, postcondition)	"Not enough elements to skip".	What we do?
How		"What the precondition prevents divide by zero?".	
		"What the precondition prevents to have a negative index?".	
		"What the precondition prevents to have more indexes than elements.".	

- 22 (Example 3):

What the iterations are?

Programming or Coding? It depends on how we address the questions above. Then we consider:

- **The programming.** If we can address them as an object explicitly in the blueprint of the specification;
- **The coding.** If we present them as a derived feature from the code and there is nothing to point at.

If we consider an "Example 1", as **the theory**, then **"Calculus for programming"** consist of "Calculus χπ " and Calculus "Core Pictograms", where:

- **Calculus χπ** will be intentions represented graphically with a horizontal line,connecting two elements in accordance:
  - To calculate the sum of the pair and rewrite the result to the element on the left;
  - To select an element on the Right by skipping some number of elements:
    - To select the first element of the pair or the second pair.
- **Calculus "Core Pictograms"** will be calculations:
  - X:=X+Y, where (X,Y) – a selected pair, and X – at the Left, and Y – at the Right;
  - Recalculation of the indexes of the second element in the first pair and the index of the first element for the next pair.
  - Recalculation of the numbers, representing the index of the second element in the first pair and the skipped numbers of the elements, selecting next pair.