FIT9131 Assignment B, S2 2023 - Quokka Selfie Quest

FIT9131 Assignment B: Quokka Selfie Quest

(image from: https://www.buzzfeed.com/bradesposito/quokka-pics)

Introduction

This assignment is due by 11:55pm Friday of Week 12 (**Friday 26 May, 11.55pm**). It is worth 35% of the marks for your final assessment in this unit. **Heavy penalties will apply for late submission.** This is an individual assignment and must be your own work. You must attribute the source of any part of your code which you have not written yourself. Please note the section on plagiarism in this document.

In preparing your program please note the following:

- **You must use the workspace environment in the Ed platform to code all parts of your program. You must not copy and paste large sections of code from somewhere else.**
- **You must acknowledge all code in your assignment that you have taken from other sources**.
- **The Java source code for this assignment must be implemented according to the FIT9131 Java Coding Standards.**
- **Only a text interface is to be used for this program**. **More marks will be gained for a program that is easy to follow with clear information/error messages**.

- **In this assessment, you must *not* use generative artificial intelligence (AI) to generate any materials or content in relation to the assessment task.**

Any points needing clarification may be discussed with your tutor in your applied learning class. You should not make any assumptions about the program without consulting your tutor.

Completion of this assignment contributes towards the following FIT9131 learning outcomes:

1. Design and construct Java programs according to standard object-oriented principles
2. Apply and demonstrate debugging processes to Java applications
3. Develop strategies for efficient and effective program testing
4. Document code according to specific programming standards
5. Identify and apply the "object-oriented" concepts of encapsulation, abstraction and polymorphism
6. Explain and apply software engineering principles of maintainability, readability and modularisation

Specification

For this assignment you will write a program to simulate a scheme where quokkas on Rottnest Island in Western Australia offer tourists selfies in return for food. This section specifies the required functionality of the program.

Background

The Quokka is a small marsupial from the south west region of Western Australia. Quokkas are found on the mainland as well as on Rottnest Island near Perth. The quokka is known to the indigenous Noongar people by various names including *ban-gup* and *quak-a*.

Quokkas are naturally curious and approachable. Their friendly nature means that they are very popular with tourists who love taking “quokka selfies".

In recent years the population of quokkas has declined largely due to decreasing habitat caused by bushfires and land clearing.

The quokkas, however, are resourceful. An entrepreneurial group of quokkas on Rottnest Island has decided to ensure their survival by posing for selfies with tourists in exchange for food - the Quokka Selfie Quest. Each day a flock of tourists arrives by ferry to Rottnest Island. The quokkas approach the tourists and offer to pose for selfies. The quokkas are willing to pose with individual or groups of tourists. They are willing to allow photos or videos, or pose for a sketch.

In order to decide how much to charge for their services the quokkas have requested a simulation of the Quokka Selfie Quest enterprise for a period of 30 days. At the end of the simulation they will receive a report indicating whether they can receive sufficient food for survival.

Quokka selfie quest simulation

The Quokka Selfie Quest program will simulate 30 days of food earned from selfies by a group of quokkas. The simulation steps through each of the 30 days. At the end of the simulation the quokka population survival is reported.

Program start up

The program begins by displaying a brief welcome message.

The number of tourist groups for each of the 30 days is read in from a file *tourist.txt*. The file will contain 30 comma separated numbers, with each number representing the number of groups of tourists for one day. The groups vary in size from 1 to 6 tourists. There is no other reading from the file during the running of the program.

The user is then prompted for the number of quokkas in the population. From this data, the program creates a collection of quokkas. Each quokka will have a unique identification code, a flag indicating whether it has a baby in its pouch, food supply and a status indicating whether it is alive or not. The details of these are as follows:

1. The unique identification code will be in the following format: Q*nnn* (where *nnn* represents a sequence of three digits).

2. There is a 20% chance that a quokka will have a baby quokka in its pouch.

3. The initial food supply is 2 bags for each quokka without a baby and 3 bags for each quokka with a baby.

4. All quokkas will be alive at the start of the simulation.

Specific actions each day

1. Set up tourist data:

The number of tourist groups for the day is used to set up a collection of tourist groups. Each tourist group in the collection will have a size, type of selfie and a status indicating whether the selfie has been taken. The details of these are as follows:

- The group size will vary from 1-6. There is a 30% chance of a group of 1 (an individual), 30% for a group of 2 and 10% for each of the other group sizes.
- The selfie type can be a photo, video, or sketch, with 60% photos, 35% videos, and 5% sketches.
    - The length of a video selfie will be a random time from 1 to 60 seconds.
    - The sketch will be either a 5 minute pose or a 10 minute pose. Each are equally likely.
- Sketches are only possible for an individual (group of 1).
- All tourist groups will have a status indicating no selfie has been taken at the start of the day.

2. Quokkas offer tourists selfies:

- The quokkas will take turns to offer the tourists a selfie.
- The order in which the quokkas approach the tourists depends on the size of the food stores that the quokkas have. The quokkas with the lowest supply of food are more desperate and will approach the tourists first.
- If there are not many tourists on a particular day then some quokkas may miss out on a chance to take a selfie.
- If there are a large number of tourists then the quokkas may need to approach the tourists a second time (or more). The quokkas will continue offering selfies until all tourist groups have had selfies.
- Each tourist group will only have one selfie.

Payment for selfies:

- The number of bags of food earned for each selfie type is as follows:
    - individual photo - 1 bag
    - group photo - 2 bags
    - individual video - 1 bag per 20 seconds of video
    - group video - 2 bags per 20 seconds of video
    - individual sketch - 6 bags for a 5 minute pose and 10 bags for a 10 minute pose
- If the quokka has a baby in their pouch then the payment is doubled.

Quokka food intake

- Quokkas eat two bags of food a day or three bags if they have a baby.
- Quokkas are able to store their food if they have an excess at the end of each day.
- If quokkas do not eat any food for two days in a row then unfortunately they die.
- If quokkas do not each enough food for five days during the 30 days then they die.

Quokka births

- Each day the probability of each quokka without a baby in their pouch giving birth to a new quokka is 0.05. (Hint: to calculate the probability of event, generate a random number from 1 to 100. There is a 1% chance of each of these numbers being generated so you can nominate numbers 1-5 for a quokka birth).

At the end of each day a summary of the population is displayed with the number of live quokkas, number of deaths, number of new quokkas born and total food supply. Note that the count of quokkas includes baby quokkas.

Specific actions at the completion of the 30 day simulation

At the end of the simulation the success of the Quokka Selfie Quest is evaluated with two calculations.

1. The Quokka Selfie Quest success factor (QSQSF) is calculated as follows:

    (1- (total_quokka_deaths / total_quokkas_at_start)) * 100

The closer to 100, the more successful the program.

2. The quokka population sustainability factor (QPSF) is calculated as:

    (total_quokkas_at_end - total_quokkas_at_start) / total_quokkas_at_start

A positive number indicates that the population is sustainable

The following summary is displayed on the screen.

- The QSQSF
- The QPSF
- Number of days where quokkas died through lack of food.

A summary is written to the file *populationFinal.txt*. The details written to the file will be the numbers of live quokkas, newborn quokkas, dead quokkas and total food supply.

Program and Class Design

The design of the program will be discussed in your Applied Class in Week 9. It is important that you attend this class.

Important Notes

1. Your program must demonstrate your understanding of the object-oriented concepts and general programming constructs presented in FIT9131. Consider carefully your choice of classes, how they interact and the fields and methods of each class. You must use appropriate data structures to store the various objects (quokkas) in the program. You must make use of **both Arrays and ArrayLists** in your program. Make sure that you discuss your design with your tutor. You must document any additional assumptions you made.
2. You will be required to justify your design and the choice of any data structures used at the interview.
3. Validation of values for fields and local variables should be implemented where appropriate. You should not allow an object of a class to be set to an invalid state (i.e. put some simple validations in your mutator methods).
4. Your program should handle incorrect or invalid input and present the user with relevant error messages. No invalid input should crash the program.

5. Exception handling should be used where appropriate.

Assessment

Assessment for this assignment will be done via an **interview** with your tutor. The marks will be allocated as follows:

- 10% - Progress of test strategy and code development, as shown via Ed workspace environment. Your tutor will assess your work during your applied session in weeks 10 and 11.
    - 5% in week 10 for the **Quokka** class and its **test strategy**
    - 5% in week 11 for a draft of the **class diagram** and two further classes, which must be a **FileIO** class and a **client** class. Note that the class diagram should show the individual classes and the interactions between the classes but does not need to include the details within the classes.
- 10% - Test strategy for the **Quokka** class.
- 10% - Class diagram, Java code quality and object-oriented design quality. This will be assessed on code quality (e.g. compliance with coding standards) appropriate design and implementation of classes, fields, constructors, methods, and validation of the object's state.
- 10% - Program functionality in accordance to the requirements.
- 60% - Oral assessment.

A reminder that you must use the workspace environment in the Ed platform (opposite this assignment specification) to code all parts of your program. You must not copy and paste large sections of code from other sources, and you must acknowledge *any* code in your assignment that has been taken from other sources.

Marks will be deducted for untidy/incomplete submissions.

You must submit your work by the submission deadline on the due date (a late penalty of 10% per day, inclusive of weekends, of the possible marks will apply). There will be no extensions - so start working on it early.

All submitted source code must compile. Any submission that does not compile, as submitted, will receive a grade of 'N'.

Oral assessment

As part of the assessment you will attend an interview following the submission date. At the interview, you will be asked questions about your code. You will asked to explain your code/design, modify your code, and discuss your design decisions and alternatives. Marks will be awarded for your answers (the marker may also delete excessive in-code comments before you are asked to explain that code).

In other words, you will be assessed on your understanding of the code, and not on the actual code itself.

For **on-campus students**, interview times will be arranged in the applied classes in Week 12 and will take place on campus during the following week.

For **online students**, the interviews will be arranged in the applied classes in Week 12 and will take place online via zoom during the following week. You must have audio and video capabilities on your computer that can be used for the oral assessment.

It is your responsibility to make yourself available for an interview time. **Any student who does not attend an interview will not receive a pass grade for the assignment.**

Submission Requirements

The assignment must be submitted by **11:55pm Friday of Week 12 (26 May 2023)**.

The submission requirements for Assignment B are as follows:

- The main class in your program **MUST** be called **QuokkaSelfieQuest.java** and it should contain the **main()** method to start the program.
- Class diagram submitted as a pdf file.
- Test strategy for **Quokka** class submitted as a pdf file.
- Submit all your work (coding, class diagram and test strategy) via the Ed platform.
- Re-submissions are allowed (and encouraged) before the submission deadline. Please ensure however that you do not click on the submit button *after* the due date. Your final submission will be used for grading purposes, and any submission made after the deadline will incur a late penalty.
- A signed Assignment Cover Sheet. [Note: You are required to download the Assignment Coversheet, sign the document and upload the pdf file in the Ed platform (you may drag and drop to the Toggle Pane)]

Marks will be deducted for any of these requirements that are not complied with.

Warning: there will be no extensions to the due date. Any late submission will incur a 10% per day penalty. It is strongly suggested that you submit the assignment well before the deadline, in case there are some unexpected complications on the day (e.g. interruptions to your home internet connection).

Plagiarism and collusion

Plagiarism and collusion are viewed as serious offences. All submitted code will be subjected to a similarity checker, and any submissions determined to be similar to a submission from a current or

past student will be investigated further. The outcome of the decision pertaining to plagiarism and/or collusion will be determined by the faculty administration. If it is determined that plagiarism or collusion has occurred then you may be severely penalised, from losing all marks for the assignment, to facing disciplinary action at the Faculty level. To ensure compliance with this requirement, be sure to do all your coding in the Ed workspace environment and do not copy and paste any code into the workspace environment.

In cases where cheating has been confirmed, students have been severely penalised, from losing all marks for an assignment, to facing disciplinary action at the Faculty level. Monash has several policies in relation to these offences and it is your responsibility to acquaint yourself with these.

Student Academic Integrity Procedure (https://www.monash.edu/__data/assets/pdf_file/0004/2300935/Student-Academic-Integrity-Procedure.pdf))