FIT 1048 – ASSIGNMENT3

PROGRAM REFLECTION

NAME:LOH ZHI WEI STUDENT ID: 31123074

CLASSES & OBJECT-ORIENTED DESIGN

In this program, I've created Player, Suspect, Item and Location class respectively.

First, the class attributes and variables are all private thus they are only accessed with public methods declared so there won't be unauthorized changes to the class data, this way of protecting the data reflects the encapsulation concept in object-oriented programming.

Through instantiation of the same classes in the program, the amount of code that needs to be written and debugged is only limited to the class instead of the whole program, isolation of the class code reflects the abstraction concept of object-oriented programming.

Due to the property of this program, the classes are interrelated, as an example player class relates to item class for inventory and location class relates to item and suspect present, but the item and suspect classes are independent.

However, with minor alterations such as using string list for the player class inventory, the classes will be independent and ready to be used elsewhere.

IMPLEMENTATION OF DESIGN

The first challenge was to populate the data of the class objects and it was solved through creating a temporary list to store the data strings for use of class mutators.

Next issue is to process the string commands ,through checking the length of the command and space between a two-word command, the verb and noun are extracted successfully.

When coding for the classes, changes are made from the initial UML class diagrams. For example, iterator functions are added to search for object based on its attributes. An example for extra function removed is updateGameStats() function which is replaced by directly updating displayGameStats().

Extra conditions for the game interactions, such as the item picked up must be removed from the location, only the suspects and item presented allow interactions ,whether the final weapon showed is in the inventory and so on are all checked carefully through if statements when processing commands.

The last place modified is the suspect class status, instead of string variable, Boolean is used as inspired by item class, the murderer will have a true status while others are false and the victim will be stored in a global variable which then allows the global suspects list without victim to be used easily throughout the program such as when moving the suspects around.

IF I WERE TO START AGAIN…

I would utilize the indexes to access class objects in the global list instead of matching the object names through iterator find if function.

This will be more efficient and by accepting numbers in the command, users can save a lot of effort on typing suspect, item and location names.

By storing the indexes instead of class objects in class variables, the classes can be made independent and better for code reuse as well.

For the quality of the storyline, I would like to specify the murderer each round for different versions of stories so the descriptions and conversations will be more dramatic and relatable.

THANKS FOR READING ! 😊

END OF REPORT