1. **The factory**

The application can parse the report in two different formats as a pdf file or as a text file. In the future it could be possible that the hospitals share the information in another file format than the two which are now implemented. To extend the functionality in the feature, a parent class “HsmrParser” was created which must be extend in the implementations of the specific parsers. Two different parsers are implemented jet, but more can be added in the future. A factory was created to get a instance of the parser based on the parser types which are defined as constants in the “[parserTypes.py](http://parserTypes.py)” file.

**Questions**

* What is a factory?
* Does the implementation of the factory method follow the Interface Segregation Principle?

1. **Single reponsibility**

The application uses the CCS classifiation to extract the information from the HSMR reports. The application uses a csv file with the ccs index and the corresponded dutch description.

This information is provided by the [CBS](https://www.cbs.nl/nl-nl/onze-diensten/methoden/onderzoeksomschrijvingen/aanvullende%20onderzoeksbeschrijvingen/hsmr-2016-methodological-report) as Microsoft Office Excel file. The nessesary data was extracted and saved as CSV file

**Question**

Review the python files starting with Ccs. Are the Ccs python files according the [s](https://en.wikipedia.org/wiki/Single-responsibility_principle" \o "Single-responsibility principle)[ingle-responsibility principle](https://en.wikipedia.org/wiki/Single-responsibility_principle): "Every class should have only one responsibility”

1. **The base classes**

In the code several base classes are used. Can you find examples of the [L](https://en.wikipedia.org/wiki/Liskov_substitution_principle" \o "Liskov substitution principle)[iskov substitution principle](https://en.wikipedia.org/wiki/Liskov_substitution_principle): "Functions that use pointers or references to base classes must be able to use objects of derived classes without knowing it." Explain your answer

1. **The local settings object**

A settings file was used inside the application to store the settings which can be changed over time or is user specific. For instance, the path to a temp directory is saved inside the settings file because they will be different between users. Also, specific table names of the HSMR report, API URLs, API headers are saved in this file because they can change and must be easily assessable. The Settings object is implemented as Singleton object inside the application to prevent multiple instances of the same object. It also prevents multiple unnecessary parsing of the settings file and eventually different settings when the settings are manually changed during runtime of the program.

**Questions**

* Search for the Settings object. What makes the Settings object a singleton object and is a singleton object SOLID?
* The hospital types codes are stored in a python module hospital\_[types.py](http://types.py). Is this a logical solution?
* Is there an alternative solution for these kind of local settings and parameters

1. **Class diagram**

Draw the class diagram for this program. Once drawn identify the SOLID solutions