DGT1038Y Coursework Semester 1

Coursework Aims

This coursework allows you to demonstrate that you:

- Understand how to develop a modular Java program with methods (other than main) and manipulate text file contents.
- Are able to take simple pseudo-code descriptions and construct working Java code
- from them.
- Can write a working program to perform a complex task.
- Can write code that it is understandable and conforms to good coding practice.

Specification

The aim of this coursework is to construct a simple simulation of a beehive. The hive will consist of a Queen bee along with a number of other categories of bees. The simulation will demonstrate the viability of the hive given the availability a varying number of flowers for pollen production.

You are not required to have any knowledge of beekeeping in order to complete this coursework and no scientific accuracy is claimed in the representation of the beehive and bees lifecycles presented here.

The simulation may bear some superficial similarities to a real beehive but is grossly simplified and in most cases likely to be quite different to how a real beehive might function.

Your task is to implement the specification as written.

No marks will be awarded for deviating from the specification in order to increase the realism of the beehive and in fact it may well cost you marks.

How the beehive works

For this coursework you are expected to follow the specification of the beehive, bees and flowers as set out below. This will not correspond exactly to a real beehive or flowers in reality but we have chosen particular aspects for you to model **that help you to demonstrate your Java programming**.

There are a number of entities and procedures that contribute to this simulation. For our purposes these include:

A beehive: The hive is the housing unit for all the bees and where they bring in pollen which gets converted to honey for storage. The beehive should be represented through the one and only Java class for this coursework.

Bees: A beehive will consist of only one queen bee, female worker bees, male drone bees. The population of worker and drone bees fluctuate depending on the varying daily egg laying of capability of the queen.

Flowers: Three types of flowers are present in the beehive’s garden, namely: Roses, Frangipani and Hibiscus. Each type of flower has a daily total pollen production rate and at the start of each day, this value is reset to its maximum pollen number. During each visit by a bee, each type of flower will have its corresponding number of pollens that can be collected.

	Roses	**Frangipani**	**Hibiscus**
Pollen Collection during each visit	20	50	10
Total daily pollen produced by this type of flower	20,000	50,000	10,000

Modelling the Beehive

aDayPasses(): This method is called repeatedly from the *main* method to simulate a new day. Within *aDayPasses,* the methods *incrementAge, resetFlowerArray,emptyStomachOfAllBees,metamorphose, feedingTime, undertakerCheck and logDailyStatusToFile* are called.

addEggtoHive(int): This method take the number of eggs as argument, find available slots in the workerBee (2D array) and add them. Eggs which do not get available slots are lost.

AllWorkerBeesGardenSorties(): This method is executed five times daily and for each time, all worker bees are made to visit flowers for pollen collection.

emptyStomachOfAllBees(): This method is called once daily to reset all bees attribute to 0 for they have not yet fed on honey.

feedingTime(): This method is called once daily for all larva and bees to be fed honey.
There is a hierarchical order that must be followed: The queen is first to be fed with 2 units of honey, followed by all the larva with 0.5 unit of honey, all workers with 1 unit of honey, and all drones with 1 unit of honey each. Eggs and Pupa should not be fed as they do not eat during these stages. In case the Queen does not receive her quota of honey on any given day, the whole beehive is doomed and the simulation should be terminated.

funeral(int): This method set all attributes of a specific row to zero (0) given the row's index as argument and keep track of the total number of death that has occurred in the beehive.

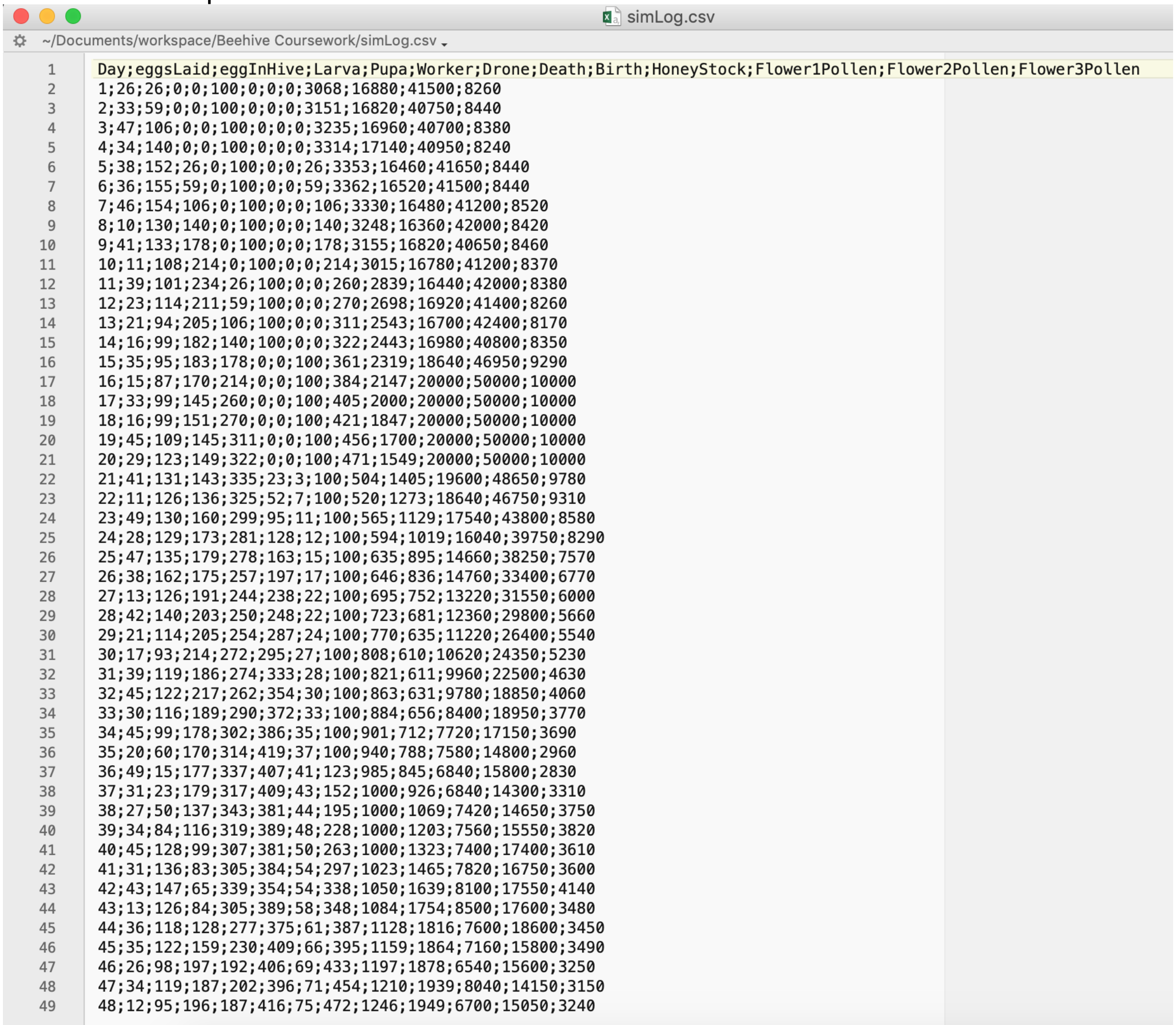
incrementAge(): Everyday this method increment the age of all eggs, larva,pupa, worker and drone bees in the hive. Worker bees have a lifespan of 5 weeks while Drones live till the age of 8 weeks from the time their eggs were laid.

initBeesArray(): This method gets called at the beginning of the simulation only and sets all the attributes of all rows in the 2D array of workerBee to zero (0). Then it checks if the simulation configuration file provided some workers which are then added to the workerBee array.

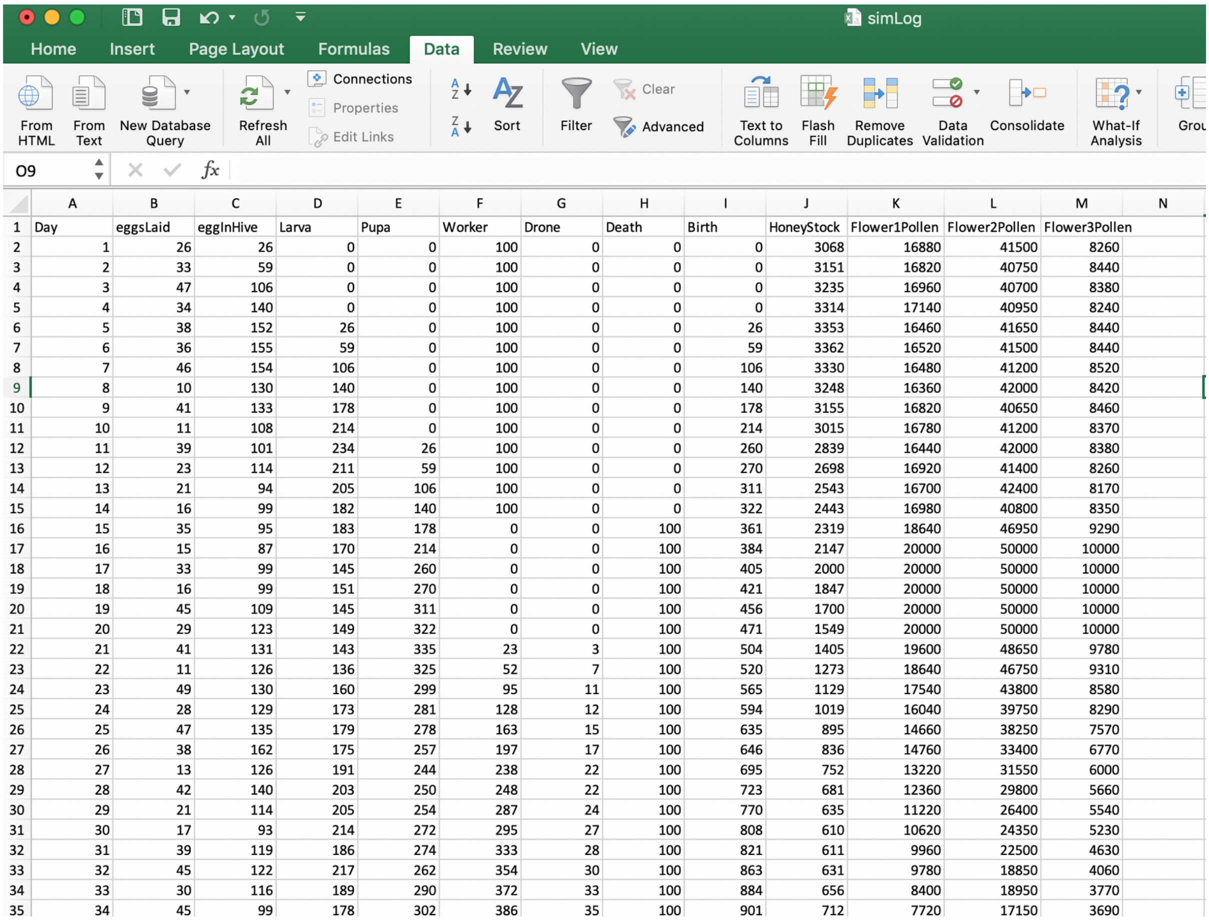
layDailyEggs(): This method gets called daily to determine the random number of eggs to be laid by the Queen on the given day. The range is between 10 and

50 eggs (inclusive). Once obtained, the number of eggs is passed as argument to the method *addEggtoHive().*

logDailyStatusToFile(): This method is called at the end of each day to append the day's data to a logfile named 'simLog.csv'. The file should have headings: *Day;eggsLaid;eggInHive;;Larva;Pupa;Worker;Drone;Death;Birth;HoneyStock;Flower1Pollen;Flower2Pollen;Flower3Pollen* with semi-column in between on the first line. Subsequent lines should consist of the corresponding comma-separated values for each heading. Screenshots of the CSV file opened in text-reading mode and in spreadsheet is shown below:



```
Day;eggsLaid;eggInHive;Larva;Pupa;Worker;Drone;Death;Birth;HoneyStock;Flower1Pollen;Flower2Pollen;Flower3Pollen
1;26;26;0;0;100;0;0;0;3068;16880;41500;8260
2;33;59;0;0;100;0;0;0;3151;16820;40750;8440
3;47;106;0;0;100;0;0;0;3235;16960;40700;8380
4;34;140;0;0;100;0;0;0;3314;17140;40950;8240
5;38;152;26;0;100;0;0;26;3353;16460;41650;8440
6;36;155;59;0;100;0;0;59;3362;16520;41500;8440
7;46;154;106;0;100;0;0;106;3330;16480;41200;8520
8;10;130;140;0;100;0;0;140;3248;16360;42000;8420
9;41;133;178;0;100;0;0;178;3155;16820;40650;8460
10;11;108;214;0;100;0;0;214;3015;16780;41200;8370
11;39;101;234;26;100;0;0;260;2839;16440;42000;8380
12;23;114;211;59;100;0;0;270;2698;16920;41400;8260
13;21;94;205;106;100;0;0;311;2543;16700;42400;8170
14;16;99;182;140;100;0;0;322;2443;16980;40800;8350
15;35;95;183;178;0;0;100;361;2319;18640;46950;9290
16;15;87;170;214;0;0;100;384;2147;20000;50000;10000
17;33;99;145;260;0;0;100;405;2000;20000;50000;10000
18;16;99;151;270;0;0;100;421;1847;20000;50000;10000
19;45;109;145;311;0;0;100;456;1700;20000;50000;10000
20;29;123;149;322;0;0;100;471;1549;20000;50000;10000
21;41;131;143;335;23;3;100;504;1405;19600;48650;9780
22;11;126;136;325;52;7;100;520;1273;18640;46750;9310
23;49;130;160;299;95;11;100;565;1129;17540;43800;8580
24;28;129;173;281;128;12;100;594;1019;16040;39750;8290
25;47;135;179;278;163;15;100;635;895;14660;38250;7570
26;38;162;175;257;197;17;100;646;836;14760;33400;6770
27;13;126;191;244;238;22;100;695;752;13220;31550;6000
28;42;140;203;250;248;22;100;723;681;12360;29800;5660
29;21;114;205;254;287;24;100;770;635;11220;26400;5540
30;17;93;214;272;295;27;100;808;610;10620;24350;5230
31;39;119;186;274;333;28;100;821;611;9960;22500;4630
32;45;122;217;262;354;30;100;863;631;9780;18850;4060
33;30;116;189;290;372;33;100;884;656;8400;18950;3770
34;45;99;178;302;386;35;100;901;712;7720;17150;3690
35;20;60;170;314;419;37;100;940;788;7580;14800;2960
36;49;15;177;337;407;41;123;985;845;6840;15800;2830
37;31;23;179;317;409;43;152;1000;926;6840;14300;3310
38;27;50;137;343;381;44;195;1000;1069;7420;14650;3750
39;34;84;116;319;389;48;228;1000;1203;7560;15550;3820
40;45;128;99;307;381;50;263;1000;1323;7400;17400;3610
41;31;136;83;305;384;54;297;1023;1465;7820;16750;3600
42;43;147;65;339;354;54;338;1050;1639;8100;17550;4140
43;13;126;84;305;389;58;348;1084;1754;8500;17600;3480
44;36;118;128;277;375;61;387;1128;1816;7600;18600;3450
45;35;122;159;230;409;66;395;1159;1864;7160;15800;3490
46;26;98;197;192;406;69;433;1197;1878;6540;15600;3250
47;34;119;187;202;396;71;454;1210;1939;8040;14150;3150
48;12;95;196;187;416;75;472;1246;1949;6700;15050;3240
```



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Day	eggsLaid	eggInHive	Larva	Pupa	Worker	Drone	Death	Birth	HoneyStock	Flower1Pollen	Flower2Pollen	Flower3Pollen	
2	1	26	26	0	0	100	0	0	0	3068	16880	41500	8260	
3	2	33	59	0	0	100	0	0	0	3151	16820	40750	8440	
4	3	47	106	0	0	100	0	0	0	3235	16960	40700	8380	
5	4	34	140	0	0	100	0	0	0	3314	17140	40950	8240	
6	5	38	152	26	0	100	0	0	26	3353	16460	41650	8440	
7	6	36	155	59	0	100	0	0	59	3362	16520	41500	8440	
8	7	46	154	106	0	100	0	0	106	3330	16480	41200	8520	
9	8	10	130	140	0	100	0	0	140	3248	16360	42000	8420	
10	9	41	133	178	0	100	0	0	178	3155	16820	40650	8460	
11	10	11	108	214	0	100	0	0	214	3015	16780	41200	8370	
12	11	39	101	234	26	100	0	0	260	2839	16440	42000	8380	
13	12	23	114	211	59	100	0	0	270	2698	16920	41400	8260	
14	13	21	94	205	106	100	0	0	311	2543	16700	42400	8170	
15	14	16	99	182	140	100	0	0	322	2443	16980	40800	8350	
16	15	35	95	183	178	0	0	100	361	2319	18640	46950	9290	
17	16	15	87	170	214	0	0	100	384	2147	20000	50000	10000	
18	17	33	99	145	260	0	0	100	405	2000	20000	50000	10000	
19	18	16	99	151	270	0	0	100	421	1847	20000	50000	10000	
20	19	45	109	145	311	0	0	100	456	1700	20000	50000	10000	
21	20	29	123	149	322	0	0	100	471	1549	20000	50000	10000	
22	21	41	131	143	335	23	3	100	504	1405	19600	48650	9780	
23	22	11	126	136	325	52	7	100	520	1273	18640	46750	9310	
24	23	49	130	160	299	95	11	100	565	1129	17540	43800	8580	
25	24	28	129	173	281	128	12	100	594	1019	16040	39750	8290	
26	25	47	135	179	278	163	15	100	635	895	14660	38250	7570	
27	26	38	162	175	257	197	17	100	646	836	14760	33400	6770	
28	27	13	126	191	244	238	22	100	695	752	13220	31550	6000	
29	28	42	140	203	250	248	22	100	723	681	12360	29800	5660	
30	29	21	114	205	254	287	24	100	770	635	11220	26400	5540	
31	30	17	93	214	272	295	27	100	808	610	10620	24350	5230	
32	31	39	119	186	274	333	28	100	821	611	9960	22500	4630	
33	32	45	122	217	262	354	30	100	863	631	9780	18850	4060	
34	33	30	116	189	290	372	33	100	884	656	8400	18950	3770	
35	34	45	99	178	302	386	35	100	901	712	7720	17150	3690	

main(): The main method calls out the *readSimulationConfig* method and instantiates the 2D array workerBee with 1000 rows and 6 columns. Most importantly it repeatedly calls out the method aDayPasses for the given number of days that the simulation needs to run or until the Queen dies (whichever comes first). You may find this code snippet useful to add a delay for each day in your simulation:

```
try {
        Thread.sleep(1000);//wait for 1 second
} catch (InterruptedException e) {
        e.printStackTrace();
}
```

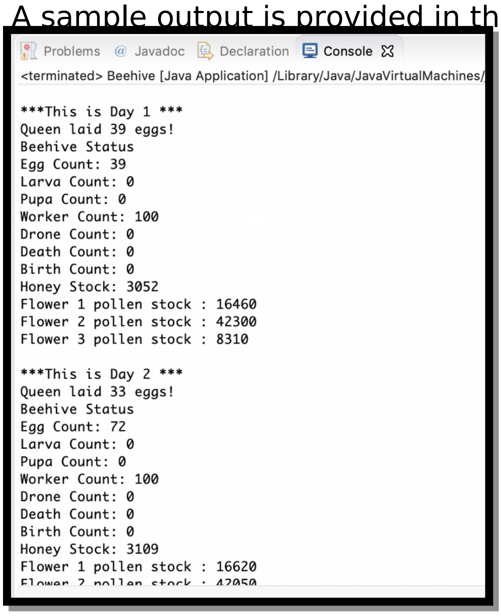
metamorphose(): This method checks each entry in the workerBee 2D array and update their types if required based on their age as detailed in the table below.

Age	Type Conversions
4	Egg hatches into a larva
10	Larva transforms into a pupa
20	Worker/Drone emerges from pupa

At age 20, either a drone or a worker bee can emerge from a pupa and the probability for a drone to emerge is 10% and occurs randomly.

printBeehiveStatus(): This method traverse the 2D array workerBee and count the total number for the different categories of entities (eggs, larva, pupa, worker,drone),total death/birth of bees that occurred in the beehive and current honey in storage.

A sample output is provided in the screenshot below:



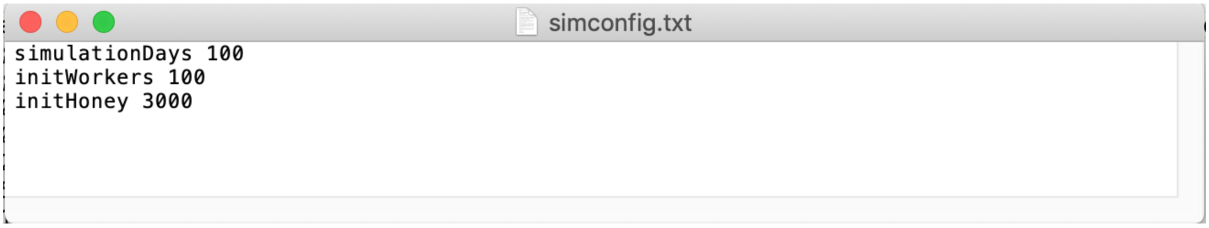
```
<terminated> Beehive [Java Application] /Library/Java/JavaVirtualMachines/

***This is Day 1 ***
Queen laid 39 eggs!
Beehive Status
Egg Count: 39
Larva Count: 0
Pupa Count: 0
Worker Count: 100
Drone Count: 0
Death Count: 0
Birth Count: 0
Honey Stock: 3052
Flower 1 pollen stock : 16460
Flower 2 pollen stock : 42300
Flower 3 pollen stock : 8310

***This is Day 2 ***
Queen laid 33 eggs!
Beehive Status
Egg Count: 72
Larva Count: 0
Pupa Count: 0
Worker Count: 100
Drone Count: 0
Death Count: 0
Birth Count: 0
Honey Stock: 3109
Flower 1 pollen stock : 16620
Flower 2 pollen stock : 42050
```

printFlowerGarden(): This method does the screen printing for the remaining pollen stock for each of the three types of flowers present in the garden. (See Flower 1-3 in screenshot above).

readSimulationConfig(): This methods gets called at the beginning of the simulation to read in some configuration parameters such as number of days to run the simulation, initial number of worker bees and initial honey stock. The format for the configuration file (*simconfig.txt*) is shown in the screen shot below:



```
simulationDays 100
initWorkers 100
initHoney 3000
```

resetFlowerArray(): This method replenish the maximum pollen capacity for each flower type on a daily basis (At the start of each day).

undertakerCheck(): This method is called at the end of each day to check whether larva or bees have fed on that particular day. If not, they die, and the method *funeral()* is called.

visitFlower(): This method enables a bee to select a flower type to visit; Each of the three types of flower has a one third probability of being selected and this is done randomly. If it happens that the visited flower type is out of pollen, the method visitFlower() is recalled a maximum of 5 times to enable the bee to obtain a different flower type with pollen in stock. Once collected, it is assumed that the pollen gets converted and stored as honey in the beehive. For every 40 units of pollen, 1 unit of honey is obtained.

The Bees Lifecycle

Honey bees develop in four distinct life cycle phases: egg, larva, pupa, and adult.
Among the adults, there are three subcategories: A queen, sperm-producing male drones and nonreproductive female workers. For this simulation, it can be assumed that a queen is already present in the beehive and therefore, only drones and workers will emerge from the
pupa. The different data for the bee entities **should** be done using a 2-dimensional array with the following attributes order:

Attribute	beeId	Age	Type (egg=1, larva=2,pupa=3, worker=4,drone=5)	PollenCollectionSorties	Eaten	Alive
Index	0	1	2	3	4	5

The queen bee lays between 10 and 50 eggs daily (fun fact: in reality it is between 1000 and 2000 eggs daily) and the ratio of drone:worker is 10:100.

Exceptions

You should be trying to use exceptions in the construction of your simulator where possible. You should be catching appropriate I/O exceptions but also might consider the use of exceptions to correctly manage:

- The input of a configuration file that does not conform to the specified file format.
- Config file not found
- Attempts to admit too many workers in the config file.
- …

Extensions

You are free to extend your code beyond the basic simulator as described above. You are advised that your extensions should not alter the basic structures and methods described above as marking will be looking to see that you have met the specification as we have described it. If you are in any doubt about the alterations you are making please include your extended version in a separate directory.
Examples of extensions:

- Create a Graphical User Interface (GUI) for the simulation
- Add more data to the config file
- Make the simulation capable of accepting a config file through command line
- Some bees can die randomly due to diseases/poor health
- Use of 'royal jelly'…
- …many more

Marking

Marks will be given for:

- Applications that compile and run
- Applications that meet the specification properly
- Successful completion of one or more extensions
- Good coding style
- Good comments in your source code

Submission

Submit **the Java source file you have written, the simconfig.txt used, the simLog.csv generated file and screenshots of the screen outputs in a word document** in a zip/rar file **BeeHive.zip/Beehive.rar**. Do not submit class files

Also include a text file **runningInstructions.txt** that contains a brief listing of the parts of the coursework you have attempted, describes how to run your code and finally a description of the extensions you have attempted if any.

Submission should be done online via the Google Classroom Platform by midnight of Sunday 12th January 2020.