# Worksheet – Genetic Algorithms Unity

## Overview

### Objectives:

1. Explore code for algorithm
2. Test and Run the first scene
   1. Edit metadata of algorithm
3. Test and Run the second scene
   1. Edit metadata of algorithm

## Preliminary

Primarily this tutorial work will require the use of Unity. The project has been tested and developed in **Unity Version 2021.1.18F.1**.

The project on MyLearningSpace is called **Genetic Algorithm Unity**.

If you need help setting up the software, we have a tool called AppsAnywhere that allows you access to some of the software we use. You can use this to install some of the software we use. <https://myapps.abertay.ac.uk/>

## Getting Started

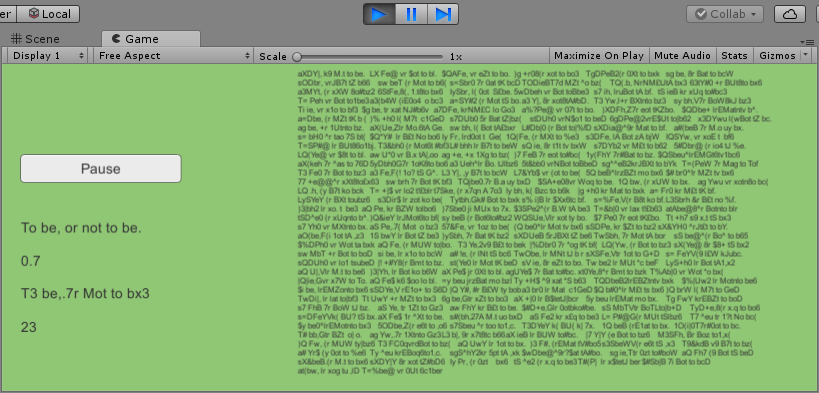
You can also download a zipped version of the project from MyLearningSpace called **Genetic Algorithm Unity**.

Project has been run and tested in Version 2021.1.18F.1

Once the project is open, make sure to open **Text\_GA\_Scene**

This can be found in *Assets* -> *Scenes*

## Running the Text Scene



You should see a scene similar to the one above. The algorithm has a target text field that it wants to find and will evolve a population of strings until it reaches this point.

Each generation has a population of 200 strings.

Each member of the population is a sequence of characters, being 20 characters in size. This allows us to create a string that we can compare with the target. Each Gene is a single character within that string and we will edit these Genes with each generation.

Each generation, we go over each member of the population, check how close the genes match with our target string, and assign each one a fitness value. Once this is complete, we perform crossover and move onto the next generation.

If a member of a generation matches our target string exactly, we tell the algorithm to stop. We have reached our end condition.

## Steps Algorithm Takes

### Crossover

The algorithm performs crossover by selecting two members of the current generation and combining random elements on their genes. The selection process is done by selecting the fittest members of the current generation. If a suitable parent cannot be found, we select a random one.

Combination of genes is done randomly.

### Mutation

Mutation is done on every Gene of each new member of the next generation. There is a very low chance of mutation, around 0.01% chance.

As the Genes are a string, the mutation has a change to randomly change one character to another randomly selected character.

### Selection

We use the Crossover and Mutation to create 200 new strings. We replace the existing generation with a new one. Normally we would keep some between generations, but I have deliberately omitted in this case.

## Second Scene

Open the second scene inside the Unity project. It should be called:

Jumping\_GA\_Scene

You can find the file below if you want to open it manually:

Assets -> Scenes

Press the Play button in the editor and Press Start in the game to begin the simulation.

### What You Should See

A picture containing graphical user interface

Description automatically generated

In this scene, the blocks are each controlled by a Gene in a Genetic Algorithm. Their aim is to combine a number of float values together to give a final ‘Jump’ value.

Their fitness is based on how high they jump. The higher the better, however if they collide with the red zone then they loose fitness. This AI will learn the exact value to make the best jump it can. The algorithm is the same as the previous example but with a different fitness and mutation function.

## Exercises

### Text Scene

Open the Text\_GA\_Scene and edit the target string the AI is trying to solve. How many generations on average does it take to complete?

1. Change the mutation rate and time to see if it improves the rate of completion.
2. Change the population size and time in a similar manner.
3. What changes other than the two above (any metadata that affects the algorithm) can be changed to improve the speed of training?
   1. You can also look into changing the fitness, mutation and crossover functions.

### Jumping Scene

Open the Jumping\_GA\_Scene.

1. Change the simulation so the Genes are multiplied together to give a jump value rather than adding.
   1. Does this approach change how long it takes to learn?
   2. Try adding all but two Genes. The last two Genes multiply against the sum of the others. Does this improve training time?
2. Investigate what metadata can be changed to improve the time for a viable solution.
   1. Mutation rate is a good start.
   2. The aim is for the agents to jump as high as possible but not touch the red zone.

### Own Genetic Algorithm

Create a Genetic Algorithm using the same library that solves any problem or gameplay mechanic you want.

**E.g.**

Fire an arrow (can just be a cube) at an angle to hit another target. You could use the following rules:

Fire 100 arrows at once. (Generation size)

No collision between arrows.

Fitness is distance from the arrow to the target.