MAKING A TEXT ADVENTURE GAME

First make sure that know what a text adventure game is.

There are two basic types:

(i) the classic text adventure like Zork, Colossal Cavern, Hitchikers Guide to the Galaxy
Here you type in commands using words. You're exploring the world in order to solve some sort of problem or puzzle and win the game.

(ii) a game that is combat based, where you have certain specific keys to do certain things. e.g. a=attack, … There can also be shops, etc. You're not exploring the world.

Some things can be applied to both types, but I'm going to focus on the first type.
If you plan to do the second type, please check with me so that I can figure out what you plan to do.

Your game needs at least the following:

1. six or more locations (rooms)
2. items that can be picked up (or equipped, etc) and possibly used – at least five items
3. a player class for the player
4. some sort of language parser. We don't want a game which just gives you 3 options in a room. In this case, you just have to try each option. In a game where you type in commands, you never get to find out certain things unless you think to do something, like "look under carpet".
5. a puzzle that needs solving or a quest that needs accomplishing in order to win (and the game should be winnable)
6. a way to lose (starving, getting injured and losing health points, running out of time: you only have 30 turns, etc).
7. and inventory.
   This should either be an array, where your inventory is fixed to a certain number of items (size), or an arraylist, where you can have any number of items, but you have to take into account the weight. You can only carry so much (until you level up!) and each item has a weight.

The rooms, player and items will all be objects. You'll have to decide on the properties that the player and items have. If needed, you can have subclasses (I can discuss this later).

Other additional possibilities:

- a room that changes based on some action (e.g. an earthquake/explosion collapses a tunnel so that it can no longer be used)
- a shop
- a combat system (enemies that spawn in unoccupied rooms?)
- doors to rooms, with keys?
- containers (taking A out of B, or putting it into B)
- darkness : if you don't have a light, then you die in a dark room
- anything above and beyond the basic requirements

~~You'll be working in groups of three, and using GitHub to coordinate code changes. You can each have different tasks. There should be one person responsible for merging changes into the master branch (more on this later).~~

You can work with a partner as long as both of you contribute to the project.

Rooms:

– These will be discussed in class (along with an example)
– These will be in a hashmap.
You'll have a name (key) to retrieve the room object (value) from the hashmap.
The hashmap is a way to store and retrieve all rooms and room info.
– The name is internal to the program and not something that the user ever needs to know.
– The name is used for moving from one room to another.

Items are similar.

I'm assuming that the location of the item will be in other lists. The Item is not going to store its own location. The item can either be in a room (itemList) or in your inventory.

Alternatively, one could design things so that the item tracks its own location. This means that everytime you list your inventory, you would have to search through the whole itemlist hashmap and find all items that have locations = "inventory". I don't think that this works as well as the initial way.

How to start with the language parser:

- make lists of possible commands
- organize them into number of words per command
- maybe look at the structure (verb-object) or (verb object preposition object)
- make note of synonyms (e.g W, West), (take, pick up)

I have an online template for how to start the program.

There are likely more advanced ways of translating English into methods to call. Don't just copy code, make sure that you understand it.

ALWAYS check with the teacher first for permission if you're copying sections of code from the internet.