ALIFE AND GENETIC ALGORITHMS: GENOTYPES AND PHENOTYPES

1. Artificial Organisms, Genotypes and Matrices

This exercise will focus on the generation and evolution of a population of artificial organisms using a genetic algorithm. The program you will develop in the next two weeks will allow you to evolve the type of organism that you like most by selecting and reproducing the best organisms in each generation. This simulates the process of natural evolution where you decide who dies and who survives and reproduce.
Let's assume the properties of an organism, such as a stickman, can be defined by a series of numbers corresponding to the following 5 genes:

- Gene 1st : type of head (3 shapes)
- Gene 2nd :size of the body (width between 1 and 20 points)
- Gene 3rd : number of right arms (from 1 to 5)
- Gene 4th : number of left arms (from 1 to 5)
- Gene 5th : height of legs (from 50 to 100 points)

Now let's assume that there is a group (population) of 4 organisms. The best way to represent the genotype of the whole population is by using a two-dimensional matrix where each row corresponds to an individual organism, each column to a gene, such as in the genotype matrix below.

GENOTYPE[1]	*Gene 0 (Head)*	*Gene 1 (Body)*	*Gene 2 (R arms)*	*Gene 3 (L arms)*	*Gene 4 (Legs)*
Organism 0	*3*	*1*	*5*	*4*	*51*
Organism 1	*2*	*12*	*1*	*1*	*98*
Organism 2	*3*	*20*	*2*	*2*	*80*
Organism 3	*1*	*9*	*3*	*1*	*62*

Once you have generated the initial random genotype, you can use it to update the organism's phenotype, that it its physical (visual) properties. For example, if you are using a movieclip "facetype_mc" with 3 different graphics (one for each type of face, respectively in the keyframes 1, 2 and 3) you can use the following ActionScript instruction to change the phenotypical property of the face of organism number 1 using the numerical information on the `genotype` matrix array:

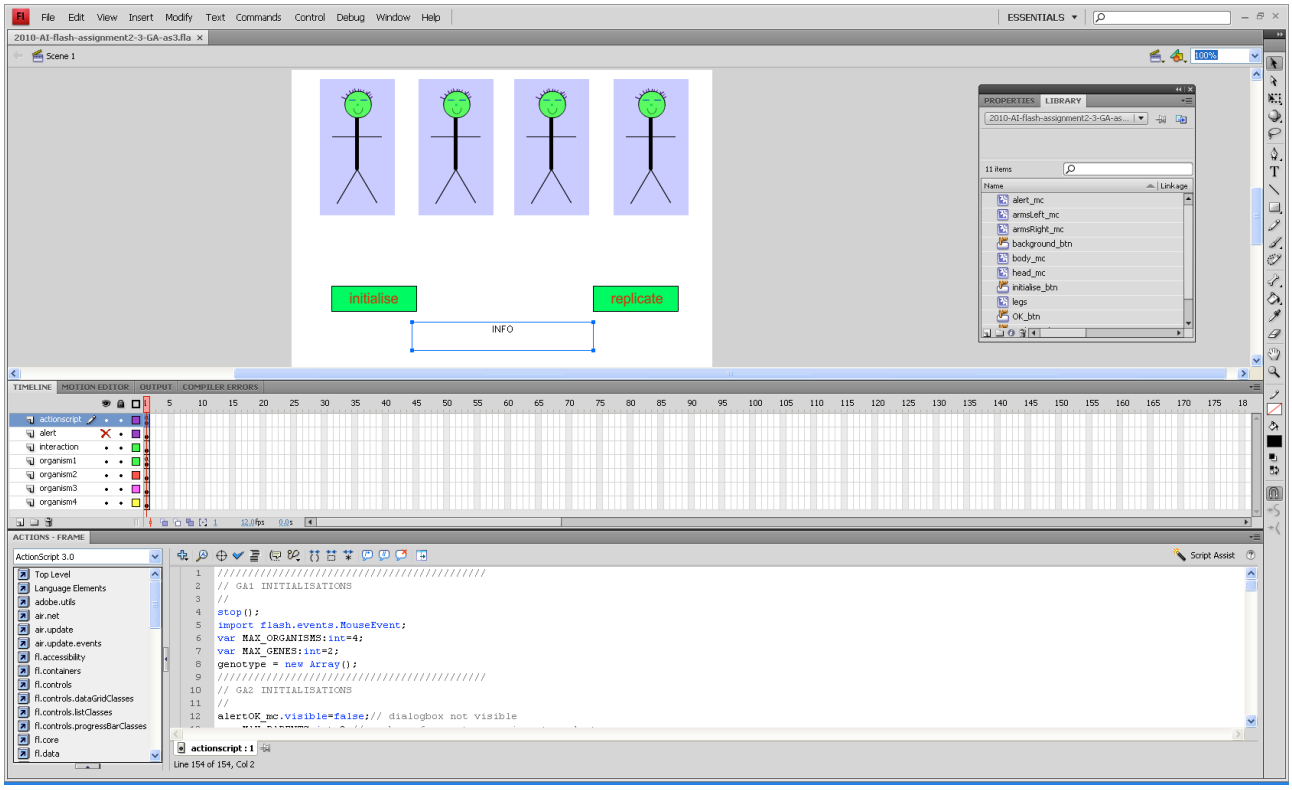
```
facetype_mc.gotoAndStop(genotype[1][0]);
```

We will now prepare a Flash movie to create the population of 4 organisms, and then prepare the function to initialise the genotype and show the phenotype of the whole population.

[1] Remember that since we have to use Flash ActionScript, the first element of each array (e.g. rows of columns of a matrix) is 0, the second is 1 etc.

1.1 Prepare Flash File

The Flash movie shall have an appearance similar to that of the following figure:



The Flash movie requires the following structure and components[2]:

- Use 6 layers. The first will contain all the ActionScript code, the second all buttons for interaction, and the other 4 for each of the organisms.
- One button to perform initialisation of genotype. First create a new button symbol, and then make an instance of it named “initialise_btn” in the second layer.
- One button to perform replication of genotype. Create another new button symbol, and then make an instance of it named “replication_btn” in the second layer.
- Four buttons to allow users to click and select each organism. Create a new rectangular button symbol as in the figure above, and then make 4 instances of it named “select_0”, “select_1”, “select_2”, “select_3” respectively in layers 3, 4, 5, and 6.
- Four faces. Create one movieclip symbol for the 1st gene that contains 3 different keyframes. Each keyframe has the graphics of a different type of face. Make four instances of it respectively in layers 3, 4, 5, and 6 as to start forming a stickman over each of the 4 selection buttons. Name the four instances o0_head, o1_head, o2_head, o3_head.
- Four bodies. Create one movieclip symbol for the 2nd gene that contains only one keyframe with the main body of a stickman. Create four instances of it in layers 3-6 respectively named o0_body, o1_body, o2_body, o3_body.
- Use a similar approach for creating the right arms (e.g. from 0 to 3), the left arms, and the legs.

Note that the code provided below will only work for the first two genes. You will have to adapt it to the remaining 3 genes depending on the type of symbols used and their instances’ names.

[2] Please use the same names as those suggested for all the instances that you are going to create below. This is necessary for the sample ActionScript 3.0 code provided here to work.

Select the first keyframe of layer 1. Open the Action window to add the following lines of code needed to initialise the movie:

```
var MAX_ORGANISMS:int=4;
var MAX_GENES:int=2;
var genotype:Array = new Array();
var MAX_PARENTS:int=2;// number of parents organisms to select
var MAX_CHILDREN:int=2;// number offspring per selected parent
var generation:int=0;// to count the generation being shown
var list_selected:Array = new Array();// list of selected org.
var number_selected:int;
```

The four capital letter variables are CONSTANTS, that is variables that do not change their value duding the whole movie. These four constants respectively define the maximum number of organisms in the population, the number of parents to be selected, the children that each will generate, and the number of genes (we use 2 here, but this will have to become 5 after you implement the remaining three genes).
The other is a global array (matrix) variable for the population genotype.

1.3 Random initialisation of genotype

You will remember from the 2nd week lecture that the first part of a genetic algorithm consists in the random initialisation of the genotype. So we need to write the function that does the random initialisation of the genotype. Type in the Action window of Layer 1 the function below. Note the use of the `.slice()` method to copy the temporary vector array `genotype_1organism` (used to generate one organisms at a time within the `for` cycle) into the main global matrix array `genotype`.

```
function random_genotype_initialisation():void {
     generation=0;
     number_selected=0;
     var org:int;
     var genotype_1organism:Array = new Array();
     for (org=0; org<MAX_ORGANISMS; org++) {
          list_selected[org]=0;
          genotype_1organism[0]=randomRangeInteger(1,3);
          // gene1 = head type (3 types)
          genotype_1organism[1]=randomRangeInteger(1,30);
          // gene2 = body width (1 to 20 points)
          genotype[org]=genotype_1organism.slice();
          // EXTEND THIS FUNCTION TO INITIALISE THE OTHER 3 GENES
     }
}
```

The above function will also need the definition of the `randomRangeInteger()` function, already used in the previous week's exercise.

```
function randomRangeInteger(min:int, max:int):int {
// returns a random integer number between the extremes min and max
   var randomNum:int = Math.floor(Math.random()*(max-min+1))+min;
```

```
    return randomNum;
}
```

1.4 Visualisation of phenotypes

Once the genotype has been randomly initialised, we need to change the visual properties of each organism (i.e. its phenotype) to match the gene values. Add the following function:

```
function show_phenotype():void {
    var org:int;
    var org_property:String;
    for (org=0; org<MAX_ORGANISMS; org++) {
         org_property="o"+org+"_head";
         this[org_property].gotoAndStop(genotype[org][0]);
         org_property="o"+org+"_body";
         this[org_property].width=genotype[org][1];
         // EXTEND THIS FUNCTION TO VISUALISE THE OTHER 3 GENES
    }
}
```

Now link the two random initialisation and visualisation function into the `onRelease` ActionScript of the button “initialise_btn” by typing the following function in the Action window of Layer 1:

```
initialise_btn.addEventListener(MouseEvent.MOUSE_UP, initialise);

function initialise(event:MouseEvent) {
    random_genotype_initialisation();
    show_phenotype();
}
```

Test the movie. Note that every time you click on the initialisation button, you see a new random population of organisms.
To visualise the numerical values on the new genotypes you are advised to add some text field where you show the content of the global genotype matrix (as in last week’s exercise).

ASSESSMENT EXERCISE 1

Generation of genotypes and phenotypes

Now we will practice on the genotypes and phenotypes of more original organisms. You may want to play with a population of artificial organisms such as insects, aliens, cars, or any other creative agent you can think of. Do not use stickman-type organisms for the coursework submission!
For example, if you choose to work with complex faces of known or made-up individuals, you must define the number of genes that control a face and their range of variability. The following is an example of genes for faces.

Gene	Range
Number of eyes	0, 1, 2 (and more?)
Eye size	Horizontal size: from 5 to 15 pixels
Nose length	1-20 pixels
Mouth shape (smiling or sad)	From –10 (sad) to +10 (smile) points
Face shape	Vertical diameter: 40-80 pixels

You first have to create the symbols and instances necessary to draw a face, such as the eyes, mouth, nose, and face shapes. Then create a population of **15 organisms** in total and modify ActionScript functions from this handout to generate random populations of organisms.
You are encouraged to be creative in the choice of the organisms and their visualisation, as this will also be part of the marking grid (see below).

Assessment criteria

This first part of the Assignment 1 exercise is worth it 20% of the module mark.
To mark this exercise, the following questions will be used:

a) Does the program include a correct implementation of the genotype initialisation and phenotype visualisation.
b) Does the program correctly use arrays for the organisms' genotypes?
c) How complicated is the organism's genotype and phenotype?
d) Does the program support a good and friendly interaction with the user to understand the evolutionary process?
e) How complex is the graphics for representing organisms' phenotypes?
f) How original/creative is the organism's representation?
g) Has the code been optimised to include customised features?