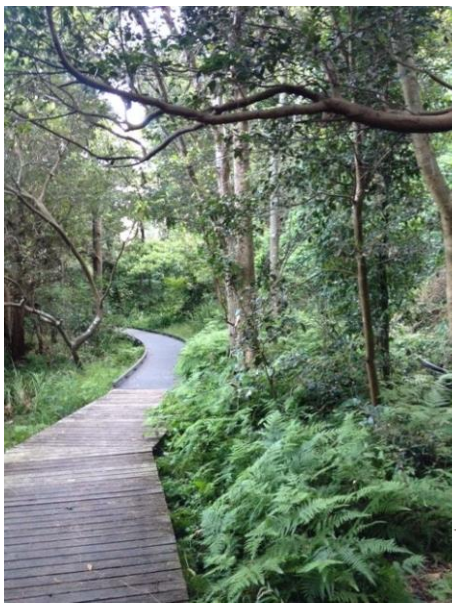
Nature Trail Game



Introduction

Assignment 1 has two components: a programming task and a unit test. These components are assessed independently but BOTH are required. The programming component is worth 10% of the marks for your final assessment in this unit and an associated test held during tutorial time is worth 5%. You must attempt both components to be awarded any marks for Assignment 1.

This document specifies the programming component of Assignment 1. This component is due by 11pm Thursday of Week 7 (18th April, 2019). **Heavy penalties will apply for late submission.** This is an individual assessment task 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.

The assignment must be done using the BlueJ environment.

The Java source code for this assignment must be implemented according to the **Java Coding Standards for this unit.**

Any points needing clarification may be discussed with your tutor in the lab classes.

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

1. *design, construct, test and document small computer programs using Java;*
2. *interpret and demonstrate software engineering principles of maintainability, readability, and modularisation;*
3. *explain and apply the concepts of the "object-oriented" style of programming.*

Specification

For this assignment you will write a program which will allow a person to play a game *Nature Trail* against the computer. This section specifies the required functionality of the program. **Only a text interface is required for this program**; however, more marks will be gained for a game that is easy to follow with clear information and error messages to the player.

The aim of *Nature Trail* is for a person (H) and the computer (C) to compete against each other to gather the most points for animal sightings along a nature trail.

The nature trail is represented on the screen by a line of space-separated underscore characters as shown in Figure 1. The letters '**S**' and '**F**' represent the **S**tart and **F**inish positions. These are the first and last positions on the trail. The positions of the human player and computer are shown on the trail. If both players occupy the same position then this is indicated as **HC.**

```
S _ _ _ _ _ H _ _ _ C _ _ _ _ _ _ _ _ F
```

Figure 1 Nature Trail representation

Game play

The *Nature Trail* game begins with a message requesting the player to enter the length of the trail. The trail can be any length from 10 to 20 places inclusive. Following this there is a message requesting the human player to enter their name. The name must be between 1 and 6 characters in length and not begin or end with a space character. The other player will be the computer.

To determine who has the first turn each player 'rolls' a 4-sided dice and the player with the higher number takes first turn. If the numbers are equal then the dice is rolled again by both players. This continues until one player has a higher number. Note that the human player 'rolls' the dice by pressing the enter key, but the game automatically 'rolls' the dice for the computer player.

The game play begins with both players (human and computer) at the start position (**S**). The players take turns to roll a 4-sided dice and move forward the number of places shown by the dice. At each position along the nature trail there is a random chance of sighting an animal and gaining points according to the type of animal. Some positions have an action because of a special feature on the trail. The action involves moving forwards or backwards along the trail. When the player moves to a position they gather any points for an animal sighting and then follow the special action. The game ends when the first player reaches the finish position (**F**). The winner of the game is the player who has collected the most points.

The following are the game rules:

1. Each player starts the game with zero points.

2. When a player lands on a position on the nature trail they have a 50% chance of sighting an animal. There are five different animals and there is an equal chance of sighting each type of animal, but only one animal can be sighted for each turn. When a player sights an animal they are awarded points according to the type of animal, as shown in Table 1. Note that animal sightings are not possible from the start and finish positions.
   HINT: talk to your tutor about how the random sightings can be implemented.

Table 1 Points awarded for different animal sightings

Animal	Points
Koala	10
Emu	7
Wombat	5
Kangaroo	2
Redback spider	-5

3. When the player lands on a position with a nature feature they follow an action as shown in Table 2. Nature features are randomly assigned along the trail at the beginning of the game. Each of the features shown in Table 2 must appear once on your trail and there must be no more than one feature per position. Note that nature features are not assigned to the start and finish positions. Note also that a player cannot move back further than the start position. If a feature causes a player to move back further than the start position then they will be placed on the start position.

Table 2 Nature features and associated actions

Nature feature	Action
Creek	Move back 2 places
Bridge	Move forward 4 places
Fallen tree	Move back 3 places
Landslide	Move back 5 places

4. A player's turn ends with the completion of any action associated with the position they land on following their dice roll. In other words, if a player rolls a ‘4’, they move forward 4 places. They gather points for any animal sighting at the position they land on. If the position has an action, for example, "Move back 5 places", then they move back 5 places and their move is completed. They do not sight any animals or follow any actions from this final position.

5. A player does not need to land exactly on the finish position. For example, if they are one position from the finish and they roll a ‘3’ then they move to the finish position and the game is over.

6. The dice roll result, any animal sighted and action performed are displayed to the screen as they occur. The players’ positions and their total points accumulated (scores) are displayed to the screen at the end of each turn.

7. The first player to reach the finish position is awarded 10 points. The game ends and the scores are calculated and displayed. The player with the highest number of points wins the game and the winner is displayed to the screen.

Program design

Your program must consist of at least five classes: *Player*, *Game*, *Dice*, *Trail* and *NatureFeature*. A suggested class design is shown in Figure 2. There are suggested fields for each class. If you include any extra fields then you must be able to justify these. You may want to include comments in your program to state any assumption made.

The remainder of this section gives details of these classes.

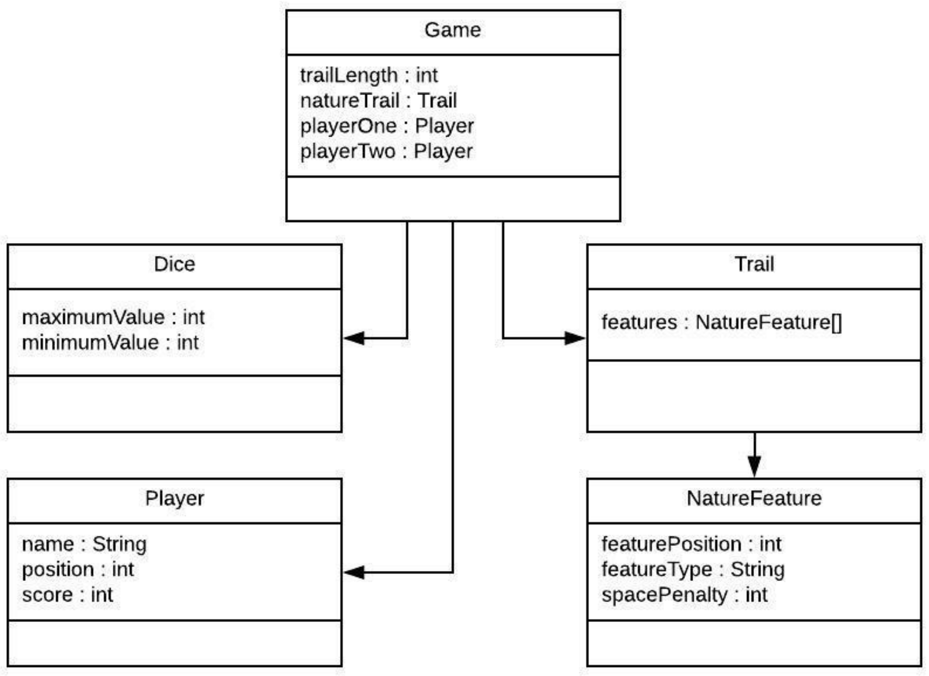


Figure 2 Class diagram for the Nature Trail game

Player class

The Player class will specify the attributes and behaviours of a player. An object of the Player class will have the following fields:

name – the name of the player.

position – the current position of the player on the nature trail

score – the cumulative game score

The class must also have a default constructor and a non-default constructor that accepts a value for the name of the player.

The Player class should also have *appropriate* accessor and mutator methods for its fields. You should not allow an object of class Player to be set to an invalid state. There should be no input from the terminal or output to the screen. A Player object should also be able to return its state in the form of a String.

Game class

The Game class will be in the same BlueJ project that contains your Player class. The Game class will manage the playing of a game. It will have the following fields:

trailLength – length of the trail

NatureTrail – an object of type Trail

Player1 – an object of type Player

Player2 – an object of type Player

Note that one of these players will be the computer.

The Game class will have methods to manage the playing of the game. These should include (at least) the following behaviours:

- Display a welcome message on the screen.
- Request the length of the trail.
- Request the human player to enter their name.
- Roll the dice.
- Move the player a number of places.
- Request the player object to increment their score.
- Randomly generate the positions of the features on the nature trail.
- Display the nature trail with the position of each player.
- Player takes a turn.
- Display the game result.

Dice class

An object of the Dice class will generate a random number from 1 to a maximum value. It will have two fields:

maximumValue – maximum value of the dice roll

minimumValue – minimum value of the dice roll

Trail class

The trail class will store an array of all the nature features on the trail. An object of the Trail class will have one field:

features - an array of 4 elements storing objects of the NatureFeature class indicating the type, position, and penalty associated with each nature feature.

NatureFeature class

An object of the Trail class will have three fields:

featurePosition – the position of the feature on the trail

featureType – the type of feature as shown in Table 2

spacePenalty – the number of places to move

Hint: You might want to store move back spaces as a negative number.

Assessment

Assignment 1 has two components: a programming task and a unit test. These components are assessed independently but BOTH are required. Note you must attempt both components to be awarded any marks for Assignment 1.

Programming Task Assessment (10%)

Assessment for this component will be done via an interview with your tutor.

The marks for your program code will be allocated as follows:

- 35% - Object-oriented design quality. This will be assessed on appropriate implementation of classes, fields, constructors, methods and validation of the object's state.
- 5% - Adherence to FIT9131 Java coding standards.
- 60% - Program functionality in accordance to the requirements.

You must submit your work by the submission deadline on the due date (a late penalty of 20% per day, inclusive of weekends, of the possible marks will apply - up to a maximum of 100%). There will be no extensions - so start working on it early.

Marks will be deducted for untidy/incomplete submissions, and non-conformances to the FIT9131 Java Coding Standards.

Please note that your program code will be submitted to a code similarity checker.

Interview

You will be asked to demonstrate your program at an "interview" following the submission date. At the interview, you will be asked to explain your code/design, modify your code, and discuss your design decisions and alternatives. Marks will not be awarded for any section of code/design/functionality that you cannot explain satisfactorily (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.

Interview times will be arranged in the tutorial labs in Week 7. It is your responsibility to attend the lab and arrange an interview time with your tutor. **Any student who does not attend an interview will receive a mark of 0 for the assignment.**

Unit Test Assessment (5%)

This assessment will be a written assessment and will be held during the tutorial times in Week 8. Keep in mind that students are expected to have finished ALL their code by the submission deadline in Week 7, and more importantly to have done their own work.

The unit test will be a closed book test wherein students will be required to WRITE code to answer a few questions. The questions will cover material from Weeks 1 – 6 of the lectures. Students should bear in mind that BlueJ or any other electronic device will not be permitted, so students need to know HOW to write their own code in order to finish the unit test.

Submission Requirements

The assignment must be uploaded to Moodle by 11pm Thursday of Week 7 (18th April, 2019).

The submission requirements for Assignment 1 are as follows:

A .zip file uploaded to Moodle containing the following components:

- the BlueJ project you created to implement your assignment.
- a completed **Assignment Cover Sheet**. This will be available for download from the unit's Moodle site before the submission deadline. You simply complete the editable sections of the document, save it, and include it in your .zip file for submission.

The .zip file should be named with your Student ID Number. For example, if your id is 12345678, then the file should be named 12345678_A1.zip. Do not name your zip file with any other name.

It is your responsibility to check that your ZIP file contains all the correct files, and is not corrupted, before you submit it. If you tutor cannot open your zip file, or if it does not contain the correct files, you will not be assessed.

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 the 20% 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).

Extensions

All requests for extensions must follow the faculty guidelines and submit the required forms as soon as possible. In addition, when emailing for an extension, please remember to include all code completed until that time. This assignment cannot be completed in a few days and requires students to apply what we learn each week as we move closer to the submission date.

Plagiarism

Cheating and plagiarism are viewed as serious offences. In cases where cheating has been confirmed, students have been severely penalised, with penalties ranging 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.

Plagiarism (http://www.policy.monash.edu/policy-bank/academic/education/conduct/plagiarism-policy.html)