|  |
| --- |
| Kingswood School |
| Survival of the fittest teaching aid |
| Comp4 project |

|  |
| --- |
| 01robinson |

Contents

[Section 1: Analysis 3](#_Toc382463947)

[The problem: identification and background 3](#_Toc382463948)

[Description of available tools 4](#_Toc382463949)

[Identification of prospective users 7](#_Toc382463950)

[Identification of user needs and acceptable limitations 7](#_Toc382463951)

[Data sources and destinations 9](#_Toc382463952)

[Data volumes 9](#_Toc382463953)

[Analysis Data Dictionary 9](#_Toc382463954)

[Structure and Data Flow Diagrams 10](#_Toc382463955)

[Entity-relationship (E-R) model 10](#_Toc382463956)

[Object analysis diagram 10](#_Toc382463957)

[Objectives for the proposed system 10](#_Toc382463958)

[Appraisal of potential solutions 11](#_Toc382463959)

[Proposed solution 13](#_Toc382463960)

[Justification of chosen solution 13](#_Toc382463961)

[Section 2: Design 15](#_Toc382463962)

[Overall system design 15](#_Toc382463963)

[Description of modular structure of system 17](#_Toc382463964)

[Definition of data requirements (Design Data Dictionary) 26](#_Toc382463965)

[Description of record structure 28](#_Toc382463966)

[File organisation and processing 29](#_Toc382463967)

[Validation required 29](#_Toc382463968)

[Identification of storage media 29](#_Toc382463969)

[Identification of suitable algorithms for data transformation, pseudo code of these algorithms 29](#_Toc382463970)

[Class definitions (diagrams) and details of object behaviours and methods 39](#_Toc382463971)

[User interface design (HCI) rational 41](#_Toc382463972)

[UI sample of planned data capture and entry designs 47](#_Toc382463973)

[UI sample of planned valid output designs 48](#_Toc382463974)

[Description of measures planned for security and integrity of data 51](#_Toc382463975)

[Description of measures planned for system security and integrity of data 51](#_Toc382463976)

[Overall test strategy 51](#_Toc382463977)

[Section 3: Technical Solution 52](#_Toc382463978)

[Section 4: System Testing 52](#_Toc382463979)

[Section 5: System Maintenance 52](#_Toc382463980)

[Section 6: User Manual 52](#_Toc382463981)

[Section 7: Evaluation 52](#_Toc382463982)

[Appendix A: Availability of Teaching Materials on Evolution 52](#_Toc382463983)

[Online research 52](#_Toc382463984)

[Interviews 52](#_Toc382463985)

[Sources 54](#_Toc382463986)

[Ants and their behaviour 54](#_Toc382463987)

# Section 1: Analysis

## The problem: identification and background

### Problem identification

The problem addressed in this project is to develop a computer-based tool which will enable secondary school teachers of Biology at a school in Bath to illustrate how evolution occurs in a population. This problem was identified from interviews with Biology teachers which found that the subject of evolution is complex and difficult to teach to younger pupils (aged around 12-13).

These interviews found that pupils in this age range find it difficult to grasp the abstract ideas of evolution as this involves following an argument with several stages to it and a number of novel concepts such as ‘genes’, ‘mutation’, ‘selection’ and ‘inheritance’. The interviews also found that presentations and handouts failed to capture the dynamic nature of evolutionary change and ‘bring it to life’. Further, due to a shortage of suitable tools and materials it is difficult for teachers to set meaningful classroom exercises on evolution and to encourage pupils to do independent learning outside of the classroom.

As a result of these problems in teaching the topic teachers have found that some pupils can have common misconceptions about natural selection. For example, some pupils persist in believing in Lamarckism - the idea that an organism can pass on characteristics that it acquired during its lifetime to its offspring. It was felt, in the interviews, that a computer-based tools could illustrate the process of evolution, enable pupils to have more of a ‘hands-on experience’ and could be an excellent way for pupils to understand for themselves why ideas such as Lamarckism are wrong - rather than being taught simply that they are wrong.

Online research and face-to-face interviews with teachers (see Appendix A for details) has shown that there are many types of videos, handouts, presentations and other written materials currently available for school teachers on the subject of evolution. In addition, there are some computer-based tools which simulate how evolution occurs and which can, for example, allow different timescales and types of populations to be selected and different simulations to be run. These computer-based tools are evaluated and a new solution is developed in this project.

My aim for this project is to create a tool to be used by in biology classes for years 9 to 11 to demonstrate to the pupils the fundamental concepts of evolution to the pupil in a fun and interesting way.

### Background - Teaching context

The teacher who were initially interviewed teach Biology teachers and each have two classes a year of 15 – 20 mixed ability twelve to thirteen year old pupils. Evolution is taught as part of a Combined Sciences course in an introduction to the natural world and a 40 minute lesson is devoted to it. In these lessons the teachers cover the following:

* Some context – the ideas of Charles Darwin.
* A definition of evolution.
* An overview of how it occurs – introducing a number of ideas including the existence of differences within a population, members of a population being more/less suited to their environment, natural selection and the lengthy timescales over which evolution generally occurs.
* Examples are introduced – these can include more material on Darwin, on his travels and the Galapagos Islands, and well-known examples of evolution such as Darwin’s finches or the evolution of butterflies in contemporary Britain and their response to pollution.
* What evolution is not/common misconceptions – Lamarckism.
* Lessons are a mixture of explanation by the teacher, using video or written materials to illustrate the subject, taking questions and classroom discussion.

Computers with internet access are available during lessons with one computer for every pupil.

## Description of available tools

The Biology teacher who has been interviewed is not currently using any computer-based simulation tools as part of their lessons on evolution. However, some tools do exist and this section assesses the leading examples.

The tools assessed fall into two categories: a) tools focused specifically on evolution, and b) more broadly-based tools - for example considering the impact of disasters on animal populations. The table below summarises the tools which have been assessed.

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Assessment** |
| **“Who Wants to Live a Million Years?”** | * Aimed at 12- 14 age group * Focused on evolution * Learning and simulation sections, together with a quiz, glossary and Darwin biography * Shows impact of a change in environment on a population and the effects of natural selection | * Learning and simulation sections * Poor treatment of mutations * Poor representation of gradual evolution processes * Not very engaging/fun * Minimal user interaction * Overall – too high level |
| **“Stop Disasters!”** | * Aimed at 16+ age group * Produced by the United Nations * Focused on disaster recovery * Shows the impact of disasters | * Engaging and sophisticated * Good level of user interaction * Good information and links |
| **“Cambrian EXPLOSION”** | * Aimed at 16+ age group * Created by Dimitri Bilenkin * Focused on the Cambrian explosion, a period of time when evolution accelerated creating a large diversity of new species in a short period of time * Shows the evolution of locomotion in a variety of creatures using a genetic algorithm | * Very good treatment of mutation and concept of evolution. * Good variety of settings which can be changed to focus simulation * Good use of statistics/data presenting them to the user as a easy to read graph * Simple and easy to use * Little documentation of complex settings * Great use of simulation output (visual representation of evolution) |

The tool “Who Wants to Live a Million Years?” (<http://science.discovery.com/games-and-interactives/charles-darwin-game.htm>) is an example of a free independent learning tool designed to help teach the concept of evolution. It is a flash-based web application aimed at teaching the basic concepts of evolution by demonstration. The content and graphics indicate that it is aimed at pupils from about 12 – 14 years.

The tool contains a learning section (‘Learn about Evolution’) and a simulation section (‘Play the Survival Game’). The learning section shows the user both text and pictures in a fun, animated way to help engage users. The simulation section proceeds as follows:

* It asks the user to choose a starting population and select variations within the population.
* It then demonstrates the impact of several changes in the environment (e.g. a warmer climate, a new predator or a natural disaster) on the numbers of the population - to show how only those best suited to the new environment will survive and reproduce while the others will not.
* If a population is wiped out the user is asked if they want to return to the beginning and select a different set of characteristics to test whether they will improve survivability.
* The simulation takes about 2 minutes to complete each time, with users expected to run several simulations.

A number of criticisms of this tool can be made:

* The simulation gives a misleading representation of natural selection as it presents mutations as a non-random event (as selected by the user) while in reality they are totally random and cannot be chosen.
* It is not clear what the different mutations are - different versions of an animal are shown in crude terms (e.g. one has longer fur, another has a longer body).
* The results of a change in environment are simply presented – there is no explanation why the different versions are increasing or decreasing in number
* The simulation is also not very engaging; there is minimal user interaction (which does not make it fun) with just a few clicks from start to finish.
* Overall, the tool is aimed at young children to provide some ‘fun’ but it provides only a very crude view of evolution.

Another example of a flash-based web application based in flash is “Stop Disasters!” ([http://www.stopdisastersgame.org](http://www.stopdisastersgame.org/)). The focus of “Stop Disasters!” is not evolution but preventing and responding to disasters – it is produced by the United Nations and the International Strategy for Disaster Reduction. It is aimed at older children than “Who Wants to Live a Million Years” and has a game section, a section on high scores from the game, information on the largest disasters in recent history and teacher resources about disasters such as tsunamis.

The simulation tool can be used to teach students about disaster prevention, as part of geography or other lessons. The objective of the game is to assess the risk of a disaster and to reduce the damage to the human population when one does occur. The simulation section proceeds as follows:

* The user selects a scenario e.g. ‘Coastal Village – tsunami’, and difficulty level (which is generated by the level of detail of the maps used).
* A disaster occurs and the user is given a mission e.g. ‘construct accommodation for 400 people with a budget of $35,000).
* The user then spends the budget to purchase various materials.
* A score is shown e.g. number of population successfully housed.
* The simulation takes about 10 minutes to run, depending on the choices made by the user.

“Stop Disasters!” is much more sophisticated and interesting than “Who Wants to Live a Million Years?” as it allows the user to interact with it more fully, it has better graphics and feels more like a game with choices – rather than a few simple steps over which the user has little influence. It does not deal with evolution but is a good example of a sophisticated interactive learning tool.

*Note*: Although “Stop Disasters!” is not specifically relevant to biology it is useful to evaluate as it shows how a simulation can be used as part of the teaching curriculum.

A web based application using JavaScript is “Cambrian EXPLOSION” (<http://www.cambrianexplosion.com/>). The application is focused on showing the evolution of locomotion in creatures. It uses a generic algorithm to generate creatures with different mutations and then runs a simulation of those species measuring how far they move in a specific amount of time. From this it calculates the generations fitness and draws graphs of how the fitness various with each generation thus showing the concept of evolution (small positive mutations each generation add up as that species has a greater change of reproducing thus transferring the mutation onto the next generation). This tool is quite sophisticated and complex although its graphics are not as good as “Stop Disasters!” it makes up for this in its simulation customizability.

In summary, the current tools which have been analysed have a number of advantages and disadvantages and none have been judged to meet all the relevant criteria. While “Who Wants to Live a Million Years?” is focused on evolution it has a number of drawbacks in the user experience and in the level of detail it shows about evolutionary processes. “Stop Disasters!” has a better user experience, being fun and informative and with significant user interaction, but is not focused on evolution. “Cambrian EXPLOSION” is a good simulation of evolution in species however it may be too advanced for the 12-13 year age group and also it focuses only on the evolution of locomotion and not the general evolution of animals.

## Identification of prospective users

Two types of users of the system have been identified – teachers in the classroom and pupils.

Teacher users are teachers of Biology who conduct classes for 12-13 year olds at Kingswood School in which an introduction to evolution is required. Class sizes are from 10-15 pupils. The teacher users have an adequate knowledge of computing and application. Teachers will use the tool to demonstrate the basic concepts of evolution to the pupils, and will act as an administrator for the tool i.e. deciding when it is to be used and setting it for homework.

Pupil users are aged 12-13, are of mixed ability and have already had some background in science. They lack any detailed knowledge of evolution, although they are likely to have heard of it, and may find the idea challenging. The pupil users know how to follow simple instructions on a computer and use simple applications. Pupils will use the tool to see how the concepts of evolution taught to them in the classroom can be illustrated and ‘brought to life’.

## Identification of user needs and acceptable limitations

Analysis of user needs and acceptable limitations has been undertaken for both sets of users – teachers and pupils.

### User needs – teachers

Teachers require a tool that quickly and easily illustrates the concept of evolution in a manner that is understandable and interesting to children and which is easy to administer. Teachers’ more detailed requirements are:

1. As a teaching tool there **should not be a steep learning curve** i.e. the tool should not be over complex in its ideas, as this would defeat the point of simplifying the teaching of the concept.
2. As the tool is to be used as both a teaching tool and a tool to prompt independent learning/further study into the subject area by pupils the tool must be able to be **used within the classroom** (or computer room) as well as **by the pupils in their own time or for homework at home**.
3. The tool needs to be **able to be used by 12-13 year old pupils**, either individually or in small groups of 4-6, and must be **extremely simple to use** with minimum tutoring required – no more than 5 minutes.
4. **Setting up and running a simulation should take no more than 5 minutes** per simulation by pupils.
5. **Total time for using the tool** – from instruction to the analysis of results – **should be about 22 minutes.**
6. The tool must be **portrayed in a way to best engage this age group,** together with any linking content suitable for this age group. The tool must be fun and engaging.
7. All pupils in the classroom and outside the classroom should have **equal access to the tool** and there should be **no minimum system requirements** (e.g. owning a smart phone), so that everyone can use it.
8. When first introduced, **the tool must be demonstrated by a teacher through a projector** so that the pupils can get the most out of the tool and so that it is used correctly and all features used to their maximum effectively.

### User needs – pupils

1. The tool must be **suitable for 12-13 year olds** in terms of the complexity of content and the ease of use.
2. It must be **fun and engaging** – that is, pupils should enjoy each simulation and want to run several.
3. It must be capable of **communicating clear learnings**.
4. It must be capable of being **accessed by all pupils** both inside and outside the classroom.

### Software

1. The tool must **work on the major web browsers**:
   1. Internet explorer 9 and upwards (current school version).
   2. Firefox version 20 and upwards.
   3. Chrome version 22 and upwards.
2. The tool must be able to **run on Windows 7** which is the current school operating system.

### Hardware

1. The **tool must** **run smoothly** on the computers provided in the ICT rooms.
   1. Most computers are fitted with: Windows 7; Intel(R) Core(TM)2 Duo CPU E8400 @ 3.00GHz; 2.00 GB Memory. *Note*: Smoothly means with no noticeable judder (≥15fps) with standard use of the tool.
2. Only a mouse and keyboards can be used as input devices for the system as these are all that are provided by every work station.
3. The tool should be able to **scale up to resolutions larger then 800x600** so that it can be displayed through a school projector onto an interactive whiteboard.

### Simulation

1. The tool should **not be expected to be completely accurate** and approximations may be made so not to increase the complexity of the simulation beyond the scope of the teaching.
2. The simulation should not be expected to model realistic ant movement it should only be expected to **model a 2d representation of ant movement**.
3. The simulation should not be expected to model a realistic environment.
   1. However should **simulate basic regenerating food sources**.
   2. Should only **simulate a 2d terrain**.
4. The simulation should not simulate the complete life cycle of ants i.e. ants are born fully grown; it should only **model the ant’s adult life**.
5. The simulation should not model the inner workings of the nest only the **concept of the nest** is where the Queen lives and new ants are born and where food is stored.

## Data sources and destinations

The pupils will input values through buttons and sliders into the simulation via the user interface. The values selected will be used to change how the simulation runs and will therefore require heavy processing (running through the simulation). The simulation will output the result in an animation onto the screen.

## Data volumes

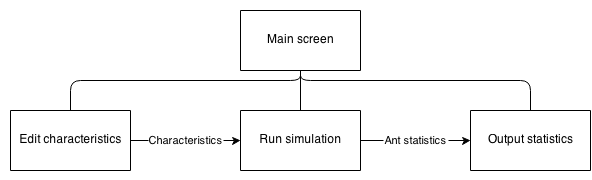
No data is saved as all data is processed in real time and outputted data is shown to the user immediately. So the size of the program is the only data volume, this will be less than 1mb.

## Analysis Data Dictionary

* **Characteristic** – A variable feature of an ant which changes it in some way e.g. how fast it can move.
* **Species** – A collection of unique characteristics define an new species in the simulation
* **Mutation** – A random event which alters a characteristic
* **Ant** **nest (or just nest)** – The home of the Queen ant and where it reproduces as well as a general food store
* **Queen** **ant** – The head of the species and the only ant capable of reproducing
* **Worker Ant** – An ant whose purpose is to collect food and supply it to the nest
* **Soldier Ant** – An ant whose purpose is to guard the nest and other ants from attack. And also to attack ants of a different species.
* **Pheromones** – Chemicals deposited by ants when they move which can be detected by antennae’s and used to alert other ants of danger or as a trial to food. Pheromones concentration slowly degrades over time due to evaporation.
* **Trial pheromone** – Deposited by worker ants and used as trials to and from food.
* **Fitness** – The ability to survive and reproduce
* **Food** – A source of nutrition which is in constant demand by ants. Without a constant supply of food an ant will die. Food will be needed by the nest to create new ants.

## Structure and Data Flow Diagrams

### Structure diagram



## Entity-relationship (E-R) model

This is not applicable to my project as I do not have any database manipulation.

## Object analysis diagram

## Objectives for the proposed system

### Ant simulation objectives

Make these SMART objectives

* Specific – Objectives should be specific to the problem being solved and should specify what should be achieved
* Measureable – You need to be able to measure whether you are meeting the objectives or not?
* Achievable – Are the objectives you set achievable and attainable?
* Realistic – Can you realistically achieve the objectives with the resources you have?
* Time – When do you want to achieve the set of objectives?

1. A tool which **illustrates the concept of evolution**. This concept is summarised in Appendix B. The main ideas of the concept of evolution should all be covered:
   1. **Organisms have characteristics** which determine their species.
   2. As a result of the mutation of characteristics, **variations capable of being inherited** exist within populations of organisms.
   3. Organisms **produce more offspring than can survive**.
   4. These **offspring**, with their different inheritances, **vary in their ability to survive and reproduce**.
   5. In conditions with competition between organisms for survival and reproduction those **organisms with traits that give them an advantage over their competitors pass these advantageous traits on**, while traits that do not confer an advantage are not passed on to the next generation. As a result we have the ‘survival of the fittest’ and a gradual change in populations – they change or may even die out.
2. The simulation should be able to generate **random food placement**.
3. The simulation should be able to simulate basic **growth of food**.
4. The tool should provide a **simulation involving movement** since evolution involves dynamic change and a static tool is not appropriate.
5. The simulation must simulate the three basic types of ants
   1. The **worker ant** – Locates and collects food
   2. The **soldier ant** – Guards and attacks ant species
   3. The **queen ant** – Creates a nest
6. The simulation must **model a basic nest** capable of:
   1. Producing new ants
   2. Being attacked and eventually destroyed.
7. The ants should have variable basic characteristics Including:
   1. **Speed** the ants move
   2. The **reproduction rate**
   3. The **amount of food ants are able to carry**
   4. The ants **eyesight** i.e. the distance the ant can see
   5. The ants **antenna size** i.e. the distance the ant can smell pheromones from
8. The simulation should be able to **introduce random mutations** (so pupils can see how the ants must suited to the environment will survive).
9. The simulation must be able to **show multiple different species at the same time**, so that their fitness can be compared by the pupils.
10. The simulation should model **energy intake from food** needed for ant to survive depending on the ants characteristics. For examples ants with more favourable characteristics such as moving faster should require much more food than ants which have less favourable characteristics.
11. The simulation must be able to model **ants fighting**.
12. The simulation must model **pheromone trails** including:
    1. Their **creation** when an ant is moving
    2. Their **evaporation** due to conditions
    3. **How ants respond** to the trials
       1. Following to find food
       2. Following back to nest

### Simulation interface features objectives

1. **Pause and play buttons** to pause and play simulation.
2. **Navigation buttons** to move around the simulation.
3. **Zooming** in and out of the simulation.
4. For each species (displayed when selecting nest) statistics measuring:
   1. The **amount of food stored in the nest** of a specific species
   2. **Number of ants** in species
5. **Editable characteristics** e.g. by sliders or buttons so that the user can change the simulation in real time.
6. **Selection of different species** so that their characteristics can be compared.

## Appraisal of potential solutions

### Smart phone app

A smart phone app written in Java (android) or Objective-C (IOS) would run the simulation on a phone.

This would fulfil the majority of the requirements:

* Due to smart phone portability the app could be used both inside and outside the classroom.
* Phones are internet-enabled and so the app can be downloaded from within the class room.
* The age group are familiar with the concept of apps and would be familiar with the interface.
* Smart phones are powerful enough to run a complex biological simulation and could maintain simulation accuracy.

However, there are some disadvantages from a smart phone solution. Not all pupils would have a smart phone and as a result not everyone in the class would be able to use the app. Furthermore, due to the fragmentation of smart phone operating systems the same app would need to be written for multiple different architectures (IOS, android …) - possibly making the project unviable. Lastly, if the exercise is done in class there would be no way to know if the pupils where using the tool or using their phones for another use.

### Web application

A web application written in HTML5 and JavaScript is another potential solution. Using the web has a number of advantages.

* The internet is available on school and home computers.
* Access to the tool would not be a problem as almost everyone has access to a computer (i.e. school computer lab) thus for filling the users requirements.
* Even young pupils are very familiar with web application interfaces and as such teachers would only be required to provide a short introduction and pupils could quickly learn how to use the application.
* There would be little issue with cross compatibility as both html and JavaScript are web standards and implemented in all modern browsers including mobile browsers.
* It would be very easy to setup as it is a essentially a webpage and therefore there is no need to download or install any applications just visit the webpage.

However the complexity of the project may increase due to the use of multiple languages (HTML, CSS, and JavaScript) and the added complication of configuring and maintaining a web server and domain. Furthermore a lot of time would have to be spent optimising the application to speed it up as JavaScript is much slower as it runs through a browser compared with natively written programs.

### Desktop application

A desktop application written in C/C++ or Java. This would have a number of advantages:

* The simulation could easily run due to C/C++’s speed and also access to a desktop.
* Like the web application in (HTML5, JavaScript) it could be used by all of the pupils at home or in the classroom for filling the objectives as C/C++ can be compiled to run on multiple different operating systems.

Although as the application would have to be cross compatible between the major operating systems (Windows, Mac and Linux) in order to make sure everyone could use it on their machine increasing the complexity of the program. Furthermore the increased setup time of the tool would make it less user friendly (i.e. user must download and install the program).

### Pen and paper simulation

A non-technical solution to the problem would be to create a set of rules and then use counters on a gridded board to simulate how the ant’s behaviour would change depending on their input values. Random mutations could be introduced by using dice.

* The simulation could be used all of the pupils.
* The simulation does not need computers and therefore the students do not have to change room to one equipped with computers, increasing the amount of time available.

However the simulation would be far too slow and the pupils may not be able to see the overall patterns emerging from it and therefore not learn the concepts of evolution. It would require more than one pupil to be fun and engaging and so cannot be done at home for homework. It would be tedious to record all of the statistics (a computer based solution could do this automatically). The scale of the simulation would be dramatically reduced to fit into the time allowed and would be less impressive as a result making the simulation less interesting to pupils.

## Proposed solution

Detailed proposed solution + justification of using JavaScript + html + css

### Simulation

An ant simulation which models three basic types of ants:

* **Worker ants** – responsible for searching, collecting and depositing food.
* **Queen ants** – responsible for locating a new nest location.
* **Soldier ants** – responsible for defending and attacking other species.

It will also model a simple nest which has the ability to create new ants in a probabilistic manner.

Food must also be modelled. Food will be collected and eaten by ants to keep them alive. Nests will use food to create new nests i.e. a nest will require a certain amount of food to create a new ant.

Pheromone trials, pheromone trials will also be modelled. These will be deposited by ants when they are returning to the nest with food. Pheromones will influence the direction of other worker ants of the same species.

### Interface

The interface will be a single page web application designed in HTML and formatted in CSS. To display the simulation the HTML5 canvas will be used (introduced in the HTML5 specification). The application will be coded in JavaScript; this will not only allow manipulation of the canvas element but also allow the code to edit items on the page. This means data and information can be put onto the page with relative ease compared with other web scripting languages.

## Justification of chosen solution

The chosen solution is a website based tool written in HTML5, CSS and JavaScript and using a school webserver backend to serve the website. This was chosen as the solution because it allows for all of the objectives to be achieved to their full with the least additional complexity.

* All simulation objectives can be accomplished with a well written optimised program in JavaScript.
* The simulation can be easily displayed in an aesthetically pleasing manner with CSS to engage the pupils.
* The tool requires very little preparation time as it only requires the browser which is already installed on school computers.
* The tool can be easily used at home by pupils as it is a website and can be navigated too from anywhere with internet and a web browser (which is assumed they have).
* The tool can be easily displayed through a projector and should be able to scale to any sized monitor (as it is a website).
* The tool will have a shallow learning curve as pupils already know how to navigate webpages.
* The tool will be able to run on existing hardware and software

# Section 2: Design

Diagrams needed:

* Overall system flow chart (website + simulation + scores + login)
* Ant behaviour diagrams (same as ant algorithm )
  + General behaviour
  + Solider behaviour
  + Worker behaviour
  + Queen behaviour
* User interface design
  + UI for webpages (login, main)
  + UI for simulation
* Class/object diagrams
  + OO diagram for ants (Queen?)
  + OO diagram for map/environment

## Overall system design

### The simulation

The simulation will simulate the principles of survival of the fittest by using ants. It will both simulate ant society structure as well as a basic environment for the ants to move in.

#### Evolution

Ants in the simulation will have a set of characteristics; these characteristics define the ant’s behaviour. For example one characteristic of an ant is its eye sight, if a particular ant has very good eye sight, it can spot food from further away, thus being more able to survive. The characteristics implemented will be:

* **Speed –** The speed the ant can move
* **Antenna size –** The size of the ants antenna (determines how far away the ant can smell pheromones)
* **Antenna angle –** The angle through which the ant can detect pheromones from its current direction
* **Eye sight –** The range at which the ant can see objects
* **Eye angle –** The angle through which the ant can see objects from its current direction
* **Jaw size –** The size of the jaw of the ant (determines how much food an ant can carry)
* **Jaw strength –** The strength of the ants jaw (determines how much damage it can do when biting other ants)
* **Sting size –** The size of the ants string (determines how far the ant can attack)
* **Pheromone concentration –** The concentration of pheromones secreted by the ant

A species is a collection of specific values of characteristics. There will be multiple species in the simulation. Ants will all belong to a species, and it is the species which determines the ant’s characteristics. When ants are born they will inherit the species which their nest has. However when a queen is born, there is a chance that the species the queen inherits is mutated. A mutation is the change of a single value of a characteristic in a species. If a queen inherits the mutated species and if it goes onto creating a nest then all ants born from this nest, will inherit the mutation.

A mutation can either be beneficial or destructive e.g. a better eyesight is beneficial as it allows ants to see further, while a decreased speed is destructive as it means that the ant will move slower. Ants which inherit mutations which are beneficial are more likely to be successful in surviving, thus they will have more ants. This means that the species will be passed onto more generations. This is the basic process of evolution and how it will be implemented in my simulation.

To decrease the advantage of some mutations there is a cost associated with characteristics. If a characteristic is more beneficial it will cost the nest a lot more *energy* (health in the simulation) to create an ant.

#### Ant behaviour

All ants must eat a certain amount of food otherwise they will lose health. Ants will have different amounts of health depending on their characteristics, if health is below 0 the ant dies.

There will be three types of ant:

* **Worker ant** – Responsible for finding, collecting and bringing food back to the nest.
* **Soldier ant** – Responsible for attacking other ant colonies destroying their nest or killing ants from the other nest to disrupt their collection of food leading to the collapse of the nest because of starvation. And also responsible for defending ants from attack from other species and also defending the nest.
* **Queen ant** – Creates a nest, and reproduces creating new ants.

**Trial pheromones** are continually laid down by worker ants; they will slowly diffuse into surrounding areas and will also slowly evaporate in the environment. When an ant finds food it will try to make its way back to the nest, as it does so it will lay down pheromones. These pheromones will then be followed by other ants either to the nest or to the food. Ants which have followed the trail and found food will lay down their own pheromones. Each time more pheromones will be laid down making the path more concentrated. Ants will be more likely to follow very concentrated routes when looking for food. Soldier ants will act as sentries and stand along busy trials defending their worker ants.

### Interface

At the start of the simulation the user will adjust the characteristics of the first species or choose random characteristics. Once the simulation is running the user will be able to edit ant’s characteristics or leave the starting characteristics and watch how the simulation plays out.

During the simulation information about the species in the simulation added to the page to allow the user to review the information and adjust characteristics or judge how well the species is doing. The information presented is:

* **Colour** – This will be useful as it will show the user the colour of the species so that they can be easily spotted on the simulation.
* **Number of ants** – The number of ants in the species. This is clearly useful as it shows the user how successful the species is.
* **Number of nests** – The number of nests in the species. This is useful as it gives a guide to how likely the species is to grow as more nests will generally mean more ants.
* **Total amount of food in nests** – This Is the combined amount of food held by the nests in the species. This again is useful as it gives a guide at the rate at which the species can expand. More food will mean more ants and a more successful species.

The user will be able to adjust characteristics and see them update live onto the simulation (i.e. without a restart). This is a useful for the user as it allows them to explore how each characteristic affects ants in the simulation. And so it will be useful when the user first uses the simulation to quickly allow them to understand all of the characteristics. As the user adjusts the characteristics they will get instant feedback in the form of text which shows the species cost as well as the cost of each ant in terms of food. The user will also be able to do the control the characteristics as a whole:

* **Update** – Push the modified characteristics to the simulation, this is needed rather than an instant update because it acts as a buffer. If a user accidently edits a characteristic it should not be pushed to the simulation.
* **Default values** – The update all characteristics to default values, needed if the user has changed characteristics and has not been able to create a viable ant. The default values will be designed to create a species which is initially successful this a good demonstration to prospective users.
* **Random –** This will set all characteristics to random values, useful to the user to show how even extreme mutations can be successful.

The user will be able to zoom in and out of the simulation as well as pan around the entire simulation using mouse gestures e.g. dragging and scrolling. It will also be possible to control the simulations time. The user will be able to:

* **Run/Pause** – Start and stop the simulation. This is clearly useful as it allows the user to observe the entire simulation in a paused state.
* **Step** – Go a single frame into the future, this is will be useful if the user wants to view an event which happens quickly e.g. an ant being chased by a soldier ant.
* **Restart** – Restart the simulation if it doesn’t go to their liking or if all ants die off.

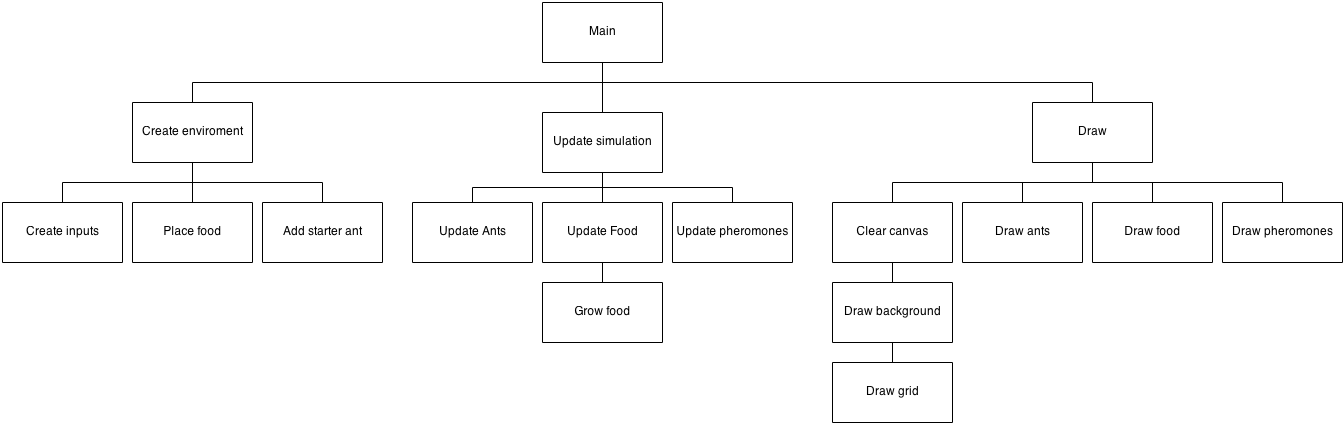
## Description of modular structure of system

The following is analysis of entities which will be in the simulation. Their expected behaviour, actions and properties.

*Note*: Classes defined may change due to unforeseen design decisions

*Note*: **Number** type is any real number

### Overall system



### Ant

The properties of a basic ant are the following:

* **Position** – The location of the ant
* **Direction** – The direction the ant is facing
* **Size** – The size of the ant
* **Species** – The species the ant belongs to
* **Nest** – The nest where the ant was born
* **Health** – The current amount of health the ant has
* **Health Threshold** – The level of health bellow which the ant is hungry
* **Health rate** – The rate at which an ant will lose health
* **Goal** – The current goal the ant is trying to complete e.g. find food
* **Target** – The target the ant is heading to e.g. a piece of food

An ant can also perform the following basic actions:

* **Take food –** Takes a single piece of food
* **Get food –** Head towards a piece of food
* **Find food target –** Finds a locate a target piece of food within viewing distance
* **Scan –** Scan surroundings for items of interest which can be seen e.g. other ants or food
* **Smell –** Smell the surrounding area for pheromones
* **Secrete –** Secrete pheromones
* **Walk –** Walk around either randomly or following pheromone trials
* **Die**

The following is the proposed class designed from the analysis of a basic ant.

|  |  |
| --- | --- |
| **Property** | **Purpose** |
| size [[number, number]] | The width and height of the ant |
| coord [[number, number]] | The x and y coordinate of the ant |
| direction [number] | The direction the ant is facing |
| id [integer] | The unique identifier of the ant, if it is required to be looked up |
| species [Species object] | The species the ant belongs to |
| type [integer] | The type of ant e.g. worker |
| nest [Nest object] | The home nest the ant was born from |
| colour [string] | The colour the ant will appear |
| health [number] | The amount of health the ant has |
| hungerThreshold [number] | The threshold bellow which the ant will be hungry |
| healthRate [number] | The rate at which the ants health decreases each tick |
| alive [boolean] | Determine if the ant is alive or not |
| goal [integer] | The current goal the ant is trying to complete e.g. get food |
| target [[number, number]] | The target coordinate of an item of interest e.g. a piece of food |
| itemsInView [Array] | An array of items in view i.e. food and other ants |
| pheromonesInRange [Array] | An array of pheromones in range |
| prioritizedDirection [number] | The direction the ant wants to move in |

|  |  |
| --- | --- |
| **Method** | **Purpose** |
| addToMap | Add the ant to the map. |
| removeFromMap | Remove the ant from the map. |
| isHungry | Determine if the ant is hungry or not. |
| takeFood | Take a single piece of food from the cell the ant is standing on. |
| atNest | Determine if the ant is standing on top of the nest. This will be used to determine when a worker ant can drop food or when a soldier is near the nest. |
| seeNest | Determine if the ant can see the nest. This will be used by worker ants so they know which direction to head in to get to the nest. It will also be used by soldier ants when guarding the perimeter of the nest. |
| findFoodTarget | Pick a piece of food to target. |
| getFood | Move towards a piece of food and stand on top of it until it has been completely used. |
| useFood | Decide how to use a piece of food e.g. eat the piece of food or carry it. |
| scan | Scan visible surroundings for items of interest, use this information to populate itemsInView. |
| smell | Similar to scan however only detects pheromones in range, populates pheromonesInRange. |
| secrete | Secrete a single pheromone onto the cell the ant is currently standing on top of. |
| wonder | Decide how to move the ant e.g. follow a pheromone trial or wonder randomly. |
| move | Move the ant a certain distance in the distance the ant is facing. |
| die | Kill the ant and remove it from the simulation. |



### Worker

A worker ant is a specific type of ant, its purpose is to find, collect and deposit food to the nest. Because of this the worker ant must be able to perform the following actions:

* **Determine if it can carry food –** Does the ant have enough room to carry extra food
* **Deposit food –** Navigate to the nest and drop food at the nest

The worker class:

*Note*: inherits from Ant

|  |  |
| --- | --- |
| **Properties** | **Description** |
| Carrying [integer] | The amount of food the ant is carrying. |
| carringThreshold [integer] | The threshold above which the ant will return to the nest and deposit food, if it cannot see any more. |

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| canCarry | N/A | Determines if an ant can carry food or not. |
| depositeFood | N/A | Navigates towards the nest, stands on top of it and drops food. |
| dropFood | N/A | Drops a single piece of food |
| useFood | N/A | Determines the best use for food i.e. eat the food or carry the food. |
| doTask | N/A | Performs the actions required to complete the current goal. |
| updateGoal | N/A | Determines the current goal of the ant. |
| updateHealth | N/A | Updates the ant’s health. |
| draw | N/A | Draws the ant |
| update | N/A | Runs through a complete update of the ant. Also determine if ant has died or not. |

### Soldier

The soldier ant is a type of ant; it is responsible for defending ants from its own species and attacking other species. The ant therefore is required to:

* **Determine if there are soldiers in view** – This is used so that the soldier ants can spread out i.e. change guarding position if they can see another ant so that they spread out.
* **Determine if there is food in the view** – This is used for when guarding food, so they know where to guard.
* **Pick a target** – Pick an ant in view to attack.
* **Follow other ants** – Follow the ants they are attacking
* **Attack** – Attack the ants they have targeted i.e. reduce the ants health
* **Guard the nest** – Guard the ants home nest
* **Guard pheromone trials** – Guard the pheromone trials, this is done as they are used by worker ants, and therefore by guarding them they are guarding a large number of worker ants.
* **Guard Food** – As food is the primary resource in the simulation, guarding it is an important feature to help one species dominate.

The solidier class:

*Note*: inherits from Ant

|  |  |
| --- | --- |
| **Properties** | **Description** |
| targetAnt [Ant object] | The ant the soldier is targeting. |

|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Description** |
| nearSoldiers | N/A | Determin if the ant is near another friendly soldier ant or not |
| pickTarget | N/A | Pick a target ant from the ants in current view. |
| follow | N/A | Follow an ant which is in view. |
| attak | N/A | Attack an ant i.e. reduce its health. |
| updateHeatlh | N/A | Update the ant’s health. |
| guardNest | N/A | Control ants logic when guarding the nest. Ant should stand still somewhere close to the nest which where it is not in view of other friendly solidier ants. |
| guardPheromones | N/A | Controls ants logic when guarding pheromones. Ant should walk normally around, following friendly pheromone trials if it finds them. |
| guardFood | N/A | Controls ants logic when guarding a food source. Ant should first walk around until it finds food, then ant should stop near food. |
| doTask | N/A | Perform the actions required to complete the current goal. If the ant has the following goals, perform these actions.   * Guard Nest – guardnest * Guard pheromones – guardPheromones * Guard food – guardFood * Attack – follow ant and attack if close |
| updateGoal | N/A | Determine the current goal of the ant.   * If no goal – Give ant a random goal * If guard nest – guard nest * If guard pheromones – guard pheromones * If guard food – guard food |
| draw | N/A | Draw the ant |
| update | N/A | Run through a complete update cycle of the ant. This includes:   * Updating the ants health * Scan and smell surroundings * Pick a target * Do task and update goal * Move ant and add it to the map |

### Queen

A queen ant is another specific type of ant. The Queen ant is responsible for creating new nests and therefore needs to be able to:

* **Pick nest site**
* **Create a nest**

The queen class:

*Note*: inherits from Ant

|  |  |  |
| --- | --- | --- |
| **Procedure** | **Parameters** | **Description** |
| doTask | N/A | Decide what actions need to be done to achieve the current goal. |
| updateGoal | N/A | Determine the current goal of the ant. |
| pickDirection | N/A | Pick the direction the ant will go in, to make the nest. |
| createNest | N/A | Create a new nest in the ant’s current location, and kill the queen ant. |
| draw | N/A | Draw the ant |
| update | N/A | Run through the ants update cycle.   * Update ants health * Determine if has died or not * If haven’t reached nest then continue moving * Else * Create nest and die. |

### Species

A species is a collection of specific values of characteristics. A characteristic is a property of an ant e.g. an ant’s eye sight (how far the ant can see). The following basic characteristics will be implemented:

* **Speed –** The speed the ant can move
* **Antenna size –** The size of the ants antenna (determines how far away the ant can smell pheromones)
* **Antenna angle –** The angle through which the ant can detect pheromones from its current direction
* **Eye sight –** The range at which the ant can see objects
* **Eye angle –** The angle through which the ant can see objects from its current direction
* **Jaw size –** The size of the jaw of the ant (determines how much food an ant can carry)
* **Jaw strength –** The strength of the ants jaw (determines how much damage it can do when biting other ants)
* **Sting size –** The size of the ants string (determines how far the ant can attack)
* **Pheromone concentration –** The concentration of pheromones secreted by the ant

There will be a chance that a species will be mutated, this will result in the change in the value of a characteristic. This can happen whenever an ant is created. However the mutation will only ever be passed on by queen ants, as normal ants cannot create new ants and therefore cannot pass on the mutation.

|  |  |
| --- | --- |
| **Properties** | **Description** |
| pheromoneConcentration [number] | The concentration of pheromones produced by the ant. |
| reproductionRate [number] | The rate at which new ants are reproduced. Used by nest to determine when to create new ants. |
| colour [string] | The colour of the ants and nests in the species. |
| id [integer] | The unique identifier of the ant. |
| ants [Array of Ant objects] | The array of ant’s belonging to the species. |
| nests [Array of Nest objects] | The array of nests belonging to the species. |
| chars.speed [number] | The value of the speed characteristic. |
| chars.anteenaSize [speed] | The value of the antenna size characteristic. |
| chars.jawStrength [number] | The value of the jaw strength characteristic. |
| chars.jawSize [number] | The value of the jaw size characteristic. |
| chars.stingSize [number] | The value of the sting size characteristic. |
| chars.eyeSight [integer] | The value of the eye sight characteristic. |
| chars.eyeAngle [number] | The value of the eye angle characteristic. |
| chars.antennaAngle [number] | The value of the antenna angle characteristic. |

### Nest

The nest implemented will be a very simple abstraction of actual ant nests. The nest must have the ability to recreate new ants. The nest must also be able to store/receive food from workers.

|  |  |
| --- | --- |
| **Properties** | **Description** |
| Size [number, number] | The size of the nest in pixels. |
| Coord [number, number] | The coordinate of the nest on the map. |
| Id [integer] | The unique identifier of the nest. |
| Species [Species object] | The species the nest belongs to, used when the nest creates new ants. These new ants will inherit the species of the nest. |
| Health [number] | The health the nest has. |
| hungerThreshold [number] | The threshold bellow which the ant is hungry and tries to save food e.g. does not create new ants below this level. |
| healthRate [number] | The rate at which the ants health decreases. |
| Alive [Boolean] | Determines if the ant is alive or not. |

|  |  |  |
| --- | --- | --- |
| **Procedure** | **Parameters** | **Description** |
| reproduce | N/A | Determine which ant to create using probabilities. |
| createAnt | Type [integer] – The type of ant which will be created | Create a new ant of a particular type; add it to the species ants list. |
| die | N/A | Create a new nest in the ant’s current location, and kill the queen ant. |
| draw | N/A | Draw the nest onto the screen. |
| update | N/A | Run through the ants update cycle.   1. Update the nests health 2. If the ant is still alive 3. If the nest will reproduce or not |

### Pheromone

A pheromone is a chemical signal deposited by ants to influence the direction of other ants i.e. lead other ants to food. A pheromone has the following properties:

* **Concentration –** The concentration of the pheromone
* **Position –** The position of the pheromone
* **Species –** The species of the ant which deposited the pheromone. This is important as in some species of ants ants will only follow pheromones laid down by an ant of their own species.

Evaporation in of pheromones will also be implemented; this means that over time, pheromones will slowly reduce in concentration as some of the chemicals are evaporated away.

|  |  |
| --- | --- |
| **Properties** | **Description** |
| Size [number, number] | The size of the pheromone in pixels. |
| Coord [number, number] | The coordinate of the pheromone. |
| Concentration [number] | The concentration of the pheromone. |
| Species [Species object] | The species the pheromone belongs to i.e. the species of the ant which laid the pheromone |

|  |  |  |
| --- | --- | --- |
| **Procedure** | **Parameters** | **Description** |
| draw | N/A | Draw the pheromone onto the screen. |
| update | N/A | Run through the ants update cycle.   1. Reduce the concentration of the pheromone depending on the evaporation rate of the system. 2. If the concentration of the pheromone is below 0 remove the pheromone. |

### Food

A piece of food has the following properties:

* **Position** – The location of the piece of food
* **Size** – The size of the piece of food
* **Density** – The density/concentration of the food

|  |  |
| --- | --- |
| **Properties** | **Description** |
| Size [number, number] | The size of the piece of food in pixels. |
| Coord [number, number] | The coordinate of the piece of food on the map. |
| amount [number] | The amount/concentration the piece of food contains |

|  |  |  |
| --- | --- | --- |
| **Procedure** | **Parameters** | **Description** |
| draw | N/A | Draw the piece of food onto the screen. |
| grow | N/A | Grow the piece of food by a certain amount depending on the environment. |

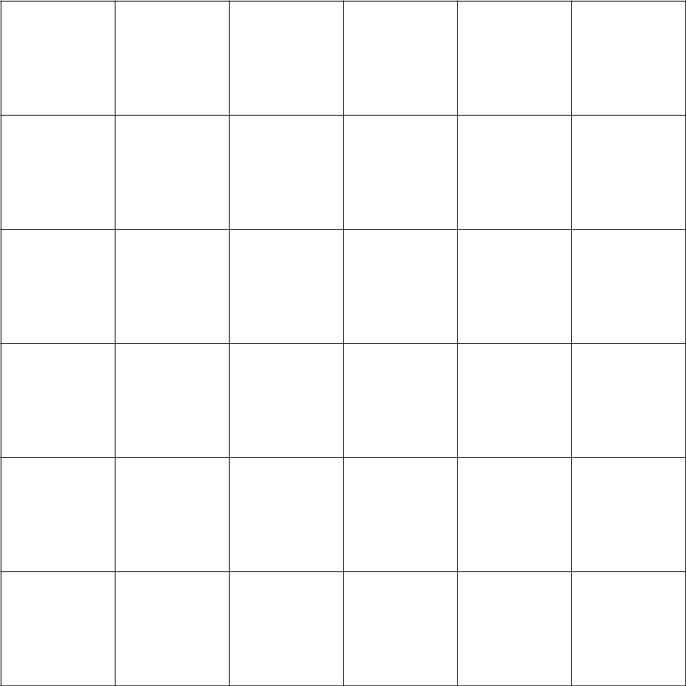
### Map

The map will model a 2d representation of a dirt based environment. Only the following entities will appear on the map:

* **Ants**
* **Food**
* **Pheromones**
* **Nests**

The map which will be implemented will be grid based. It will be made up by a fixed quantity of cells in a rectangular formation. Cells will be a fixed size. Ants, food, pheromones and nests will only be able to exist in a single cell i.e. one ant cannot be in two cells at the same time.

Height



Width

A Cell

Although the map will appear to be 2d and flat, in fact it will be in fact a torus. This will simply mean that if an ant walks of the top of the map, they will appear to walk in from the bottom (also true for the left and right sides of the map). This is done so that the map will be able to appear larger is it is not limited by walls. It also more closely models the natural environment of ants i.e. not in a confined space.

### Configuration

The configuration options which will be available in the program. This includes definitions of the environment in which the ants will be living in.

* **Food heath ratio** – The ratio of which one piece of food translates into so much health.
* **Food amount** – The amount of food which will be in the map.
* **Food grow amount** – The amount of food which grows each time.
* **Food grow rate** – The rate at which food grows.
* **Pheromone evaporation rate** – The rate at which pheromones evaporate in a single tick.
* **Max pheromone concentration** – The maximum concentration a pheromone can have.

Characteristics in the simulation will not only have a value but a number of other properties including:

* **Minimum value** – The minimum value the characteristics value can be.
* **Maximum value** – The maximum value the characteristics value can be.
* **Type** – This is the type which will be accepted as input i.e. an integer or a number or a string.
* **Description** – This is the description of the characteristic. This will be used for displaying the characteristic in the user interface.
* **Health modifier** – This is the amount at which a single increase in value will cost the species in terms of health.
* **Default value** – This will be used when the simulation is first loaded.

## Definition of data requirements (Design Data Dictionary)

|  |  |
| --- | --- |
| **Identifier** | **Description** |
| Characteristic | A collection of values relating to a property of an ant. This includes:   * **Value** – The value of the characteristic, type depending on the characteristic * **Maximum value** – The maximum value the value can have * **Minimum value** – The minimum value the value can have * **Type** – The type the value is e.g. integer or number * **Description** – A text description of the characteristic will be used in the user interface for hover text. * **Health modifier** – How the characteristics value alters the species health cost * **Default value** – The default value of the characteristic, used when first opened. |
| Species | A collection of specific values of characteristics. |
| Mutation | An alteration in the value of a single characteristics value to a random value. |
| Cell | A single square in a grid which can contain a single piece of food as well as many ants and many pheromones. |
| Map | A list of all cells in the map.  *Note*: An index is the position of a specific cell in the map. Indices can be converted to and from coordinates using the algorithm stated in the algorithms section |
| Worker ant | The ant responsible for searching, collecting and depositing food back to the nest. |
| Soldier ant | The ant responsible for attacking and defending ants. |
| Queen ant | The ant responsible for founding a nest. |
| Nest | Responsible for reproducing new ants and collecting food left by workers. |
| Food | An item on the map which is collected by ants and converted Into health to be used by ants to continue to live or by nests for creating new ants. |
| Pheromone | A chemical released by ants when they are finding their way back to their nest. |
| Goal | The aim of an ant. Different ants can have different goals depending on their purpose. The goals available will be:   * **None** – This goal will be used when ants are first created, they will be assigned a default task when they first run updateGoal in their update loop. * **Find food** – This goal is used by all ants, it will cause the tasks needed to find food in the map * **Get food** – This goal is used by all ants, it will cause the tasks needed to collect a piece of food * **Drop food** – this goal is used by only worker ants, it will cause the tasks required to drop a single piece of food off at the nest. * **Go to nest site** – This goal is used by the queen ant to head to the new nest site. * **Create nest** – A goal used by the queen ant to cause the tasks required to create a new nest. * **Guard nest** – A goal used by soldier ants to cause tasks for guarding the nest. * **Guard pheromone** – Used by the soldier ant to cause the tasks associated with guarding pheromone trials. * **Guard food** – Used by the soldier ants to cause task for finding and searching for a food source. * **Attack** – A goal used by soldier ants to perform the actions required to attack another targeted ant. |
| Task | A task is the specific action which will be required to be performed. Many tasks make up a single goal. An example of a task is picking up food, which will pick up a single piece of food. |
| Target | A target can be a piece of food, an ant or a nest. The target will be used differently depending on the current goal. The target represents what the ant is heading towards. |
| Direction | The direction the ant is currently facing |
| Prioritized direction | The direction the ant wants to move in |
| Size | The size of the object in pixels |
| Coord | A Cartesian coordinate |
| Health | Health will be a quantity used by ants to live. Ants start with a particular amount of health. This amount slowly decreases until at 0 the ant dies, the **healthRate** is the amount of health lost each tick. Health is restored by eating food. A single piece of food can be converted into a particular amount of health. The ratio of this is set by the **healthRatio** variable. The **hungerThreshold** is the amount of health bellow which the ant or nest is deemed to be hungry. |
| Ant type | This is the type of ant e.g. a worker, soldier or a queen. |
| Id | A unique identifier, needed to compare if two objects are the same or not. |
| Tick | A single loop of the entire program. Many ticks may occur each second. |
| Species cost | The cost of the species i.e. the sum of all health modifiers applied to each characteristics value in the species. |
| Food cost | The amount of food an ant is given when it is born. Part of this cost goes to the species cost, the left over is the amount of health the ant starts with. This will be different for each type of ant. |
| Queen steps | The minimum and maximum number of ticks the queen will move for when locating a new nest |
| Worker, soldier, queen probability | The probabilities of each type of ant being created. The probabilities are relative to each other i.e. they will be normalised. |
| Environment | The properties of the habitat the simulation is running in. Properties include:   * **Food heath ratio** – The ratio of which one piece of food translates into so much health. * **Food amount** – The amount of food which will be in the map. * **Food grow amount** – The amount of food which grows each time. * **Food grow rate** – The rate at which food grows. * **Pheromone evaporation rate** – The rate at which pheromones evaporate in a single tick. * **Max pheromone concentration** – The maximum concentration a pheromone can have. |
| wrapped coordinate | A coordinate which has been corrected for wrapping (due to the map being a tours). |
| map index | The index for a particular cell in the map. |

## Description of record structure

This is not applicable for my project as there are no databases used.

## File organisation and processing

This is not applicable for my project as there is no data stored or retrieved from files.

## Validation required

Input validation will be done by sliders, this will reduce the amount of validation of input needed as it restricts the user to input values within a specific range and of a specific type. This is done to reduce the number of validation errors the user receives and thus streamlining their use of the application.

However the following characteristics will require extra validation and may produce validation errors or warnings:

* **Queen steps min and Queen steps max** – As these two characteristics act as a range of values it is important for min to the less then max. If this is not the case the user will be given an error and how to correct it.
* **Soldier, Queen and Soldier Food cost**– If an ants food cost is less or equal to the species cost the user should receive a warning as it means that the ant will be created with its health less than or equal to zero. This means that the ant will die instantly when it is created. This is likely not what the user wants to do and so a warning is issued (This is not an error however and the simulation can be run with ants with health less than zero).

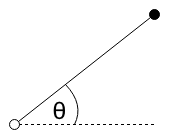
## Identification of storage media

This is not applicable for my project as there is no data stored.

## Identification of suitable algorithms for data transformation, pseudo code of these algorithms

### Angle between two points

Find the angle from the horizontal in radians between two coordinates. This will be used by ants to target pieces of food or targeting other ants. It is needed so that the ants will know which direction to head in to get to a particular coordinate.



coord

target

**DECLARE** integer dx

**DECLARE** integer dy

**SET** dx = targetX - coordX

**SET** dy = targetY - coordY

**RETURN** atan2(dy, dx) + Math.PI / 2;

*Note*: atan2 is a built in function to calculate the arctan corrected for the sector the angle lies in.

### Boundary

The map will act like a torus i.e. if an ant goes off one side of the map, it will reappear on the other side of the map. This pseudo code will update a coordinate to its *wrapped* coordinate if needed.

|  |  |  |
| --- | --- | --- |
| **Identifier** | **Type** | **Description** |
| coordX, coordY | number | The coordinate of the entity. |
| boundsXMin, boundsXMax | number | The minimum coordinate and maximum coordinate which the map covers in the X direction. |
| boundsYMin, boundsYMax | number | The minimum coordinate and maximum coordinate which the map covers in the Y direction. |

**IF** coordX < boundsXMmin **THEN**

**SET** coordX = boundsXMax - abs(coordX)

**ELSE IF** coordX >= boundsXMax **THEN**

**SET** coordX = coordX - boundsXMax

**ENDIF**

**IF** coordY < boundsYMin **THEN**

**SET** coordY = boundsYMax - abs(coordY)

**ELSE IF** coordY >= boundsYMax **THEN**

**SET** coordY = coordY – boundsYMax

**ENDIF**

**RETURN [coordX, coordY]**

*Note:* abs is a built-in function which returns the absolute value of a number.

### Get a block of cells

It will be necessary to sample blocks of cells by ants to see what lies around them. This algorithm will return an array of all the cells which lie around the ant.

**DECLARE** block = []

**FOR** y = (coordY – sizeHeight) **to** (coordY + sizeHeight) **DO**

**FOR** x = (coordX – sizeWidth) to (coordX + sizeWidth) **DO**

block.push([coordX, coordY])

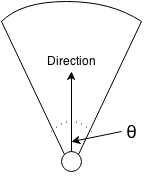
**ENDFOR**

**ENDFOR**

**RETURN** block

### Get a sector

Get sector will return the list of cells which lie within the sector at a particular coordinate facing a certain direction with an angle θ. This will be used by ants to sample the map in front of them to see any items of interests such as other ants, food and pheromones. Ants will have a certain viewing angle (the angle through which they can see).



**DECLARE** block = [];

**FOR** y = (coord.y – radius) **TO** (coord.y + radius) **DO**

**FOR** x = (coord.x – radius) **TO** (coord.x + radius) **DO**

**SET** searchCoord = {

x: x,

y: y

}

**SET** angle = angleTo(coord, searchCoord)

**SET** dist = distance(coord, searchCoord)

**SET** minSector = validateDirection(direction - angle / 2)

**SET** maxSector = validateDirection(direction + angle / 2)

**IF** angle >= minSector **AND** angle <= maxSector **AND** dist <= radius) **THEN**

block.push(searchCoord

**ELSE** **IF** (direction <= angle / 2 **OR** direction >= Math.PI \* 2 - angle / 2) **AND** (angle <= maxSector **OR** angle >= minSector) **AND** dist <= radius **THEN**

block.push(searchCoord)

**ENDIF**

**ENDFOR**

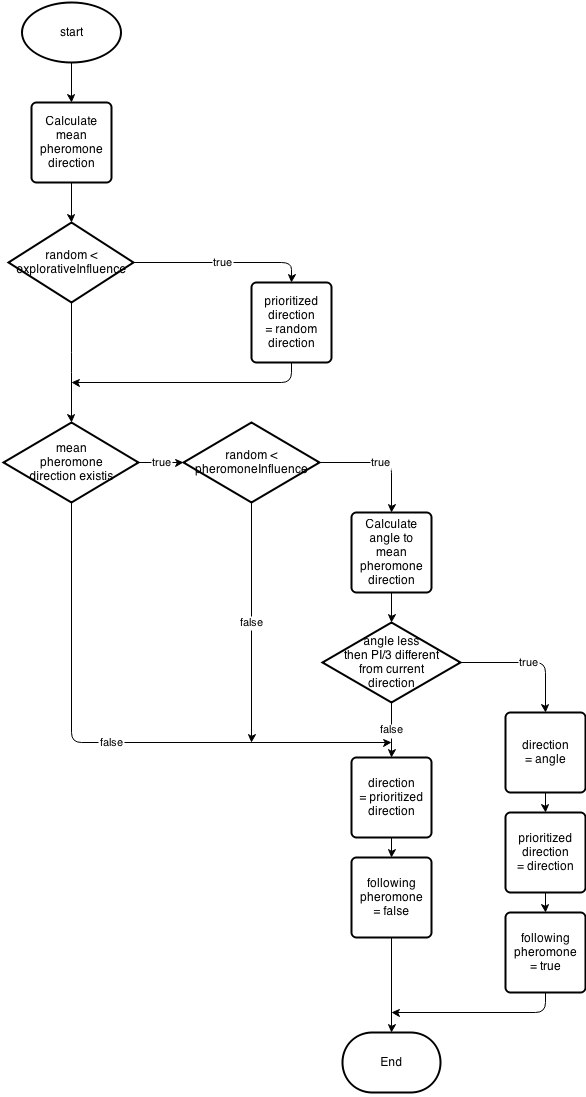
**ENDFOR**

**RETURN** block

### Ant non goal driven movement

A task which all ants will do is non goal driven movement. This means wondering around the map without a specific target. This is done when ants are searching for food, or when soldiers are looking for their nest or guarding pheromone trials. Ants will walk randomly however their direction will also be influenced by pheromones they come across. The higher the concentration of pheromones the more likely the ants will follow are particular direction.

The prioritised direction is used as a long term direction goal. This goal could last from 5 to 100 ticks. It is the direction the ant will tend towards and can change randomly. The explorativeInfluence is chance that the prioritised direction will change. The purpose of the prioritised direction is to create straight paths (rather than ant’s constantly jerking around as they change direction randomly).



### Ant pheromone secretion

The following algorithm is used to add a pheromone to the map. If there is already a pheromone of the same species on the map it will add to the concentration of that pheromones, this means that pheromones will get stronger if ants secreting pheromones walk over them. This is a key feature of the simulation as it allows trials to form to food.

**FOR** pheromone in cell **DO**

**IF** pheromoneSpecies == thisSpecies **THEN**

pheromoneConcentration += thisPheromoneConcentration

**IF** pheromoneConcentration > MAX\_PHEROMONE\_CONCENTRATION **THEN**

pheromoneConcentration = MAX\_PHEROMONE\_CONCENTRATION

**ENDIF**

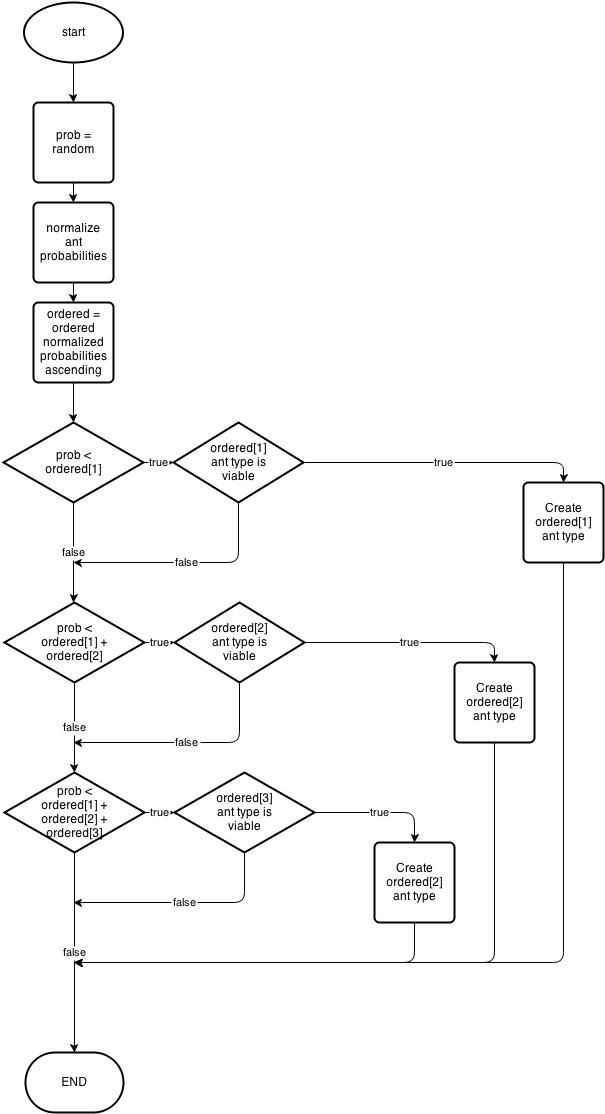
**ENDIF**

**ENDFOR**

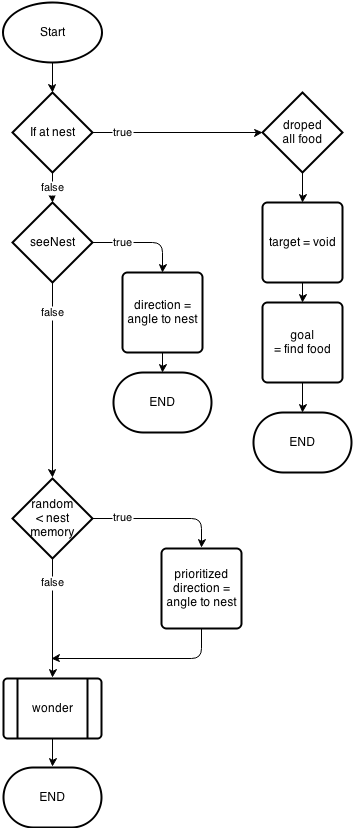
pheromone = **new** Pheromone(this.species.chars.pheromoneConcentration, this.coord)

### Nest reproduction

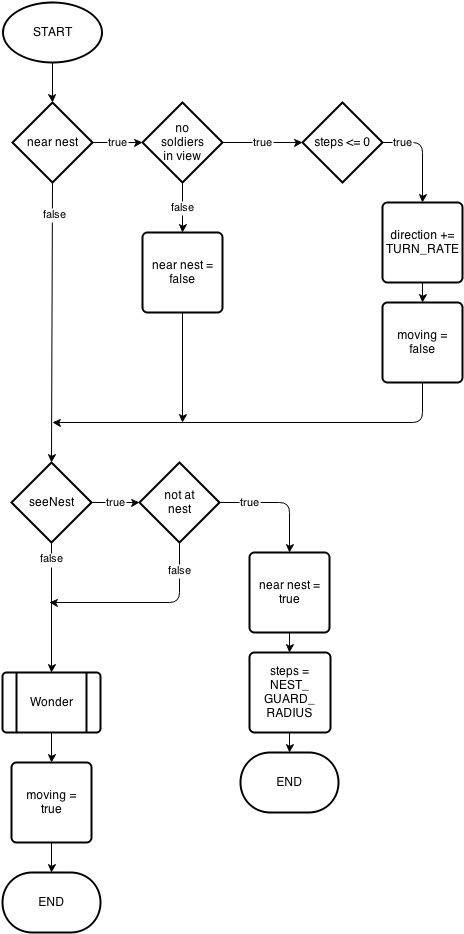
The following algorithm will be used by the nest to decide which ant will be created. It uses a probabilistic approach in that it is more likely that some ants will be created than others.



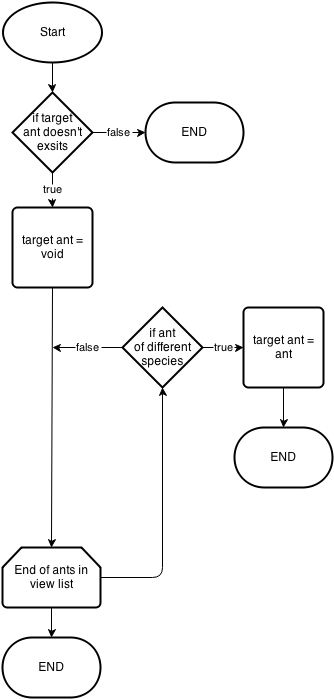
### Worker ant depositing food



### Soldier nest guarding



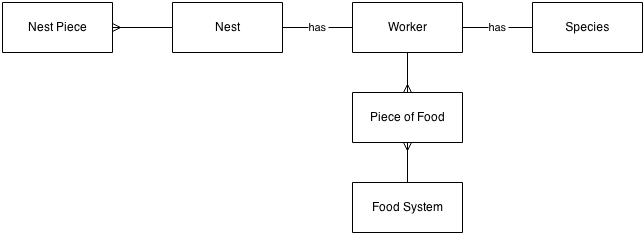
### Soldier pick target



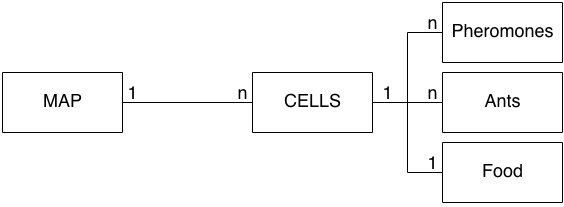
This algorithm will be used by soldier ants to pick a target ant to attack.

## Class definitions (diagrams) and details of object behaviours and methods

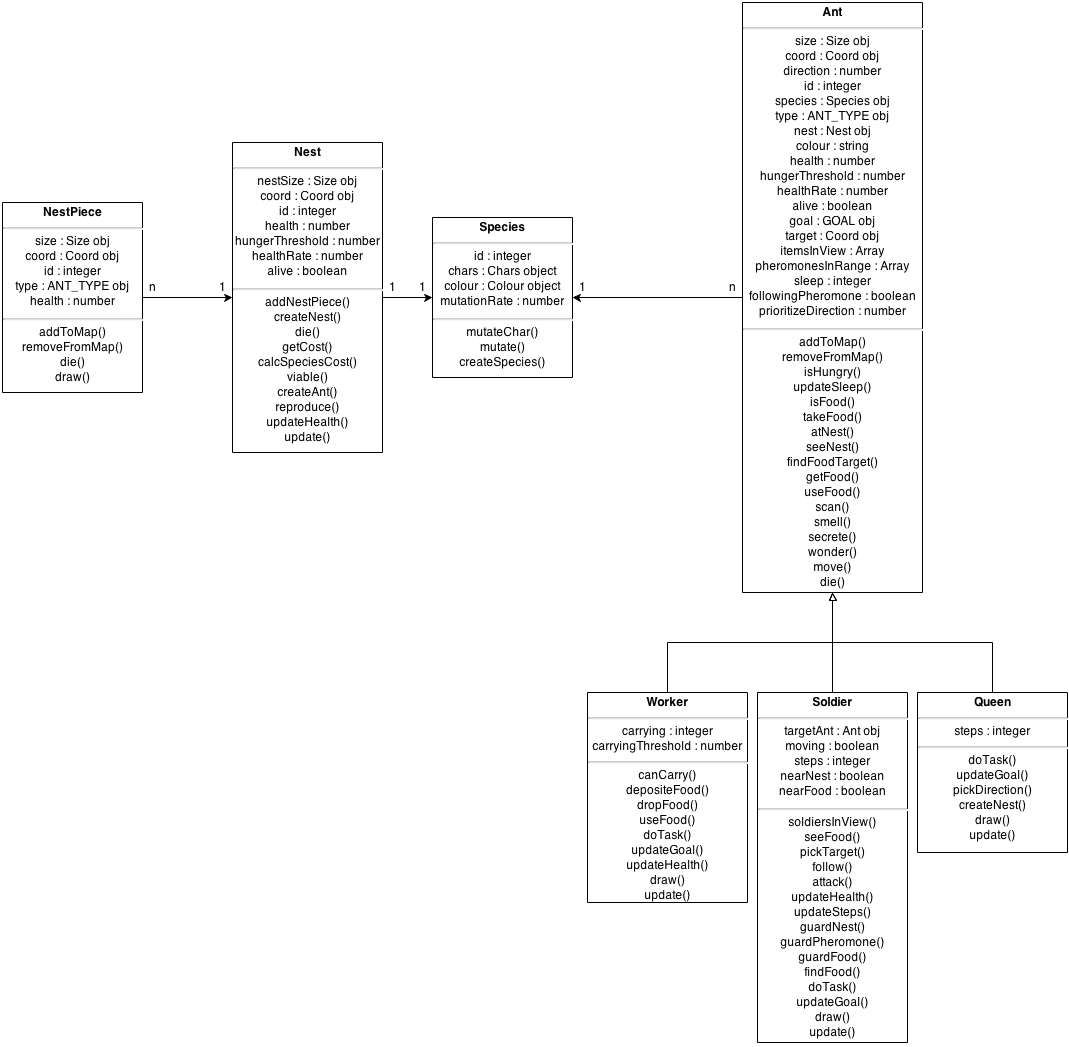
### entity relationship - antAnt object behaviour

The diagram above shows the relationship between an ant and other objects in the simulation. This is true for all ants however the worker ant is slightly different. A worker ant can also hold multiple pieces of food.

### Map object behaviour

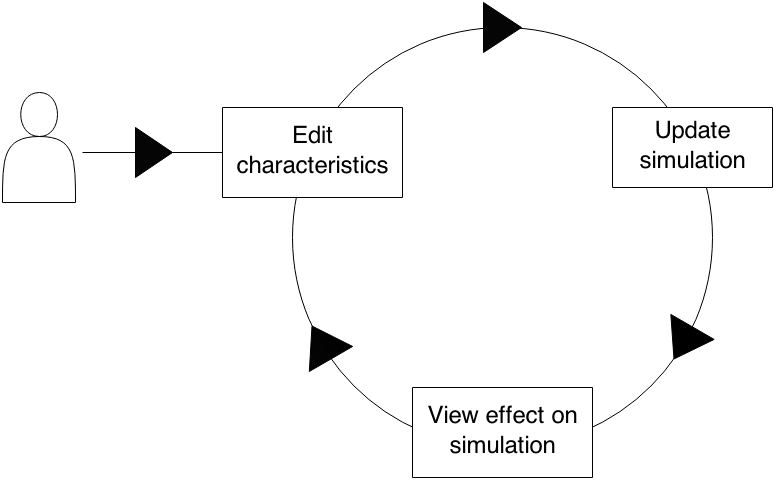


### Classes

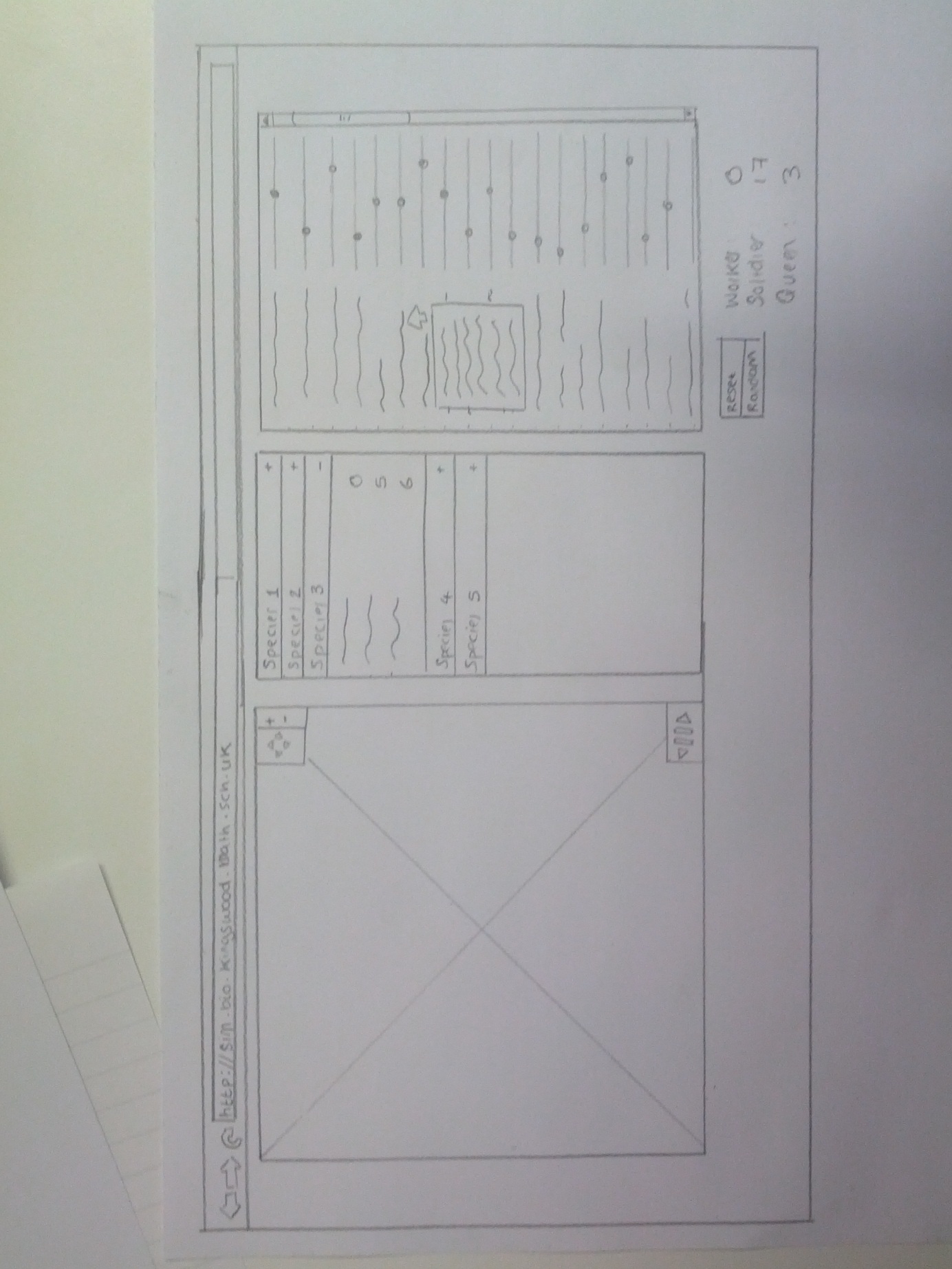


## User interface design (HCI) rational

### User interaction

This is the expected use of the simulation by the user. The interface will be designed to be allow the user to perform the following process as easily as possible:

### Interface Mock-ups



**6**

**5**

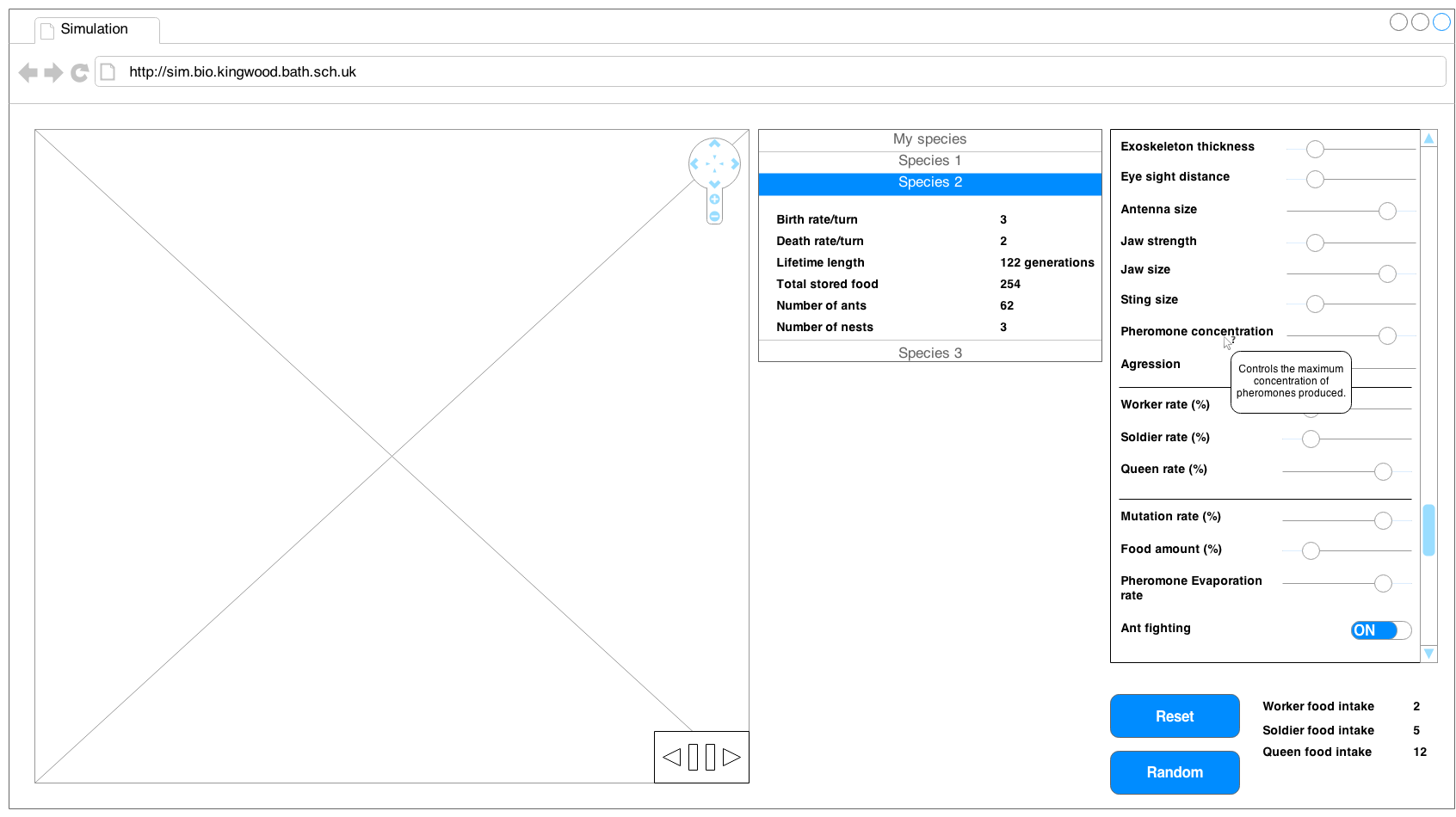
**4**

**3**



**2**

**1**



The interface is split into three main sections. Each section has a single purpose, this will make the layout of the system easy to learn and also will make it more intuitive. The positing of the panels is such that it mimics the users flow through the program. The user will observe the simulation, then look at the data produced by the simulation and finally edit the characteristics in the simulation i.e. flowing left to right across each panel.

#### The simulation panel

The simulation panel is located on the far left of the screen. It will be the first thing the user sees when they start the application. The simulation will start running so that it is obvious that this is the simulation and also provide as a point of interest for the user.

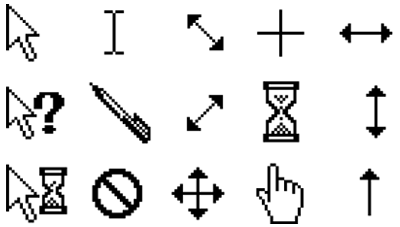
The main item in the simulation panel is the simulation itself. The design of the simulation is stated in the UI sample of planned valid output designs. However there are also two other main items in the panel. These are both controls for the simulation:

* **Time controls** – Located in the bottom right of the simulation. These will be used to control how time is displayed in the simulation. It will have three buttons: A pause/play button and on either side a step forward and step backward a single tick. The icons used will be intuitive to the user of their function because of their universal use. The colours used will be neutral in comparison with the bright colours of the simulation. This is because the controls should not attract attention away from the simulation. The size of these controls is small compared with the size of the simulation. The reason for this is because the controls are on top of the simulation and big controls would reduce the amount of viewable simulation.
* **Navigation controls** – Located in the top right of the simulation. These controls will allow the user to both zoom in and out as well as pan around the simulation. Again the icons chosen are popular in many other applications and synonymous with map navigation. Again these controls will have neutral colours. The position of these controls will be in line with the time controls. The reason for this is so that the user will not have to move their mouse too far to use the controls and also so the user will not have to move their mouse across the simulation.

The user will not only be able to navigate in time using the buttons provided, they will also be able to use a number of shortcuts which will allow them to perform these actions much quicker than hitting a button. The shortcuts will be:

* **Spacebar** – Pause/play the simulation. The spacebar is used as it is the biggest button on the keyboard and it very accessible to the user. Also it is used in a number of other applications as the pause button and so the user will already be familiar with the concept of it being the pause button.
* **“f” key** – The “f” key would step forward a single tick in the simulation. “f” is chosen as it is the first letter of “forward” and therefore will be memorable.
* **“b” key** – The “b” key would step backwards a single tick in the simulation. “b” will be used as it is the first letter of “backwards” and so it will be easily memorable for the user. What’s more, the “b” and “f” keys are in close proximity on the keyboard and so it will be easy for the user to press either of these keys.

Navigation will also be able to be done without the use of pressing buttons on the screen. The advantage of this is that it will be much easier and faster for the user to perform these actions. The user will use the arrow keys to navigate. The reason this was chosen is that many games and mapping programs use the arrow keys to pan around a map. The plus and minis (“+” and “-”) keys will be used as a shortcut for zooming in and out. Again the use of these keys is common practice for zooming in many other applications and so the user will intuitively be able to use this function.

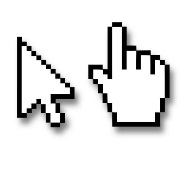
Finally the mouse will be another way to navigate around the map. Holding the moues down on a spot and dragging in a direction will be used to pan around the map. And scrolling forward and backward will be used for zooming. To make it obvious that the user can use the mouse to interact with the simulation the cursor of the mouse will change when it is move over the simulation from the default pointer to a navigation cursor.

#### The data panel

The data panel is located in the centre of the screen. The data panel will be used to display information about the species in the simulation. It will be automatically updated as the simulation runs. To make the information more compact, the data about each species will be expectable and contractible. The colour of each species in the table will be the same colour as it appears in the simulation. This will be done so that the user can easily spot which ants correspond to which species. And therefore speeding up their fluency with which they use the simulation.

The information displayed about each species will be:

* **Birth rate/turn** – This is the number of ants born each turn for the species.
* **Death rate/turn** – This is the number of ants which die each turn.
* **Lifetime length** – This is the time in number of ticks that the species has been around for.
* **Total stored food** – This is the total amount of food in all nests in the species.
* **Number of ants** – This is the number of ants in the species.
* **Number of nests** – This is the number of nests in the species.

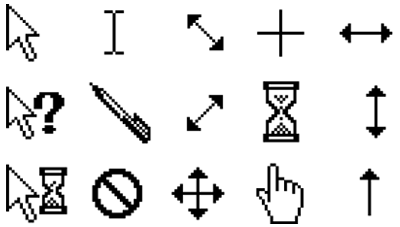
A single species will be able to be selected by clicking on it in the table. To symbolise a selected species it will be coloured a blue colour which is synonymous with highlighted items in most web browsers. This will help the user to understand what happens when they click on a species from the table. What’s more the user’s cursor will change to a pointer rather than a default cursor when the user hovers over a species. This will make it obvious to the user that the item is intractable and will change when clicked. The reason the user will know this is because the pointer cursor is used when you hover over a hypertext link on a webpage.

When a species is selected, the simulation will centre on a nest of that species in the simulation. Also the characteristics of that species will appear in the configuration panel. This allows the user to easily see more information about the species i.e. the location of its nests in the simulation and its characteristics.

#### The configuration panel

The configuration panel is located on the far right of the screen. It contains the characteristics of the selected species and allows them to be edited by using sliders. Sliders are used as they force type and range validation on the input. This is extremely useful as it decreases the amount of errors and warnings which the user will have to see (Errors and warnings are bad as they disrupt the users flow and may cause announce).

The characteristics will be arranged in a logical order i.e. group related characteristics together. This will allow the user to more easily find characteristics quickly and explore how similar characteristics affect the simulation.

As the impacts of the characteristics are not immediately obvious to the user a hover text will be provided i.e. when the user hovers over a characteristic a description of the characteristic will appear. So that the user will use this feature the cursor will change to a help cursor when the user hovers over a characteristic.

Aside from the characteristics, controls for choosing characteristics will also be available. These controls are:

* **Random** – This will generate completely random characteristics. This is useful to the user as it easily allows them to see a wide range of the types of species which may appear in the simulation.
* **Reset** – This will reset all the characteristics of a species to the default values. This is clearly useful as it allows the user to mess around with characteristics to see their effect and then easily reset them all without refreshing the page.

Finally, data about the consequences of the characteristics values picked is displayed at the bottom of the panel. This information displayed will be:

* **Worker food intake** – displays the amount of food it will take to generate a worker ant.
* **Soldier food intake** – displays the amount of food it will take to generate a worker ant.
* **Queen food intake** - displays the amount of food it will take to generate a worker ant.

This is useful to the user as it allows them to instantly see feedback from the change in characteristics i.e. as a change may take some time to be noticeable in the simulation.

#### Colour scheme and fonts

There will only be a single font used in the entire interface. This is to increase consistency in the page. The font used will be Helvetica. Helvetica will be used as it is available on most modern operating systems and also it provides a clean and comfortable look for the page compared with more formal serif fonts like times new roman.

The colours used in the interface will be neutral black and white in each of the panels. The reason for this is to draw attention to the colours in the simulation which should be the main point of focus of the page. However a bright background around the panels may be used to make the page look fun.

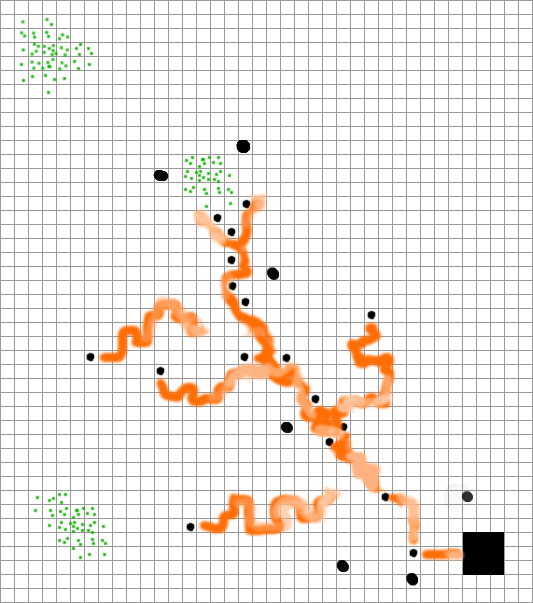
## UI sample of planned data capture and entry designs

Data will be collected when the user edits a characteristic. Each characteristic is listed in the characteristics panel. Ranges are used to enter the data into the simulation, the reasons for this is to reduce the number of validation errors and warnings the user will have to see. This is because range inputs limit the input to a certain type and also between a certain ranges of values.

*Note*: For UI for data capture see Configuration panel in Interface Mock-ups

## UI sample of planned valid output designs





#### Colours

Each species in the simulation will have a unique colour. This colour will be randomly generated. This will mean that the simulation will be interesting as it will have many different colours. All ants, nests and pheromones of the species will have this same colour.

Pheromones in the simulation will be displayed in the colour of the ant species however a much lighter tint, the strength of the pheromone determines the opacity of the colour so if there is lots of pheromones with a higher concentration or newly laid pheromones then the colour will be more vivid showing that is more active and recent while older more stagnant trials will be very light, as to not attract attention from the user.

Food is displayed as green a colour often associated with leaves, grass… which are the food the ants eat in real life. This will be intuitive to the users as many games use the colour green the symbolise food.

#### Shapes

In the simulation a number of different shapes will be used to differentiate between different items. This is needed as ants, nests and pheromones from the same species will be the same colour and so to determine which is which their shape is used.

Worker ants will be a square. A square as opposed to a more lifelike representation of an ant is used as it will decrease clutter on the screen and allow the user to focus more on the behaviour of the ant rather than how it looks. When an ant is carrying food, it will have a smaller green square on its back. This will be useful as it will show to the user which ants are carrying food and so they can more easily monitor these ants behaviour. The square is green as it shows that they are carrying food, which is also green.

Soldier ants will be displayed as squares also. However soldier ants will have a white cross on them. The cross will turn red when they are attacking another ant. Red was chosen as it is often seen as a colour associated with anger and therefore attacking/fighting will be logically assumed. The change in colour of the cross is useful as it tells the user extra information about the current goal of the ant or why they are behaving in a particular way e.g. why are they following another ant.

The queen ant differs from both the soldier ant and the worker ant in its shape. The queen ant will be a circle. This is because queen ants are of particular importance as they have the ability to create a new nest and therefore will be of interest to the user. Furthermore queen ants will be much less likely to be born and so it will be a rare event when one is created. To maximise the chance that the user will be able to spot the queen ant its shape will be different from all of the other ants.

A nest in the simulation will be a large square. This is chosen as due to its size it will be immediately obvious to the user where it is.

Food in the simulation will be displayed as a collection of dots. The number of dots will represents the amount of food in the cell. This is done so that it will be easy for the user to estimate the concatenations of food in particular regions of the map just by glancing at the area.

Pheromones will be displayed as long trials behind ants. This is much like in real life hover the trials are invisible to the naked eye. The trials will be visible so that the user can easily see the areas of high activity by looking for the areas with the most concentrated pheromones. It will also be nice for the user to be able to see the trials of where ants have been moving, and also see what trials are influencing the movement of ants.

## Description of measures planned for security and integrity of data

There is no sensitive personal information recorded or entered into the system so the integrity of data is not applicable to my project.

## Description of measures planned for system security and integrity of data

There will be no

## Overall test strategy

As part of the development strategy methods from test driven development will be used to create parts of the application. This means that for some parts of the project unit tests will be written which will be used to aid development. This will mean that a testing framework will be used during the development of the project to generate and automatically tests these tests.

White box test Ant, worker, queen, soldier and nest classes because methods of these classes will primarily contain logic and so it is important to test each non trivial logical path to insure that the procedures perform correctly. Utilities functions i.e. functions which do not belong to any class but provide useful actions which can be applicable to many situations, will be black box unit tested. This is because these functions will more mathematically oriented rather than logically e.g. the angle between two points.

As the tool will be used outside the classroom i.e. it is set for homework, it is important for the toll to work on multiple platforms. So it will be tested on multiple platforms (Operating systems and browsers) to make sure that it can be used outside the classroom. Due to the large numbers of possible operating systems and web browser combinations it is needed to limit these tests to only the most popular choices.

The simulation will be very intensive for large numbers of ants, so it will be needed to carry out a stress test to check that the simulation will be able to cope on the hardware available in the class room.

User interface functional testing will also be carried out to test that the user interface works as expected. This includes that buttons perform the correct actions and that inputs and outputs are displayed correctly.

Finally it will be necessary to test the simulation against real life biological models. As the simulation will model ants it should be able to be compared to real life ants and produce similar results/patterns. This is needed to show that the simulation truly models ants.

# Section 3: Technical Solution

# Section 4: System Testing

# Section 5: System Maintenance

# Section 6: User Manual

# Section 7: Evaluation

# Appendix A: Availability of Teaching Materials on Evolution

## Online research

The following sources illustrate the types of materials which are available for school teachers on the subject of evolution and related subjects (such as conservation).

|  |  |  |
| --- | --- | --- |
| **Website** | **Materials available** | **Assessment** |
| http://www.arkive.org/education/  (searched 2/10/13) | * Presentations, teacher notes and spreadsheets on numbers of Finch variations in the Galapagos Islands and on peppered moths * Aimed at 14-16 and 16-18 age groups | * Focus on conservation * Attractive materials * No online or computer simulation tools on evolution but some data which can be used in classrooms |
| More to be added |  |  |

## Interviews

Interview with Dr Sheffrin(Head of sciences and Biology teacher) have been conducted at Kingswood School, Bath on 18/10/13. The findings were as follows:

### How is the topic of evolution currently taught?

Taught in year 8, 11 and sixth form biology. Depending on the age group it is covered in various levels of complexity, it is taught over a series of 45 minute lessons of classes of roughly 15 and homework’s. The majority of the work is done using worksheets.

### What topics are covered?

* Mutation
* Natural selection
* Survival of the fittest
* Darwin vs. Lamarckism
* Adaptation
* Habitats and environments
* Characteristics

However for the sixth form topics on DNA and specific process are covered.

### Hardware and software available?

During classes an IT room is available with a computer for all of the pupils. There is little software specifically for use of biology teaching however there is lots of generic software such as web browsers available on all of the machines.

### How much time is available to use the tool?

Half of a lesson and a prep.

### Good example animals which can be modelled and will be interesting to observe?

Animals which have large populations are best to model as having more generations show a speeded up version of evolution which in animals with smaller populations takes a much longer amount of time and is therefore less interesting to simulate. Insects would probably be the best to simulate.Appendix B: Evolution – the main concepts

From Wikipedia – entry on ‘Evolution’

“Evolution by means of [natural selection](http://en.wikipedia.org/wiki/Natural_selection) is the process by which genetic mutations that enhance reproduction become and remain more common in successive generations of a population. It has often been called a "self-evident" mechanism because it necessarily follows from three simple facts:

* Heritable variation exists within populations of organisms.
* Organisms produce more progeny than can survive.
* These offspring vary in their ability to survive and reproduce.

These conditions produce competition between organisms for survival and reproduction. Consequently, organisms with traits that give them an advantage over their competitors pass these advantageous traits on, while traits that do not confer an advantage are not passed on to the next generation.

The central concept of natural selection is the [evolutionary fitness](http://en.wikipedia.org/wiki/Fitness_%28biology%29) of an organism.[[96]](http://en.wikipedia.org/wiki/Evolution#cite_note-Orr-96) Fitness is measured by an organism's ability to survive and reproduce, which determines the size of its genetic contribution to the next generation. However, fitness is not the same as the total number of offspring: instead fitness is indicated by the proportion of subsequent generations that carry an organism's genes.[[97]](http://en.wikipedia.org/wiki/Evolution#cite_note-Haldane-97) For example, if an organism could survive well and reproduce rapidly, but its offspring were all too small and weak to survive, this organism would make little genetic contribution to future generations and would thus have low fitness.

If an allele increases fitness more than the other alleles of that gene, then with each generation this allele will become more common within the population. These traits are said to be "selected *for*". Examples of traits that can increase fitness are enhanced survival and increased [fecundity](http://en.wikipedia.org/wiki/Fecundity). Conversely, the lower fitness caused by having a less beneficial or deleterious allele results in this allele becoming rarer — they are "selected *against*". Importantly, the fitness of an allele is not a fixed characteristic; if the environment changes, previously neutral or harmful traits may become beneficial and previously beneficial traits become harmful. However, even if the direction of selection does reverse in this way, traits that were lost in the past may not re-evolve in an identical form.”

Search undertaken 8/10/2013

# Sources

## Ants and their behaviour

<http://en.wikipedia.org/wiki/Ant>

<http://en.wikipedia.org/wiki/Eusociality>

<http://en.wikipedia.org/wiki/Pheromone>

<http://en.wikipedia.org/wiki/Ant_colony>

<http://en.wikipedia.org/wiki/Leafcutter_ant>