Introduction:

Names: Tjun Tji Oon (vuID: 2575214)

The system I am going to design and implement in this course is a game in the style of Zork in that it is a text based adventure game which the player (user) will affect via entering commands in the terminal. As a reference, more information about the game Zork can be found here:

https://en.wikipedia.org/wiki/Zork

As well as a walkthrough so you can see how the game plays out:

https://www.youtube.com/watch?v=1q9Q2gwqw7U

The game I am creating is called PORK. As you can probably tell, its name is heavily inspired by Zork however that is where the similarities end, with the only major characteristics shared between the two games being the genre. This is because PORK is a game where the main protagonist is a pig! In fact if this game were to be commercialized its blurb might sound something like this:

PORK! You are a run of the mill pig living a peaceful life on a nondescript farm. PORK is a text based interactive role playing game in which the player navigates the world through the eyes of a pig! With your fellow porcine or otherwise animal allies, it is YOUR job to find a way to keep the meat on your bones and free the animals from the farmer's tyranny. Talk, steal, fight and explore your way through the wondrous world of PORK!

The major goals for this game are:

- To provide a fun experience for the player.
- To immerse the player in the world of PORK and make them feel like they have many choices with serious consequences at any given moment.
- To challenge the player's critical thinking skills through a puzzle like progression system (at least at some point).

The key goals for the design and implementation of the game from the designer's perspective:

- Make the game as easily customizable/configurable as possible through object oriented programming.
- Separate the 'engine' from the storyline/player choices as much as possible.
- Write well annotated and easily understandable code in a clear and logical flow.

As for how the game works, it will take place via text, and so the opening will render the player subject to some text crawl that sets the background for the game. From which point on I will introduce the player to the game's mechanics through a tutorial of some kind. Following that, the player will be left to their own devices and have to use their wits to figure out how to progress

through the game. At some point in the game, I would like to implement a puzzle or logical element to the game (this might even be throughout the whole game- making it an adventure/puzzle hybrid game) by making the player have to follow certain steps to complete the game. In addition to that, I would like to have some combat situations occur in game and thus implement a level, inventory and attack system. At this stage in planning, the ideal combat situation would not involve any luck but rather the outcome would depend on the player's strategy and preparation before coming into the battle (i.e having to be a certain level etc.).

The way I will try to achieve this design is by implementing the game via 4 separate classes; one of them will be a 'character' class whose function will be to provide a template for the playable and non playable characters in the game. The class might contain attributes such as health or level- which will be useful when implementing the combat system. The second class will be an area or zone class, which I will use to implement the different areas or levels in the game. This class will contain the objects the player can interact with i.e items, characters etc. The next class will be an item class with attributes such as damage or durability perhaps (although this is not a super necessary feature for this game). The final class will be the main engine of the game which will most likely be involved in playing out the static pre-written story sequences and doing things like keeping track of how many moves the player has made.

Functional features

Name: Tjun Tji Oon

This section will list all the functional requirements for the implementation of PORK. The criteria behind choosing the functional requirements mainly revolve around the fact that it is a game, which means a user has to be able to 'play' it, implying that the user must be able to perform an action to affect the game. Furthermore, a user must be offered the chance to win, lose or pursue an objective in the game or it ceases to be a game and more a simulation. These concepts were further refined to be the functional requirements of this particular game.

ID	Functional requirement	Description
1	Commands	The player must be able to interact with the world by entering commands in the terminal. The available commands are: - i : opens inventory - h : displays health - l : displays level - c : shows character info - attack : enters combat mode or allows you to attack an object - look around : generates a list of objects you can

		interact with - go to : allows you to go to different areas ie. house provided they can be accessed - pick up : allows you to pick an item up - eat : allows you to eat an edible item in your inventory - use : allows you to use certain objects - talk to : allows you to talk to certain characters - steal from : allows you to steal from certain characters - exit : quits the game
2	Environment	The player's environment shall be able to be interacted with and changed permanently, i.e when items are picked up they cannot be picked up again.
3	Zones	The player shall be able to transfer to different areas of the 'map' at certain points of the game.
4	Story	The game shall have at least 2 distinct endings depending on the player's choices throughout the game.
5	Items	There shall be items found throughout the game that the player can use/consume.
6	Battle	There shall be at least one battle situation where the player can choose to either attack, use an item in their inventory, or run, and lasts until either combatant is dead or the player chooses run (and the battle is escapable).
7	Exit	The game shall be able to be quit via the in game command 'exit'

Quality features

Name: Tjun Tji Oon

The quality features were chosen in order to make the game the highest quality product possible, the features that make the program a *game* are found in the functional features. The quality features are non-necessary to the game (as in you can play the game without them) but add more value to the product (the product would be really bad without them).

ID	Name	Quality attribute	Description
1	Command check	Reliability	The program should check whether the command entered is valid and follows the appropriate syntax: i.e talk to [character] rather than talk [character]. This would create a uniform, reliable platform for the player to navigate the game.
2	Object oriented	Maintainability	The game should be implemented using different classes for different purposes so that adding or removing things from the game is easier. The game becomes more flexible and configurable.
3	Tutorial/help	Usability	There should be a tutorial on how to play the game in the very beginning, as well as a help function that prints all the available commands for the user. This is to ensure the player can play the game easily without misremembering or mistaking commands for example.
4	Instant responsiveness	Responsiveness	The game should respond instantly after the player performs an action.

Libraries

Name: Tjun Tji Oon

Library	Description
Java class library	This library will be used mainly for the packages util and lang, which will be used to implement the command parser by allowing the game to take user input.