HaNoi University of Science and Technology

School of Information & Communication Technology

Software Requirement Specification – SRS

Evolutionary Algorithm Visualization

Subject: Software Engineering

Group No. 5

Lương Minh Triết

Phạm Vũ Huy

Nguyễn Xuân Phong

*Hà Nội, ngày 10 tháng 5 năm* *2021*

Table of Content

Table of Content ii

1 Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Glossary 1

1.4 Reference Document 4

2 General Discription 5

2.1 Actor 5

2.2 General Use Case Diagram 5

2.3 Professional Process 5

2.2.1 Software Usage Process 5

3 Functional Requirement 7

3.1 Use Case UC001 “Input Population Size” 7

3.2 Use Case UC002 “Visualization” 8

1. **Introduction**
   1. ***Purpose***

This document demonstrates evolutionary algorithm which contains Genetic Algorithm, Particle Swarm Optimization and Hill Climbing algorithm. All the algorithms are visually demonstrated using JavaFX. This document describes the purposes and the functionalities of the application, the interface and how the application reacts to interactions.

* 1. ***Scope***

Evolutionary algorithms are a heuristic-based approach to solving problems that cannot be easily solved in polynomial time and anything else that would take far too long to exhaustively process. When used on their own, they are typically applied to combinatorial problems.

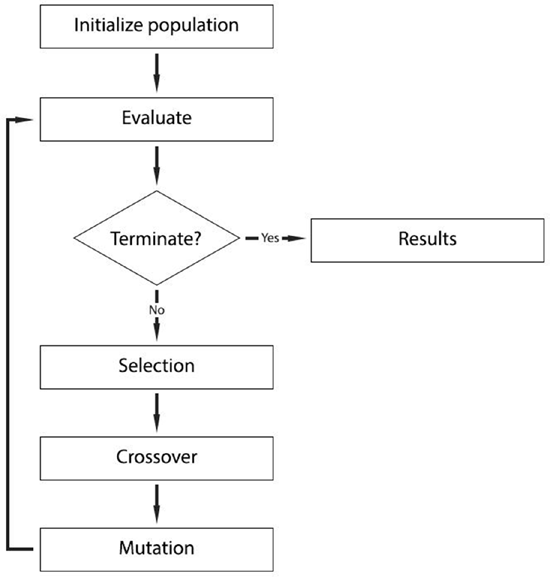
An EA contains four overall steps: **initialization, selection, genetic operators, and termination**. These steps each correspond, roughly, to a particular facet of natural selection, and provide easy ways to modularize implementations of this algorithm category.

In this application, the users are able to provide and create an initial population of vectors. Then, they will be able to choose one of the three algorithms to solve the problem and the chosen approach will be visually demonstrated to the users.

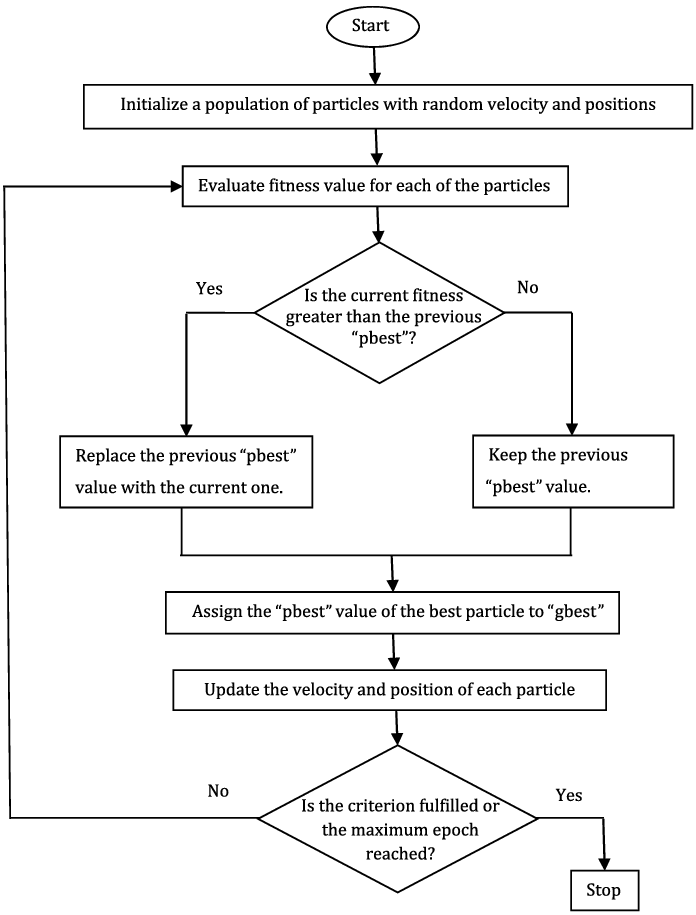
The program will only solve the Knapsack problem with 5 items and bag’s max weight of 10.

* 1. ***Glossary***
* The *evolutionary algorithm* is the main object of interest in evolutionary computation. There is a problem to be solved, and the solution is conceived to lie somewhere in a space of possible candidate solutions – the search space.

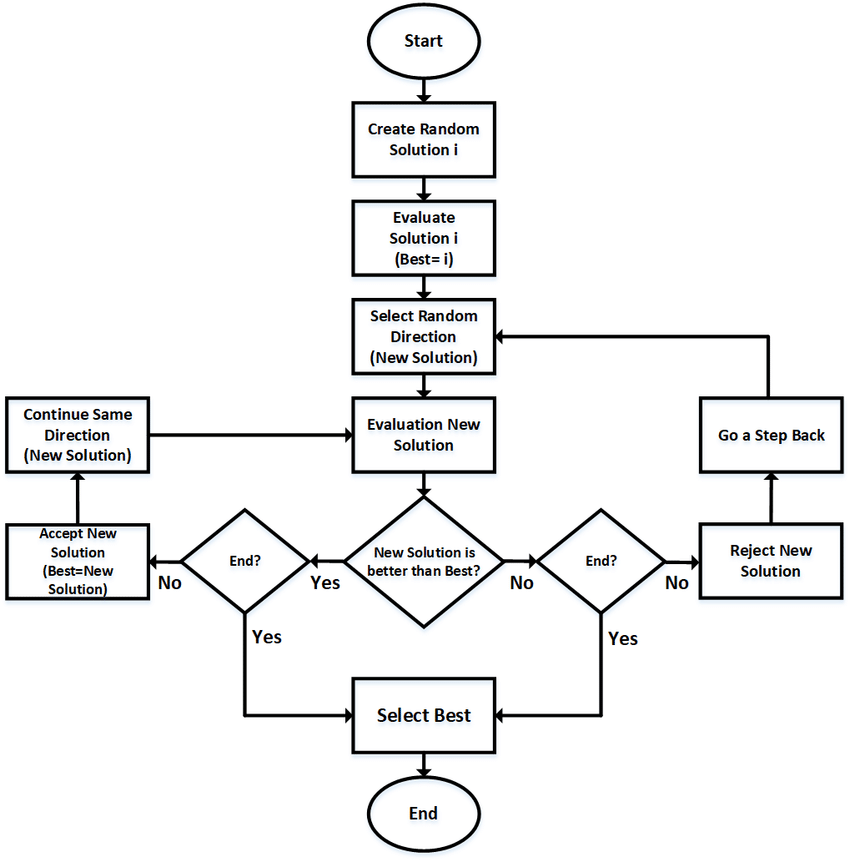
- **Genetic algorithms: Genetic algorithms simulate the process of natural selection to generate high-quality solutions for optimization problems and search problems.**



* Particle swarm optimization: The general idea of PSO is inspired by a flying swarm of birds searching for food. A PSO is used to track the local mode of the similarity measure and to seek a good local minimum, and then the conjugate gradient is utilized to find the local minimum accurately.



* Hill climbing: Hill-climbing solves the problems where we need to maximize or minimize a given real function by choosing values from the given inputs. This solution may not be the global optimal maximum.



* 1. ***Reference documents***

- <https://www.geeksforgeeks.org/genetic-algorithms/>

- <https://www.sciencedirect.com/topics/engineering/particle-swarm-optimization>

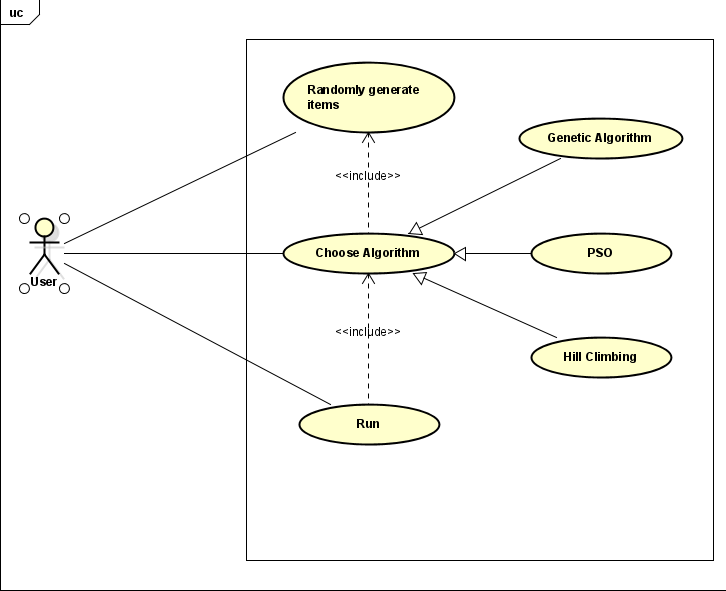
- <https://www.geeksforgeeks.org/introduction-hill-climbing-artificial-intelligence/>

1. **General description**

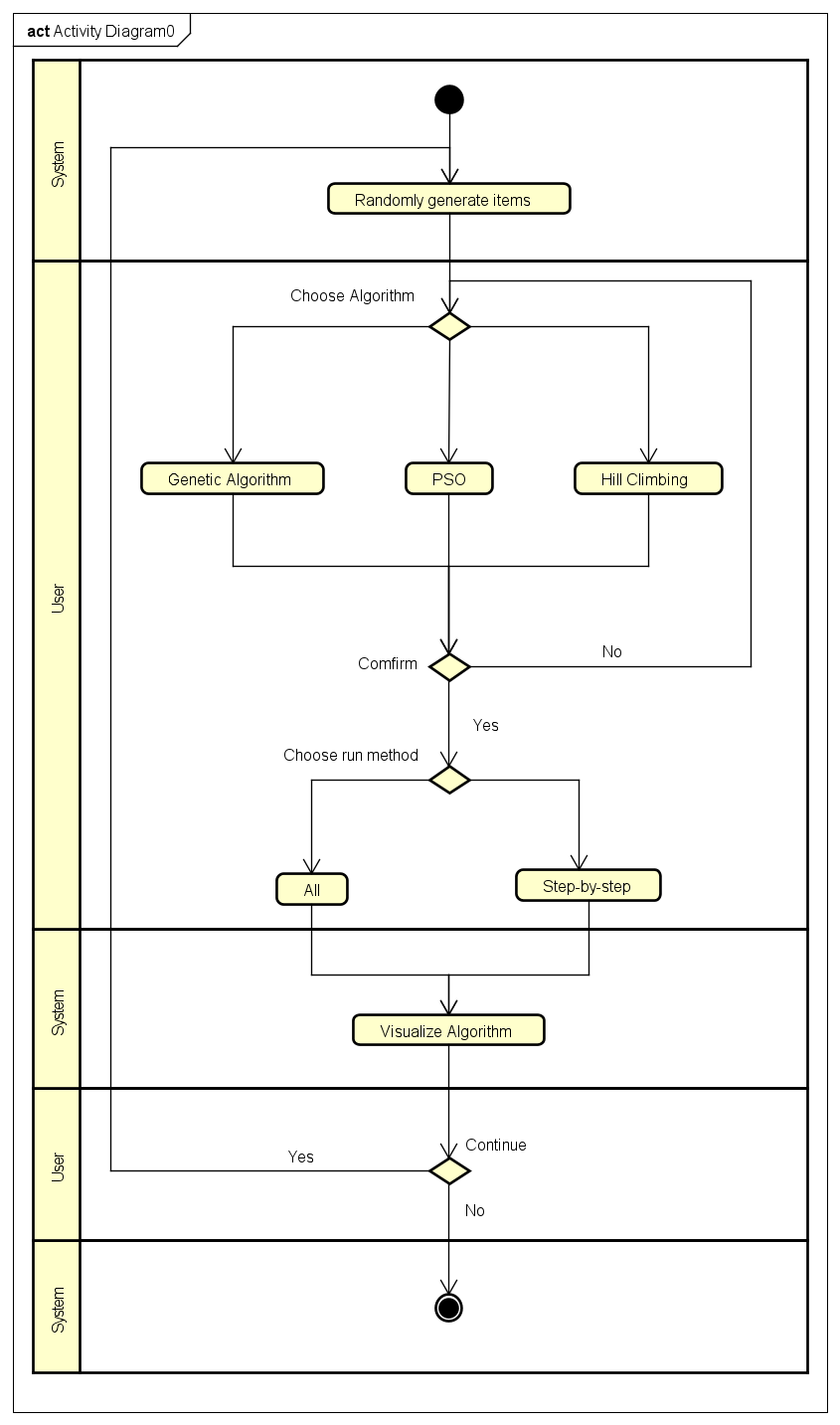
* **The system will visually demonstrate the algorithms and show the best value with the population randomly generated by user**
  1. ***Actors***

The application includes 1 actor: Users

* 1. ***General Use Case Diagram***



* 1. ***Professional Process***
     1. ***Software Usage Process***



1. **Functional Requirements**
   1. ***Use Case UC001 “Randomly generate items”***

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Code** | UC001 | **Use Case Name** | Randomly generate items |
| **Trigger** | User | | |
| **Precondition** | None | | |
| **Basic Path** | |  |  |  | | --- | --- | --- | | **No.** | **Performed by** | **Action** | | 1. | System | Display the interface for user | | 2. | User | Generate the population | | 3. | System | The elements of the vector will be created randomly | | 4. | System | Call Use Case “Choose Algorithm” | | | |
| **Postcondition** | None | | |

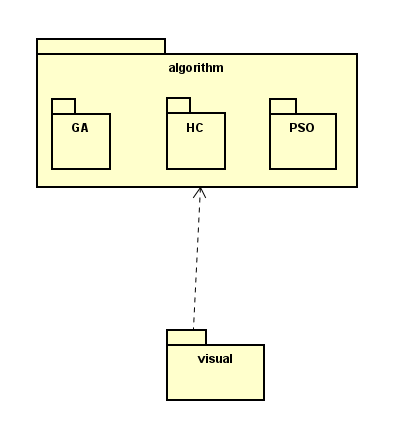
* 1. ***Use Case UC002 “Choose Algorithm”***

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Code** | UC002 | **Use Case Name** | Choose Algorithm |
| **Trigger** | User | | |
| **Precondition** | Items generated | | |
| **Basic Path** | |  |  |  | | --- | --- | --- | | **No.** | **Performed by** | **Action** | | 4. | User | Choose one of the three algorithm | | 6. | System | Call Use Case “Run” | | | |
| **Postcondition** | None | | |

* 1. ***Use Case UC003 “Run”***

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Code** | UC003 | **Use Case name** | Run |
| **Trigger** | System | | |
| **Precondition** | Algorithm chosen | | |
| **Basic Path** | |  |  |  | | --- | --- | --- | | **No.** | **Performed by** | **Action** | | 1. | User | Choose run all or step-by-step | | 2. | System | Run to the algorithm last step or step by step | | 3. | User | Choose another algorithms | | | |
| **Alternative Path** | |  |  |  | | --- | --- | --- | | **No.** | **Performed by** | **Action** | | 2a. | System | If the virtualization is not completed then modify the parameters | | 3a. | System | If user choose to try other algorithms then call Use Case “Choose Algorithm” again | | | |
| **Postcondition** | None | | |

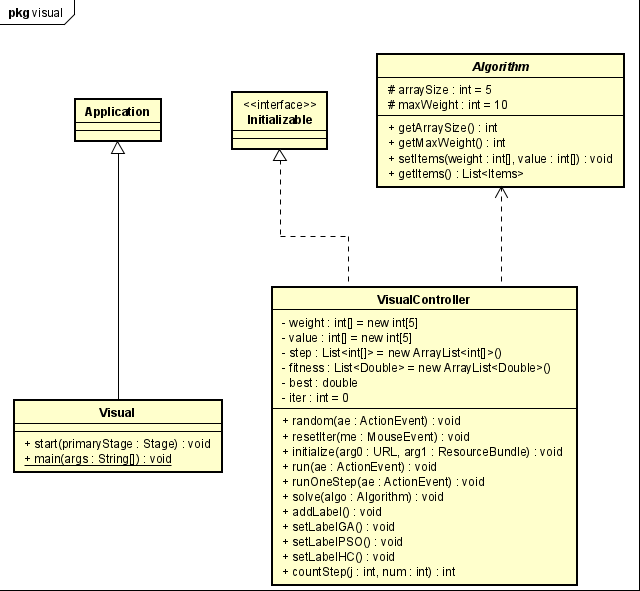
1. **Package Diagram**

****

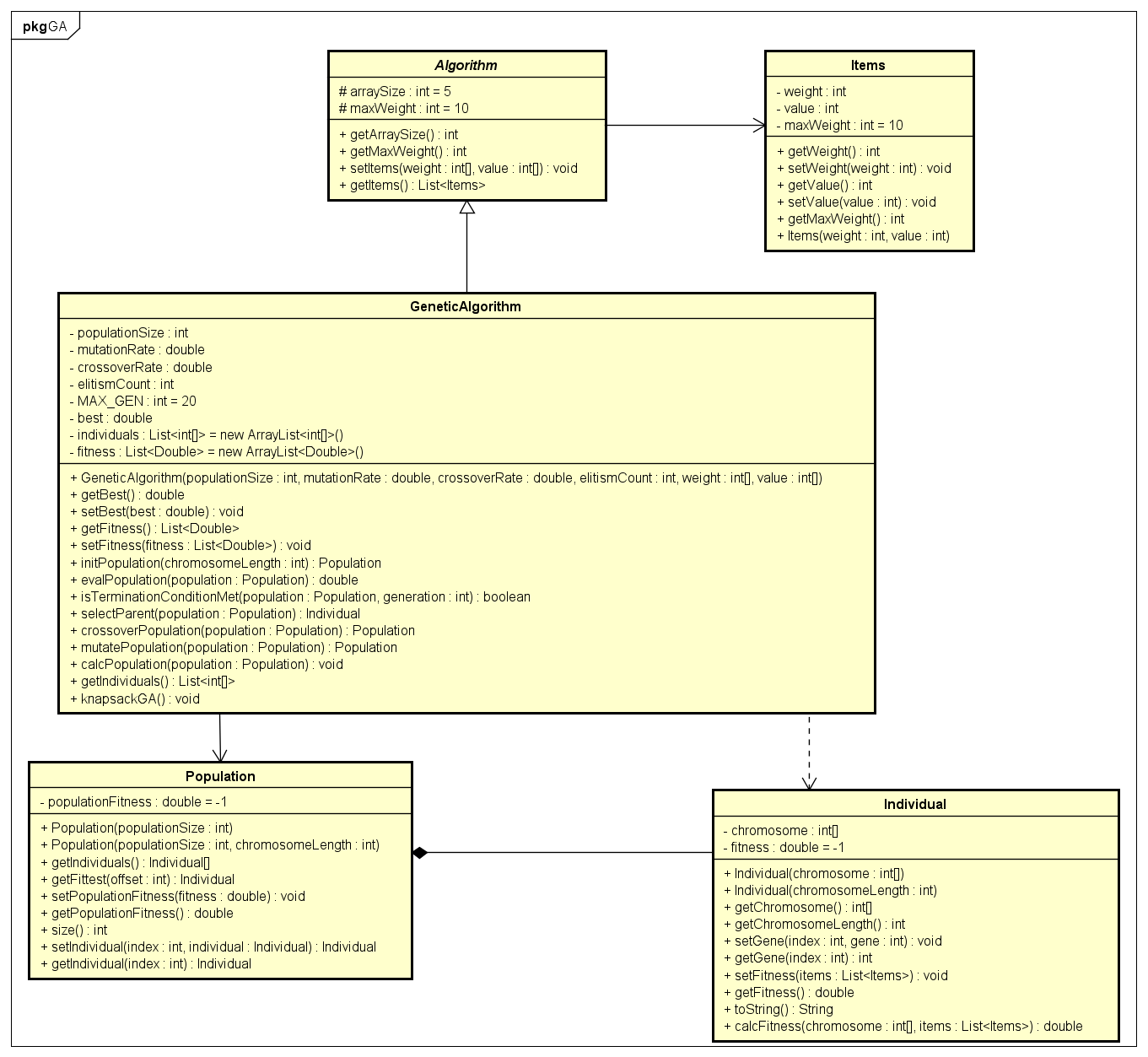
1. **Class Diagram**

**NOTE:** The algorithm, items, individual class is in algorithm package but I add it to the class diagram of other package to show the relationship of them all.

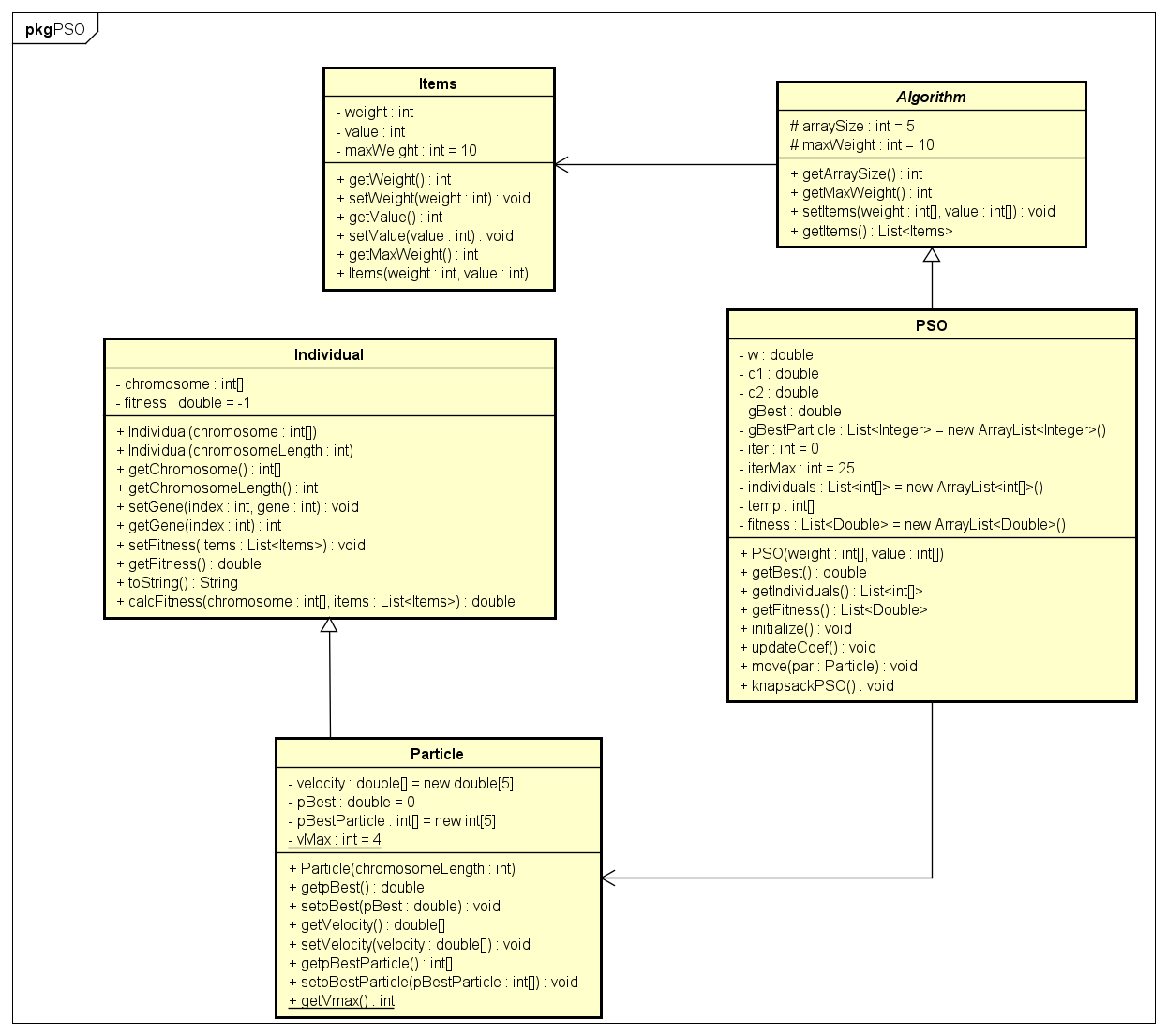
* 1. **Visual package**

****

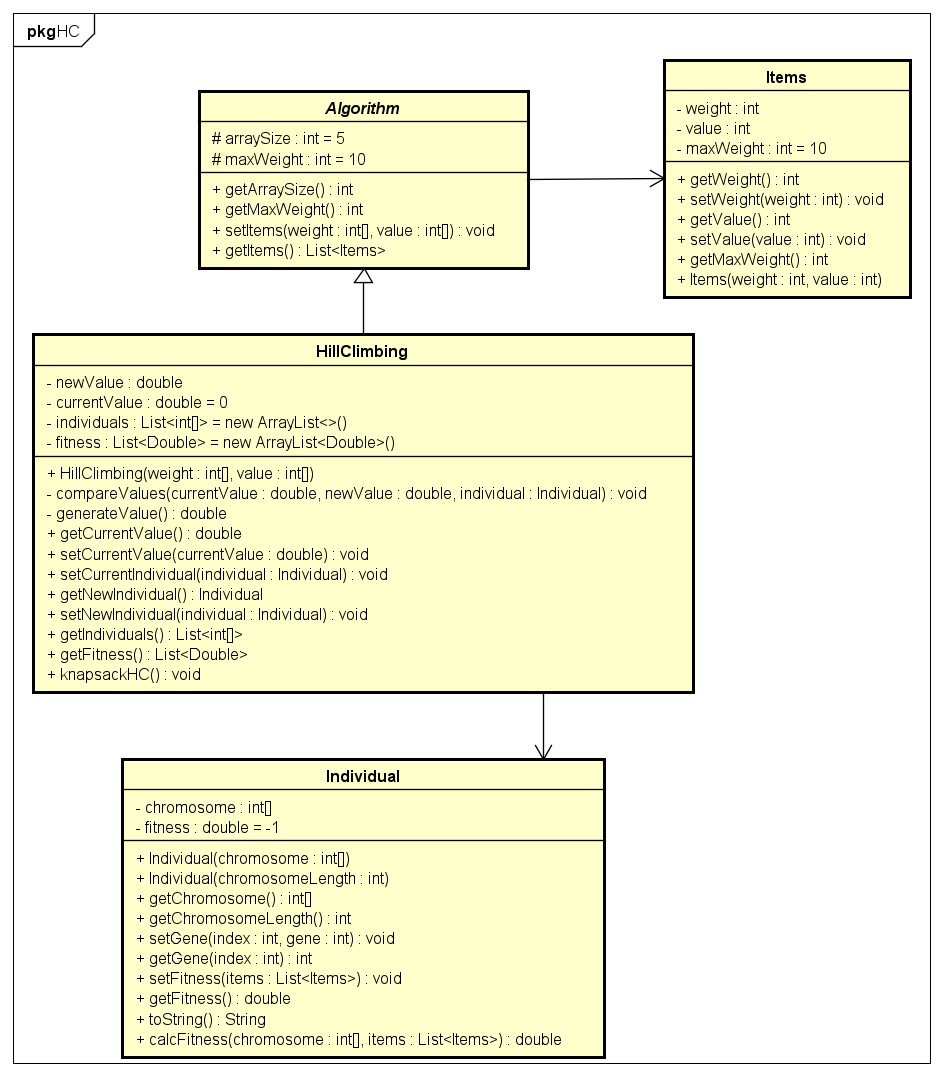
* 1. **Algorithm package** 
     1. **Genetic Algorithm package**



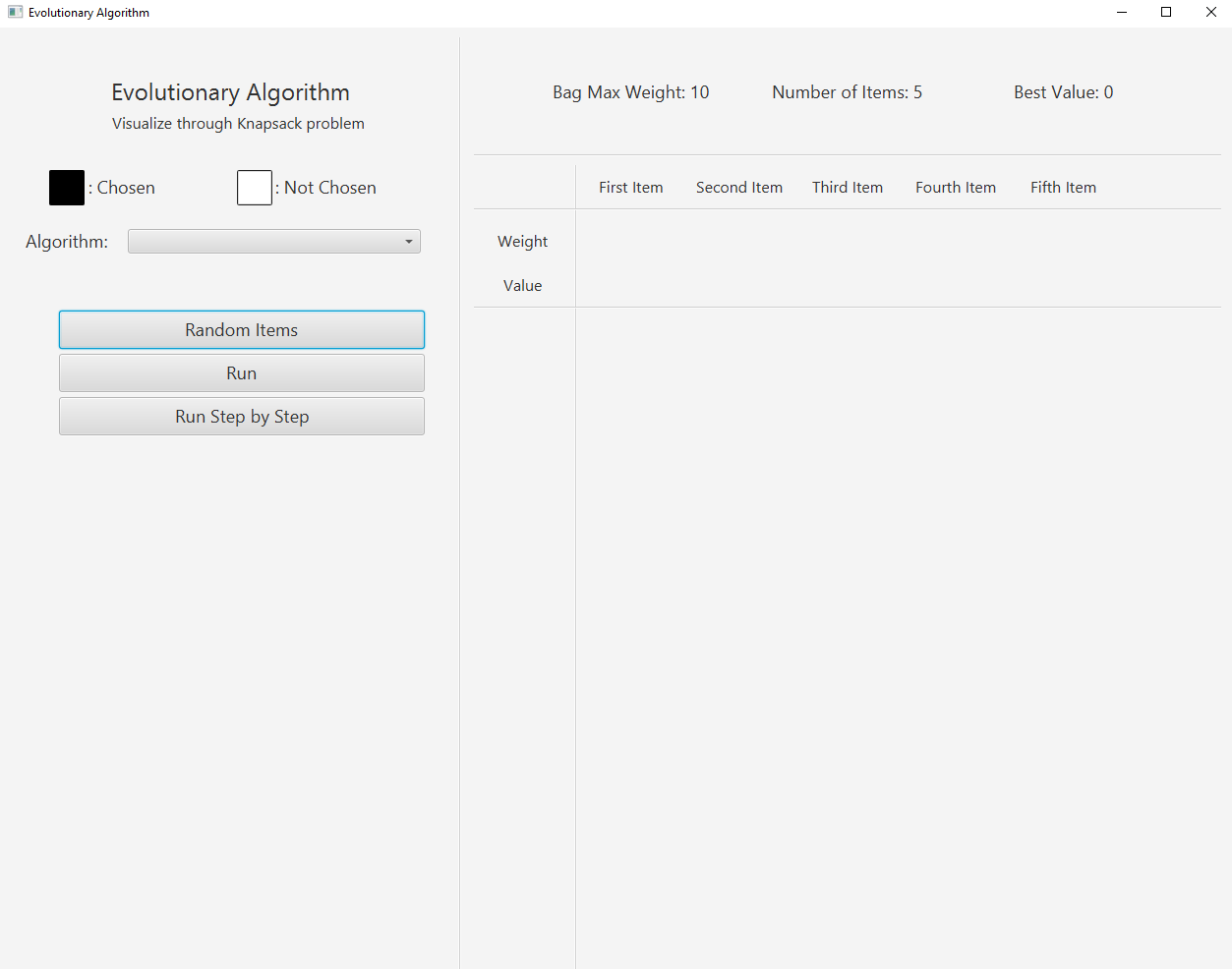
* + 1. **PSO Package**



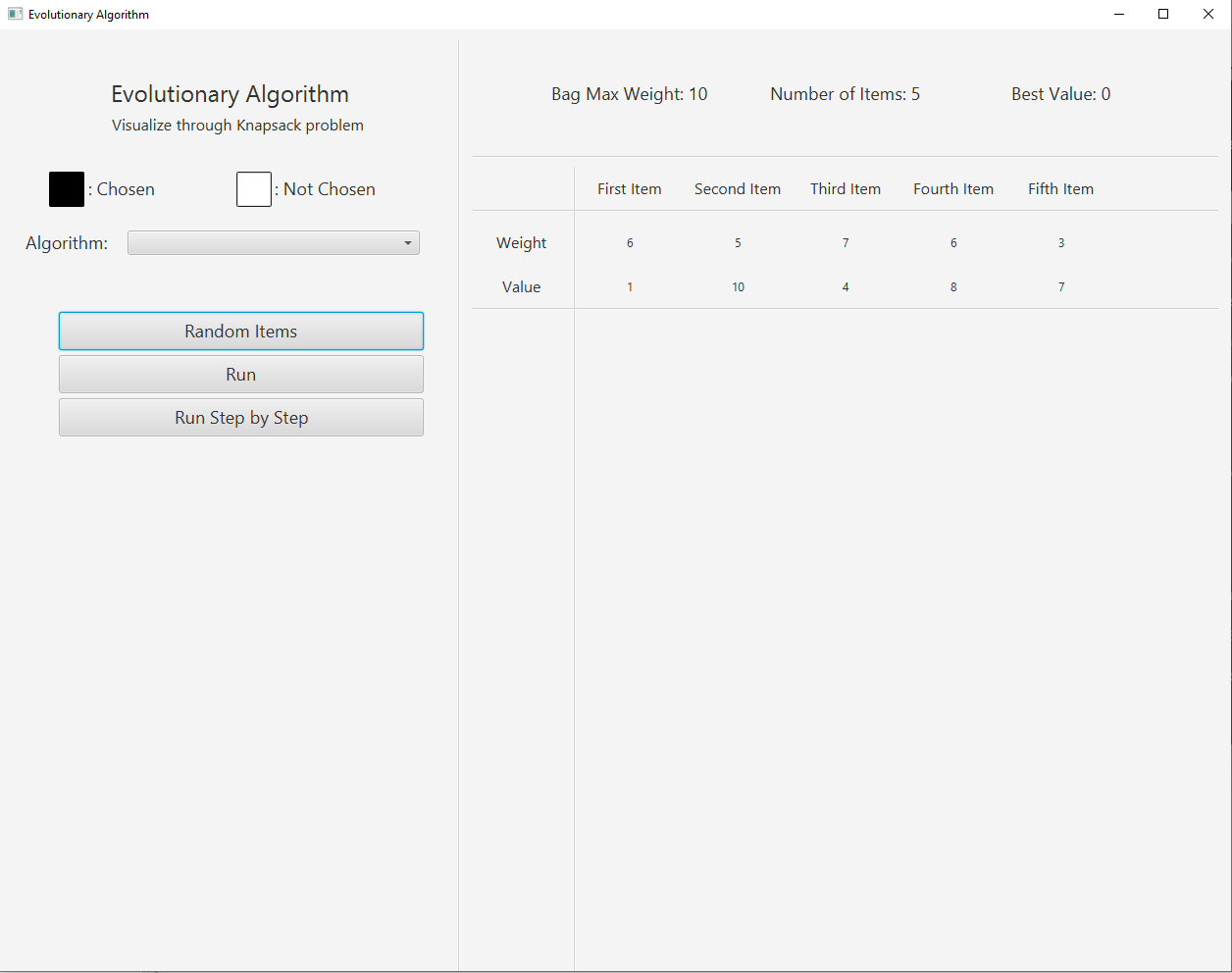
* + 1. **Hill Climbing Package**



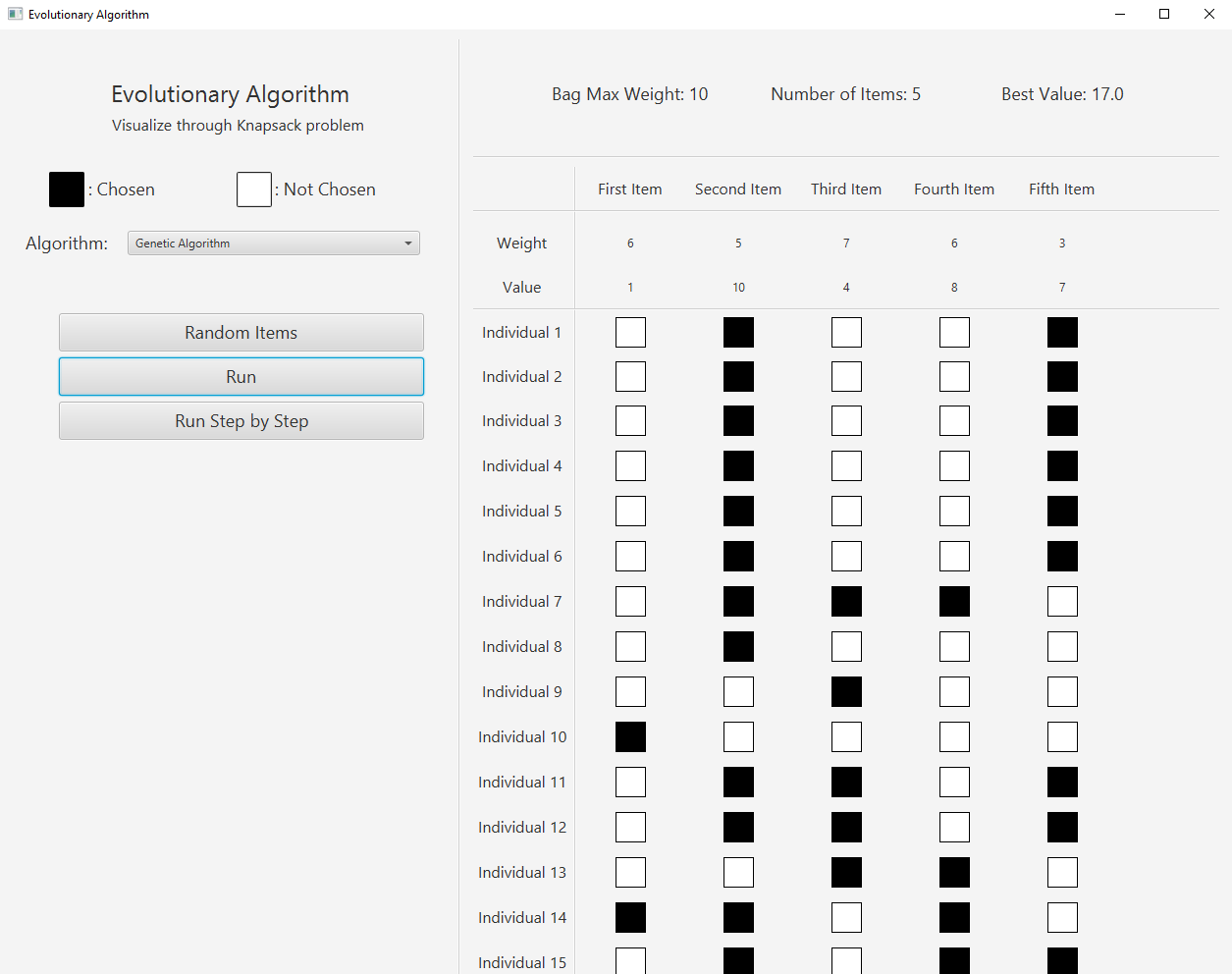
1. **Program Demo**
   1. **The Interface**

****

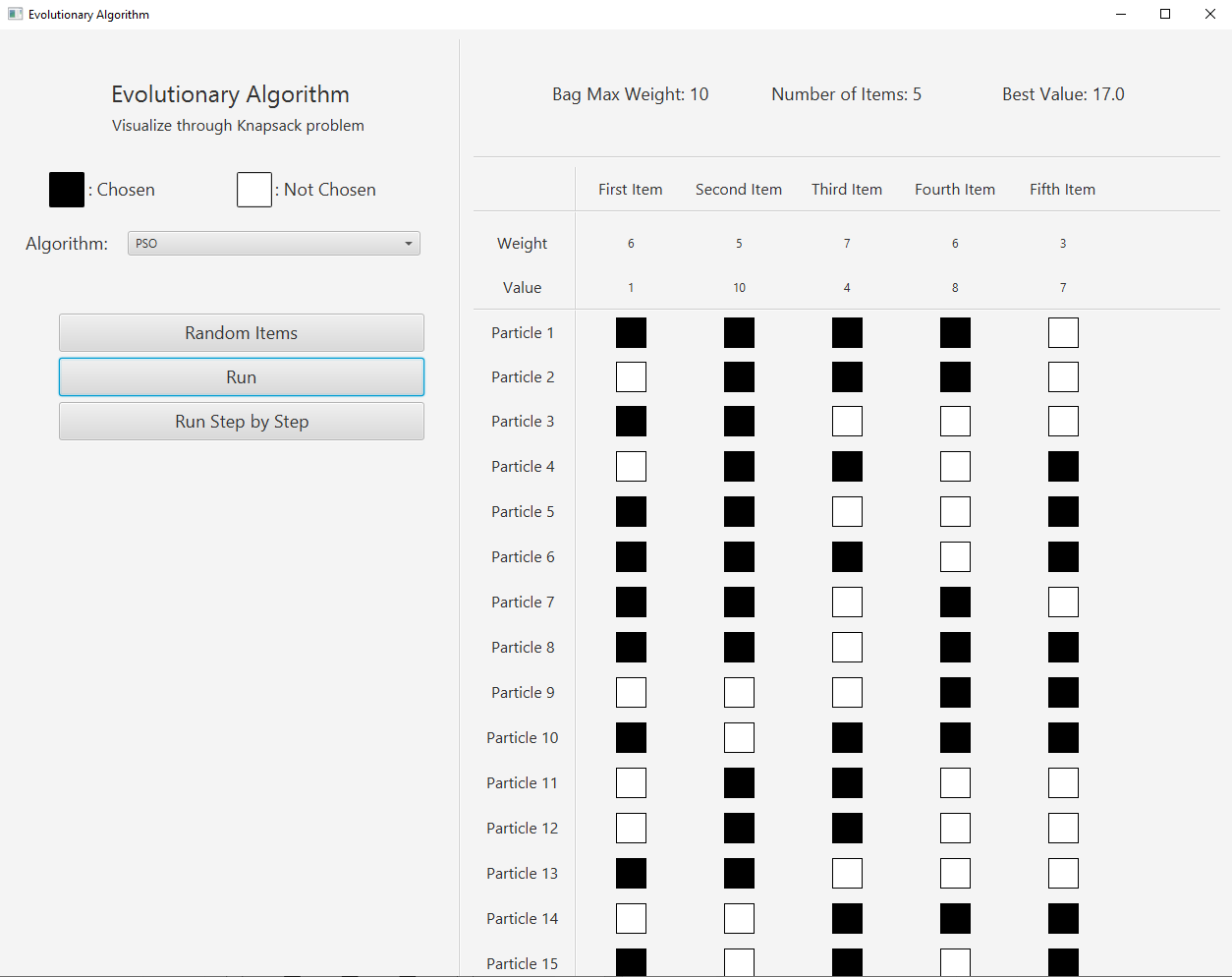
* 1. **Random Items**

****

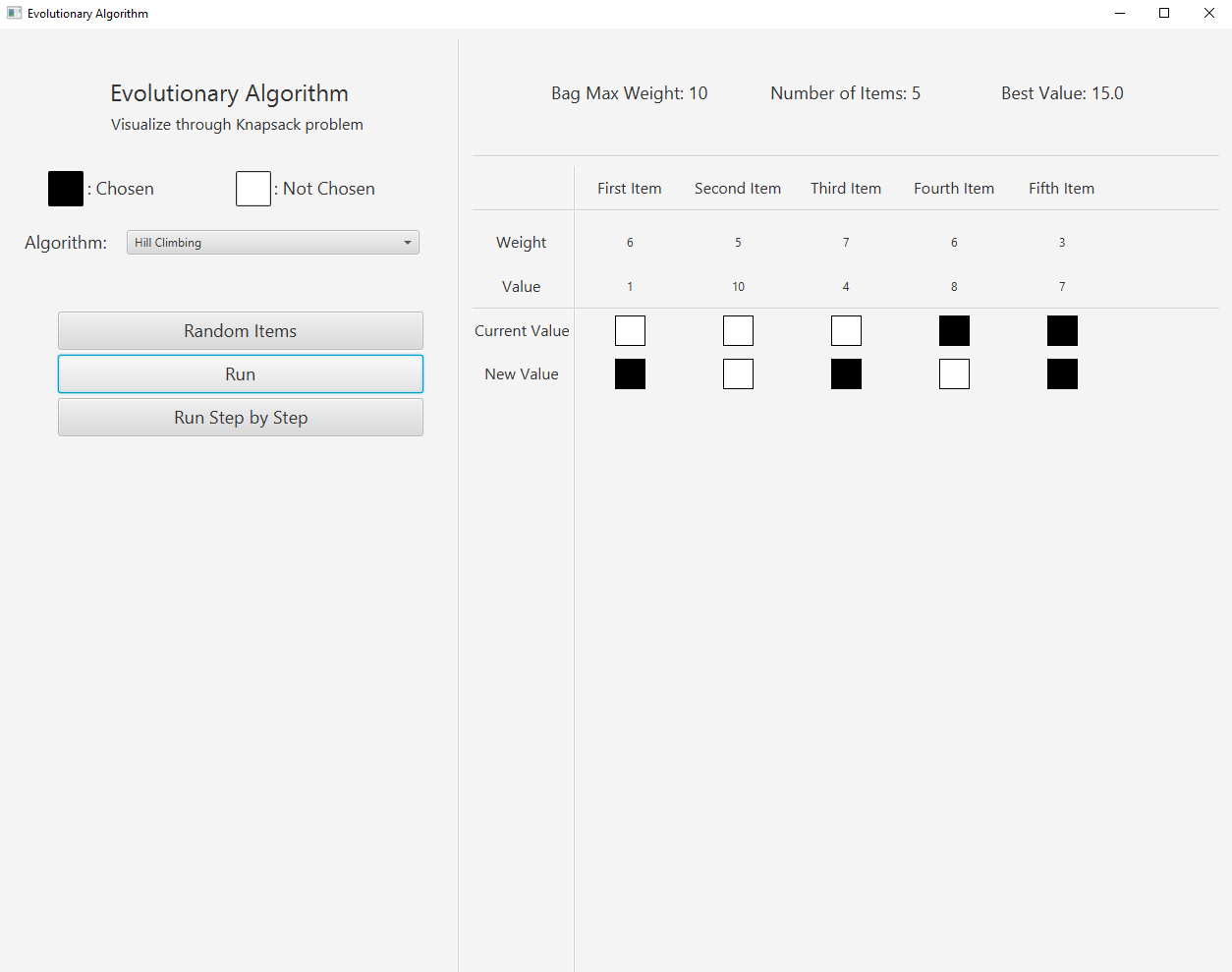
* 1. **Run**
     1. **Genetic Algorithm**

****

* + 1. **PSO**

****

* + 1. **Hill Climbing**

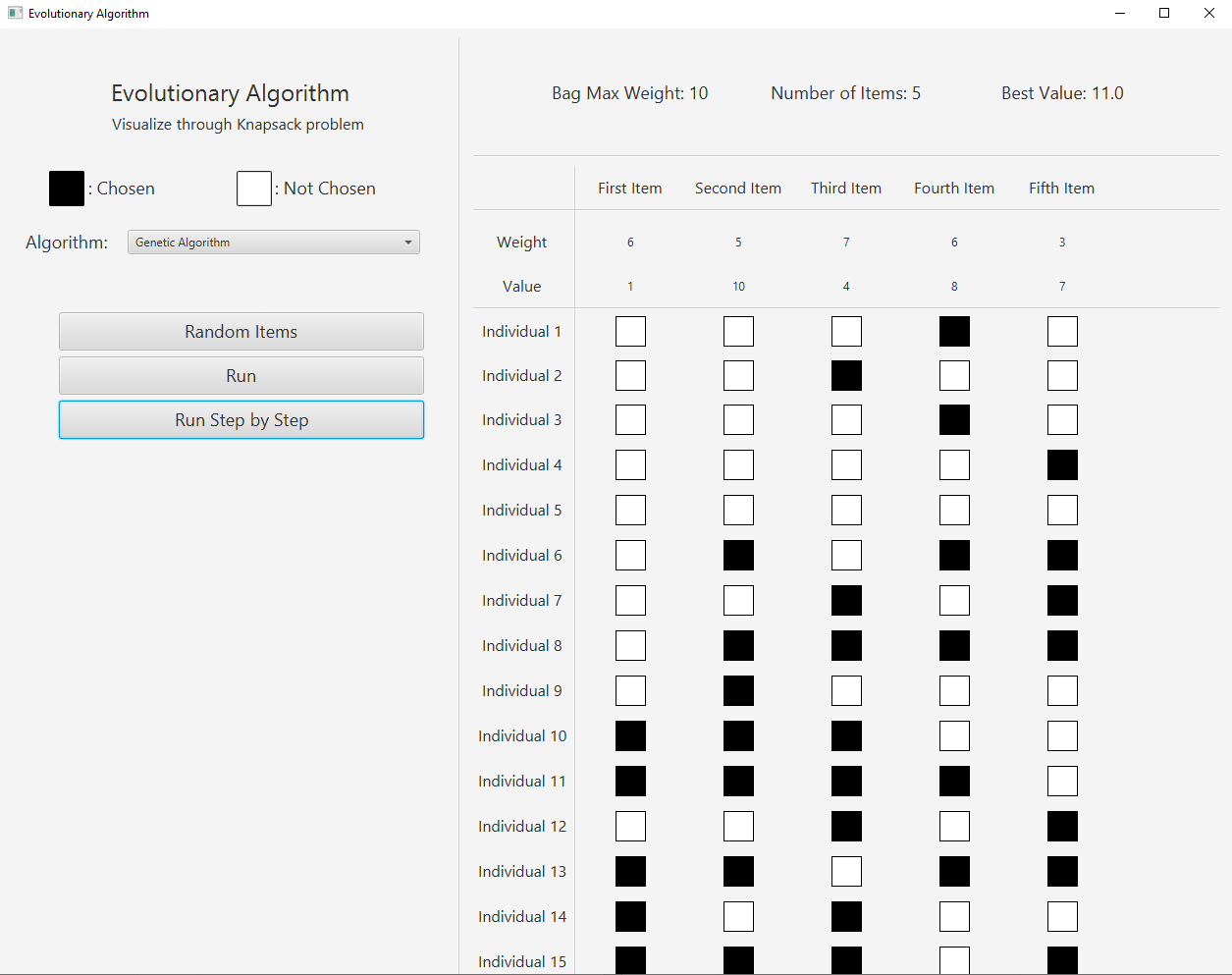
****

* 1. **Run Step by Step**

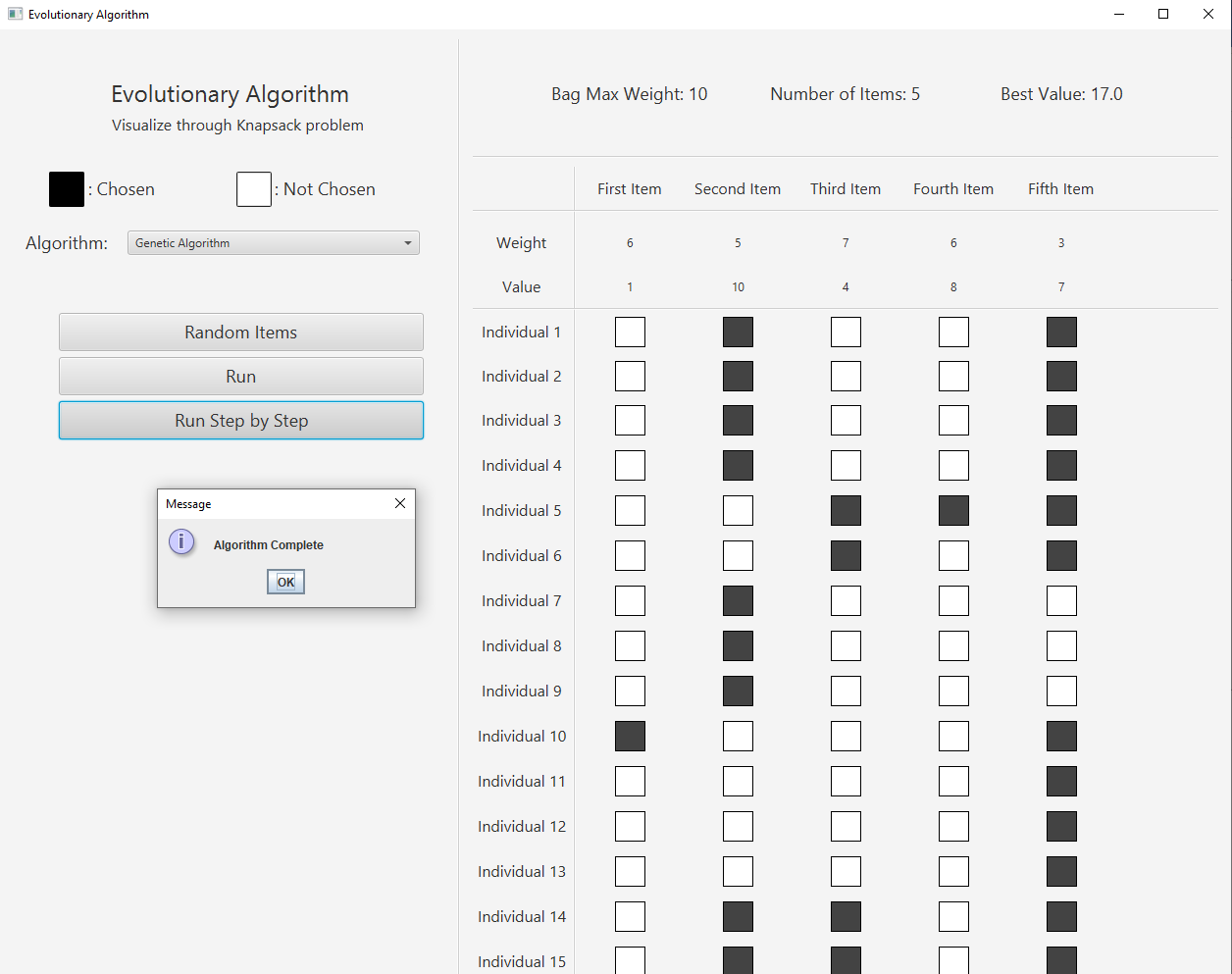
**NOTE:** Each algorithm has to go through many iterations to get to the result and it’s impossible to show them all so I just show the first and the final step

* + 1. **Genetic Algorithm**

1. **First Step**

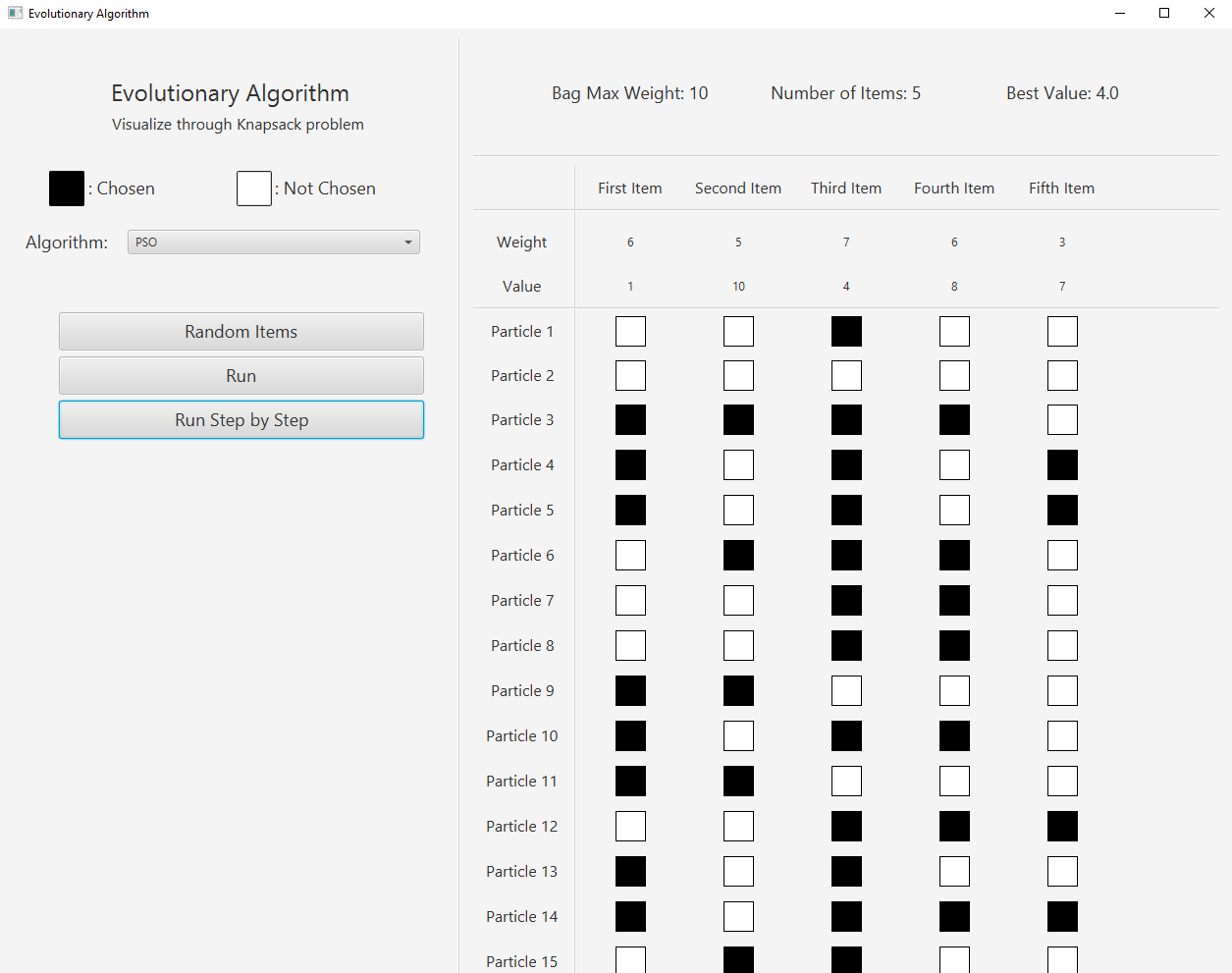
****

1. **Final Step**

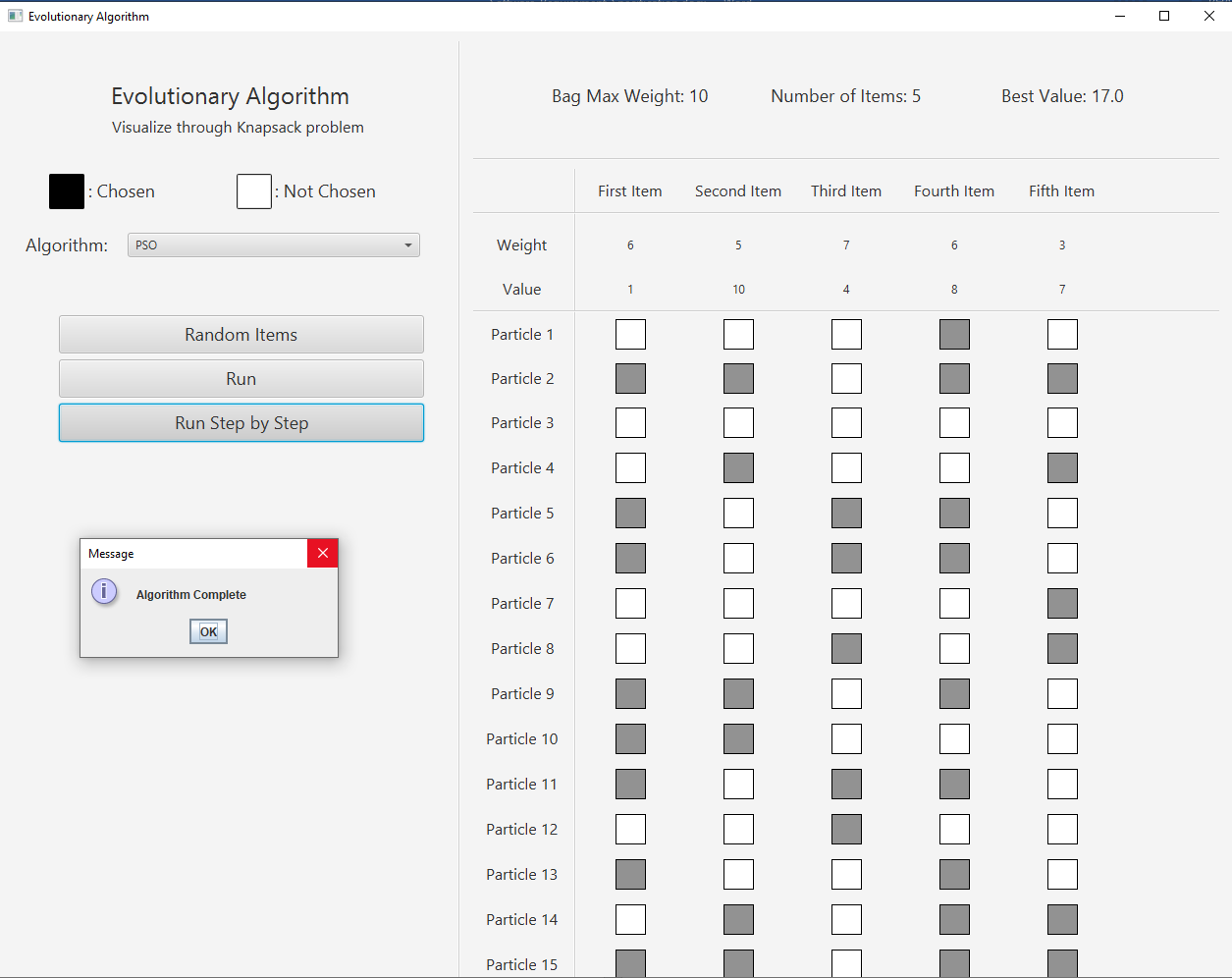
****

* + 1. **PSO**

1. **First Step**

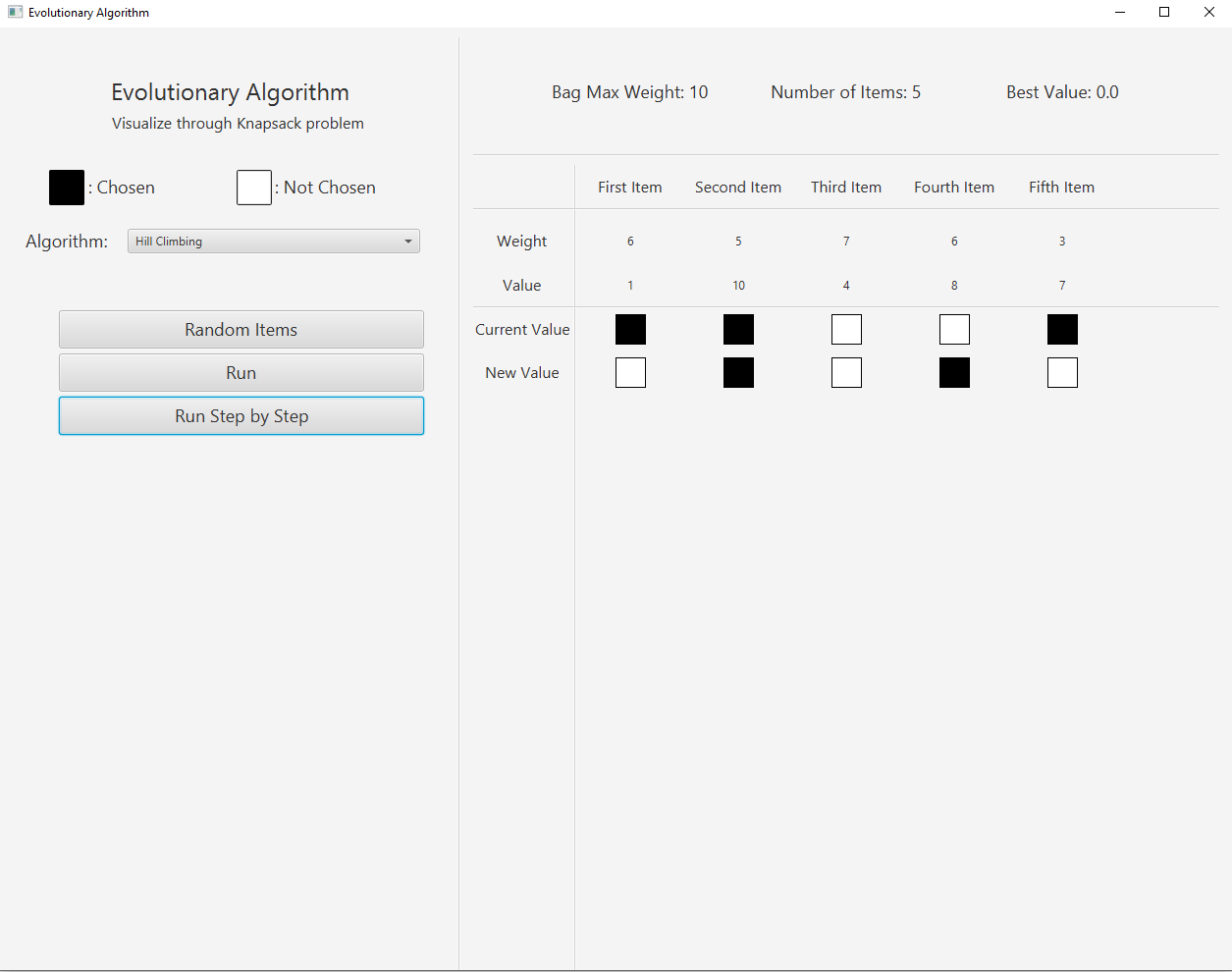
****

1. **Final Step**

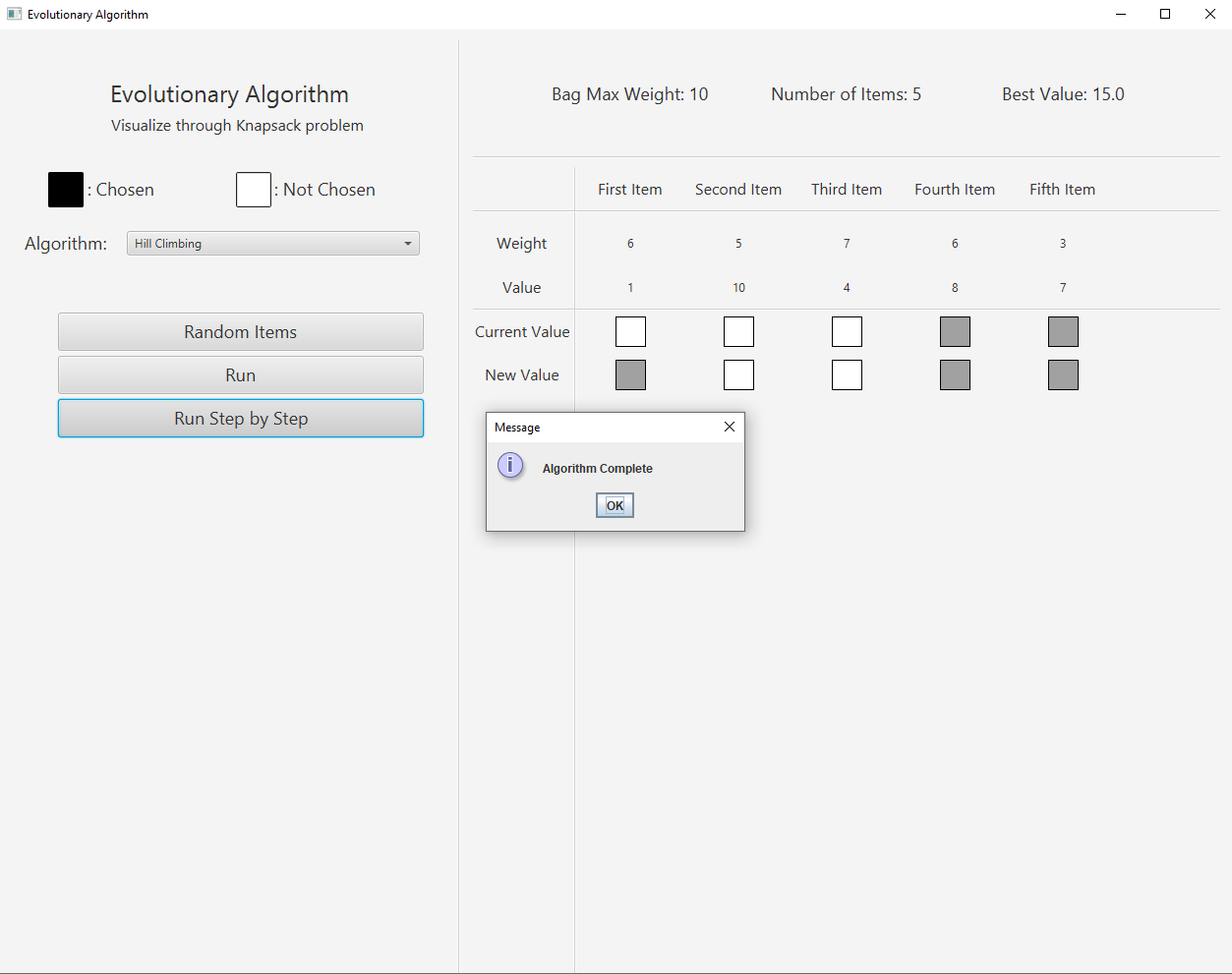
****

* + 1. **Hill Climbing**

1. **First Step**

****

1. **Final Step**

****