Task 2

Module Code	Module Name	Classes
CM1101	Computational Thinking	Theory; Python; Communication
CM1102	Web Applications	PHP; HTML; SQL; WebTechnology
CM1103	Problem Solving With Python	Python
CM1202	Developing Quality Software	SoftwareDevelopment; Communication
CM1205	Architecture and Operating Systems	AssemblyLanguage
CM1206	Fundamentals of Information Systems	Communication; Java
CM1207	Introduction to Java	Java
CM1208	Maths for Computer Science	Mathematics
CM1209	Object Oriented Java Programming	Java; ObjectOrientated
CM6122	Mobile Development with Android	MobileApplications; SoftwareDevelopment
CM2101	Human Computer Interaction	HumanComputerInteraction
CM2102	Database Systems	Databases; SQL
CM2105	Data Processing and Visualisation	DataVisulisation; Python
CM2202	Scientific Computing and Multimedia Applications	SignalProcessing; ImageProcessing; Graphics; Mathematics; Matlab
CM2203	Informatics	SemanticWeb; Ontologies
CM2204	Advanced Programming	C
CM2207	Introduction to the Theory of Computation	Theory; Java
CM2302	Communication Networks and Pervasive Computing	Communication; Forensics
CM2303	Algorithms and Data Structures	Algorithms; DataStructures
CM6213	Enterprise Web Applications with Java	WebTechnology; Java; ObjectOriented
CM6222	Cloud Performance and Scalability	Cloud
CM3102	Graphics, Visualization and Computer Vision	Graphics; Matlab
CM3104	LargeScale Databases	Databases; SQL
CM3105	Security and Forensics	Forensics
CM3107	Knowledge Management	KnowledgeManagement
CM3108	Computational Intelligence	ComputationalIntelligence
CM3109	Combinatorial Optimisation	Optimisation
CM3113	Computer Vision	ImageProcessing

Task 3

When developing the ontology I first looked at the provided list of modules and their descriptions, I noticed that many modules had overlapping topics and other modules were very unique, this arises from computer science being such a broad subject with many branches. After reading through this list and gaining an appreciation for the core concepts of each module I began to create my classes. I made sure that where possible I used class names that could be applied to multiple modules in an attempt to highlight fundamental similarities and differences of the modules. This serves to unify the concepts in similar modules so a student can select a broader range of topics with little overlap to gain the widest knowledge base from the course. Alternatively if a student wishes to specialise they can select similar modules that cover similar topics. The danger when creating these classes is with creating too many specific classes for each module. I have endeavoured to keep my ontology as simple as possible and limit the classes I have used to simplify the process. I believe this better serves either purpose of selecting similar or dissimilar modules, the more classes you use the more overlaps will be present and the more difficult it will be to choose modules effectively.

With all modules mapped to the ontology it provides a different way to choose modules. Whereas normally you are confronted with a list of names that don't always mean much and lengthy descriptions to read through you are instead presented with a list of concepts. So if a student knows that they have an interest in java they can view all the modules linked to java and have a look at other tagged classes to decide if that module as a whole is something that interests them. The ontology makes it much easier to categorise modules at a glance and quickly identify where overlaps exist. Conversely it also makes modules with few overlaps stand out. This helps with reviewing the course and module content because some modules are very similar and can be merged and at the other end of the scale some modules are so unique and cover so much content they could be split or parts could be integrated into other topics, this all serves to make a much more streamlined learning experience. Overall I think that using the ontology is a much better way to represent the core concepts of the modules and the course as a whole opposed to the current format which manages to be both segmented and repetitive.