TextAdventure

This repo is used to store all the projects that implementing the game Text Adventure, in a way so called "stepvis improvement"". We start by analysing a simple game written in structure of imperitive / functional programming

https://da.wikipedia.org/wiki/Programmeringsparadigme

In the following 4-5 weekes, we are going to focus on re-writting the game while adding the following new features and concepts

> Flow chart and UML for object oriented programming

> Design Principles and Design Patterns

> Classes inheritance

> State Machine

Introduction

The game "text adventure" is a type of IF (Interactive Fiction), here is one example: https://eblong.com/zarf/zweb/dreamhold/

Chart Drawing

> ### Flow chart
>
> > *Do it yourself*: Draw a flow chart for the game "text adventure" in the folder Step1. https://en.wikipedia.org/wiki/Flowchart#Common_symbols

> ### Class diagram
>
> > *Do it yourself*: read through the UML tutuorial https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/ and try answer the following questions:
> >
> > - What is UML class diagram ? What can UML class diagram visualize ?
> > - What are the three main components in the UML Class notation
> > - This tutorial https://www.gleek.io/blog/class-diagram-arrows.html introduced several relations between classes. In our project, we are going to use the following ones. Make sure that you understand the meaning of these relations. can you find examples of each relation from the game text adventure ?
> >
> > > - Generalization
> > > - Association
> > > - Aggregation
> > > - Composition
> >
> > - Analyse the "class diagram for an ATM system" in this article, https://www.lucidchart.com/pages/uml-class-diagram. Describe the relations between the

> > classes **Bank**, **Customer**, **ATM**, **Account**, **ATM transactions**, **Current Account** and **Saving Account**

Some improvement ideas ?

Home Work

> - Change the codes of text adventure in Step1 into object oriented. You can follow the steps:
>
> > 1. Make a list of all the involved objects in the game.
> > 2. Add the attributes and operations (methods) to each object (class)
> > 3. Draw the UML class diagram.
> > 4. Implement according to the UML (Optional, can wait til next time)

Or if you'd rather make you own story

> - Make a sketch (in words or drawing or both) of your own text adventure game. It must include at least four scenes, two characters, some inventories to collect and a win condition.
> - Follow the 1,2,3,4 from above