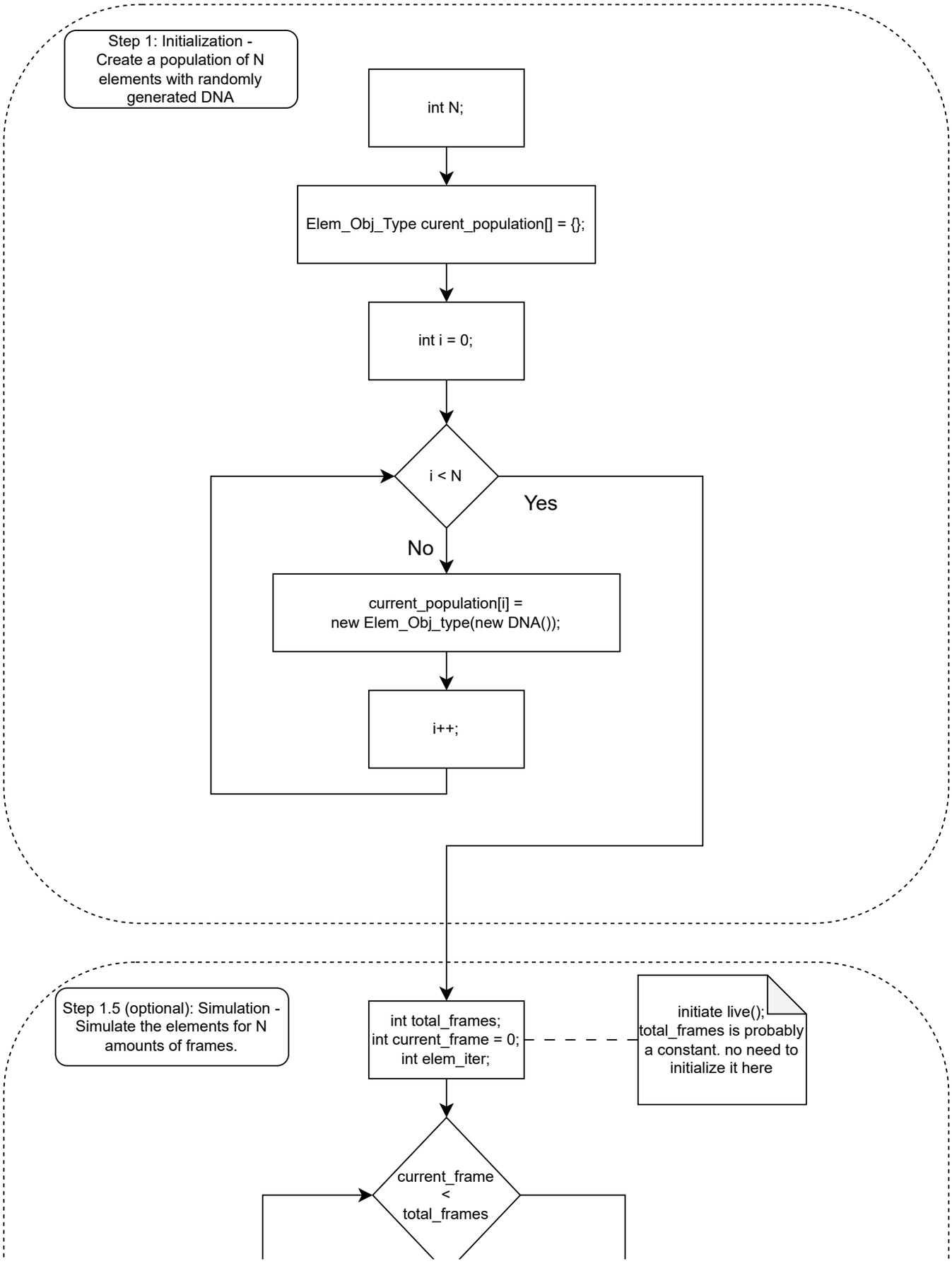
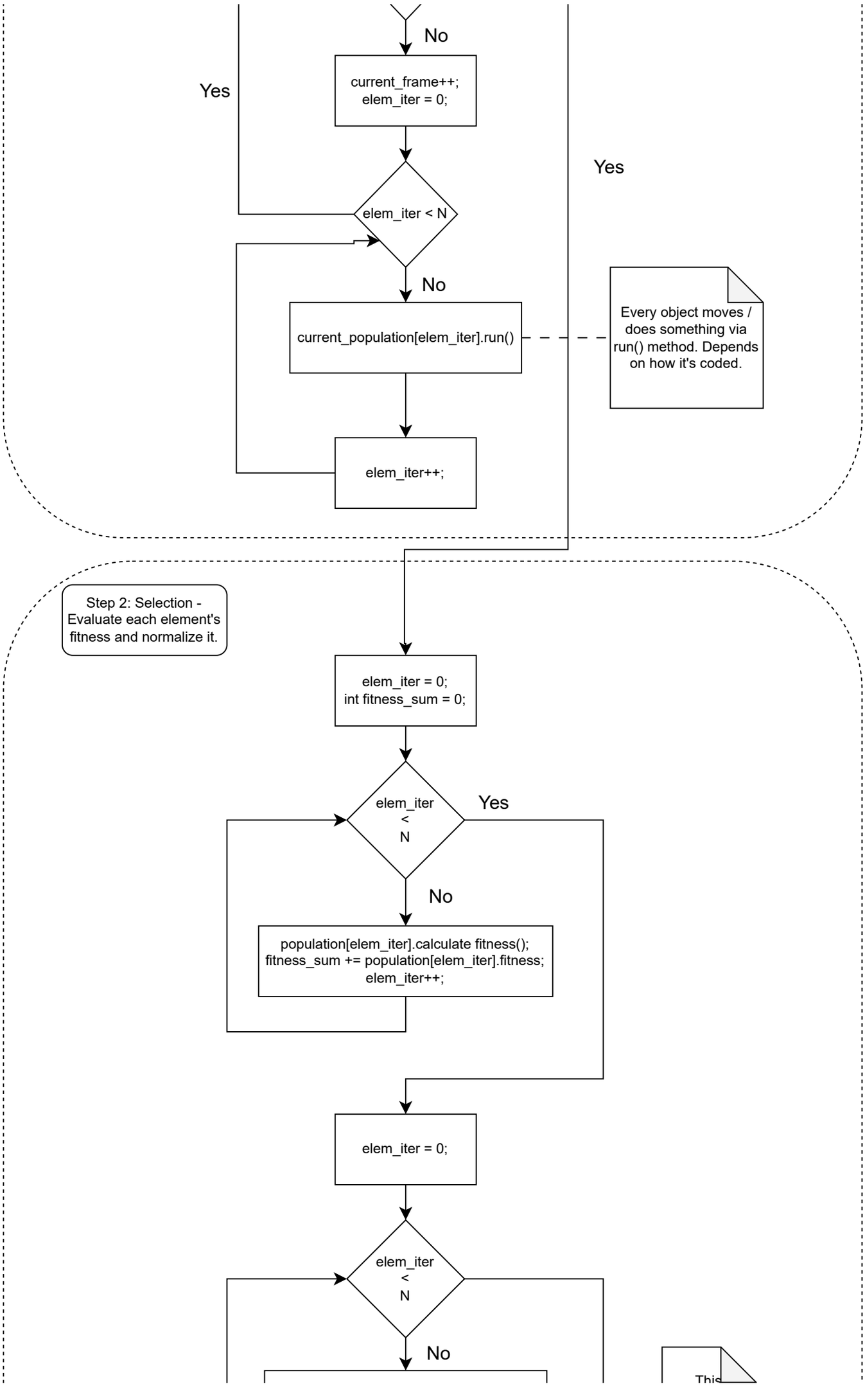
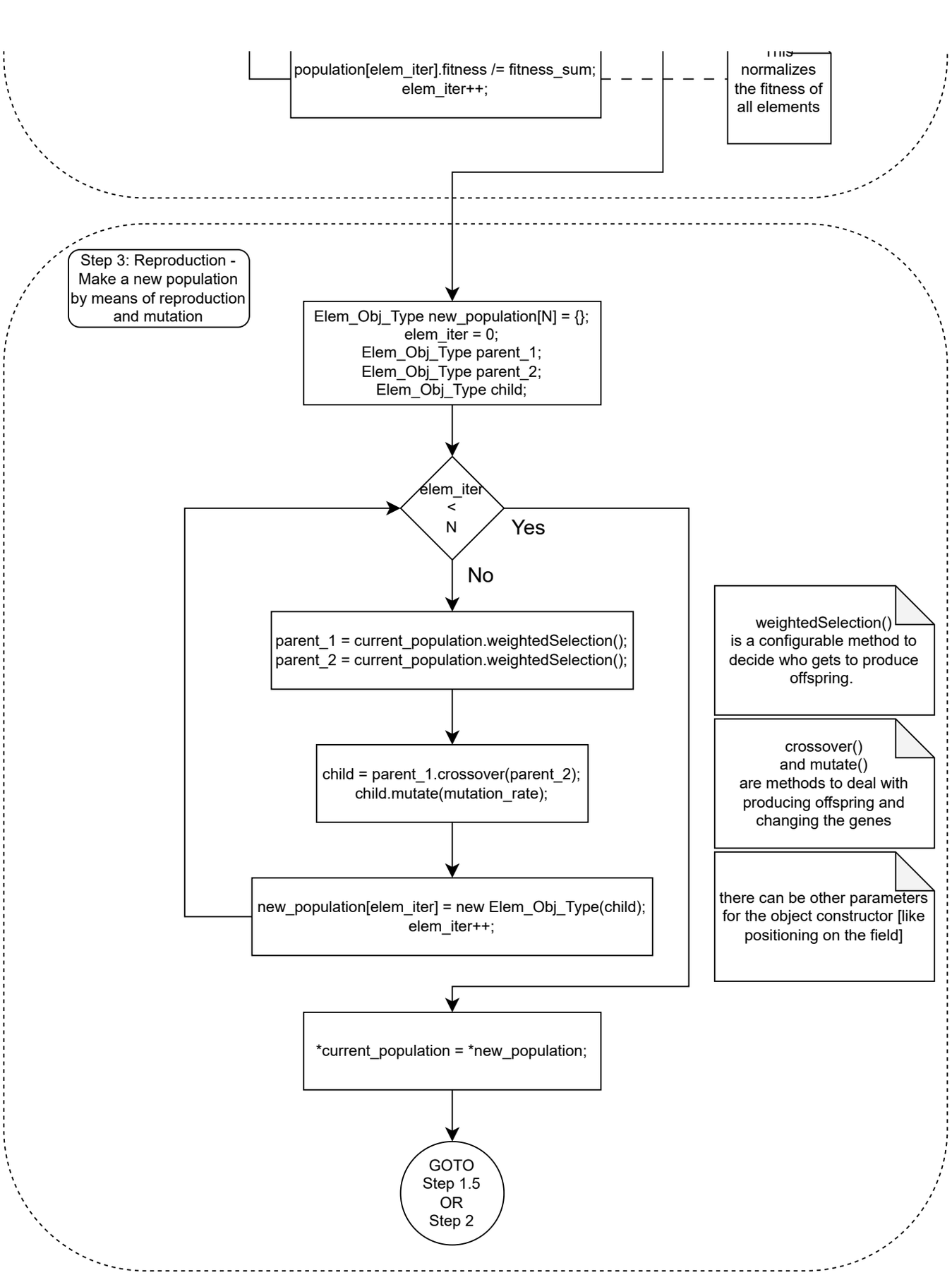

Step 1: Initialization - Create a population of N elements with randomly generated DNA
int N;
Elem_Obj_Type curent_population[] = {};
int i = 0;
i < N
Yes
No
current_population[i] = new Elem_Obj_type(new DNA());
i++;
Step 1.5 (optional): Simulation - Simulate the elements for N amounts of frames.
int total_frames;
int current_frame = 0;
int elem_iter;
initiate live();
total_frames is probably a constant. no need to initialize it here
current_frame < total_frames


No
current_frame++;
elem_iter = 0;
Yes
Yes
elem_iter < N
No
current_population[elem_iter].run()
Every object moves / does something via run() method. Depends on how it's coded.
elem_iter++;
Step 2: Selection - Evaluate each element's fitness and normalize it.
elem_iter = 0;
int fitness_sum = 0;
elem_iter < N
Yes
No
population[elem_iter].calculate fitness();
fitness_sum += population[elem_iter].fitness;
elem_iter++;
elem_iter = 0;
elem_iter < N
No


population[elem_iter].fitness /= fitness_sum;
elem_iter++;
This normalizes the fitness of all elements
Step 3: Reproduction - Make a new population by means of reproduction and mutation
Elem_Obj_Type new_population[N] = {};
elem_iter = 0;
Elem_Obj_Type parent_1;
Elem_Obj_Type parent_2;
Elem_Obj_Type child;
elem_iter < N
Yes
No
parent_1 = current_population.weightedSelection();
parent_2 = current_population.weightedSelection();
child = parent_1.crossover(parent_2);
child.mutate(mutation_rate);
new_population[elem_iter] = new Elem_Obj_Type(child);
elem_iter++;
*current_population = *new_population;
GOTO Step 1.5 OR Step 2
weightedSelection() is a configurable method to decide who gets to produce offspring.
crossover() and mutate() are methods to deal with producing offspring and changing the genes
there can be other parameters for the object constructor [like positioning on the field]