Doctors’ surgery README

When working with the UML diagram it was first determined that the healthcare professional class was the parent class to doctor and nurse. From this it was determined that implementing an inheritance-based class structure would be the best practice.

The order of classes being established was determined by the diagram as well as the relationship and utilization of the attributes or functions within the class. One that was difficult to determine was patient and prescription. To be able to run the function request\_prescription it required a prescription to exist, this was difficult to implement and ultimately the repat prescription function was not completed.

Another function that was not completed was the appointment schedule, the appointment can be created and called upon but adding additional information such as the time, name or date of the appointment has not been completed. When calling upon the appointments listed in the schedule it just calls upon where the appointment in question is held. This was not completed due to time restraints when working on the system in question.

The best implementation of an appointment schedule that could be thought of when working on the system was to use a list. Adding an appointment would amend the list and a new value would be added. By working with the index the entries in the list could be removed or added. Notes are left within the code explaining the different ways this could be done however as stated previously this was not completed due to running out of time.

When testing the system, scenarios were created and by running these scenarios and comparing the results it was easily identifiable if the code worked as intended.

One scenario created was to test if a doctor could create or prescribe a prescription. This was tested by creating a patient and then through the doctor class and its function, creating a prescription then looking for said prescription and calling it to return with the prescription in question. This was successful however implementing a repeat request was not completed and was difficult to work out how to make it work with the layout of the current system.

The scenarios used to test the current system state can be found at the bottom of the systems code, if there had been further time to develop this systems code more scenarios would be written and implemented for a fully rounded and substantial testing phase to the development cycle.

In conclusion, the creation of this surgery object-oriented system is incomplete, some of the code brings up errors or is incomplete. For this to be a working system it would require more research into object-oriented systems and time to edit and develop the classes and functions within them. Furthermore the implementation of public and private attributes would be implemented to protect patient data and to clean up the system.