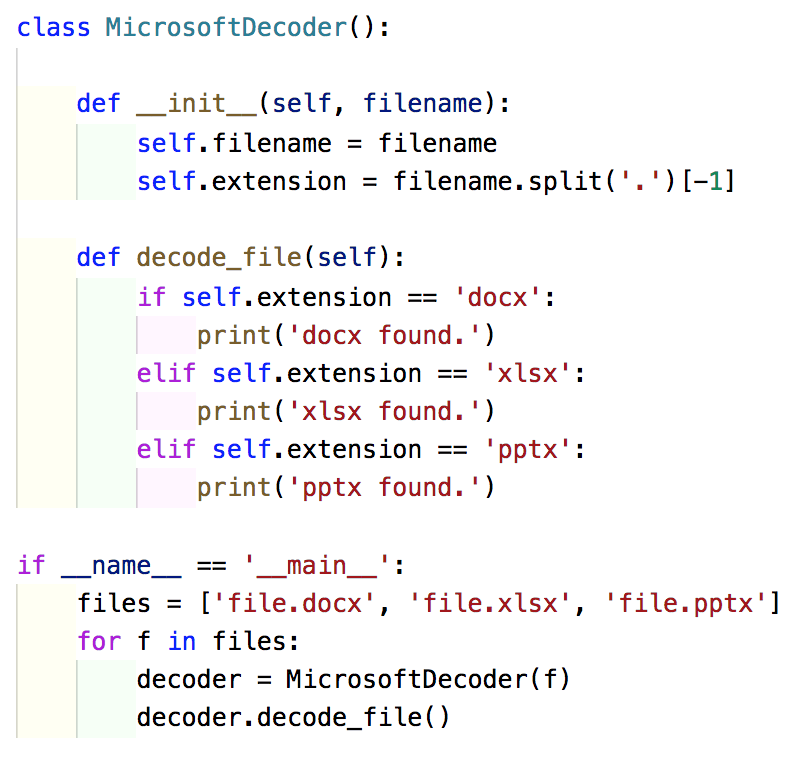
# IN710 Object Oriented Systems Development

# Semester 1, Practice Exam 2019

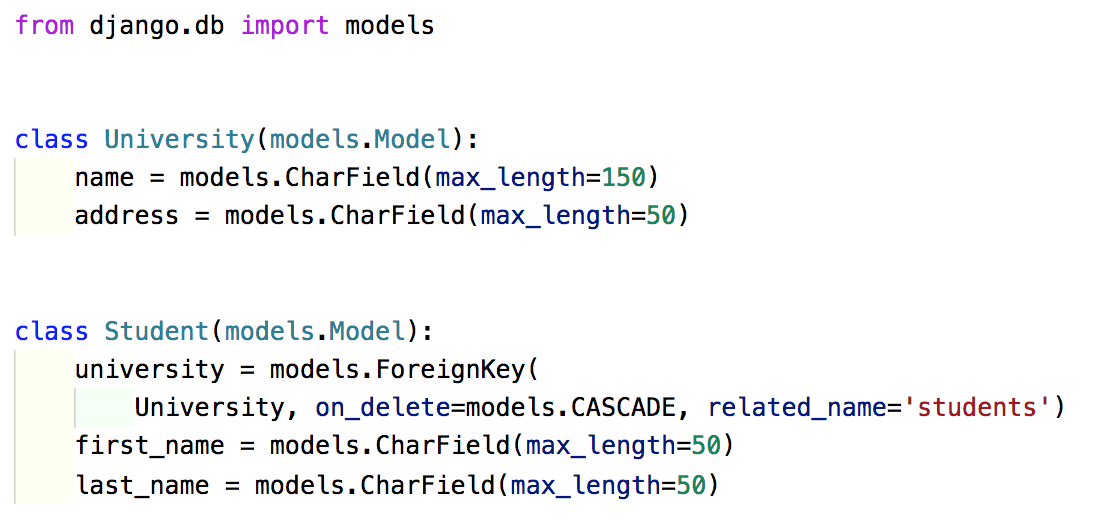
1. One of the **Gang of Four** design patterns “defines an interface for creating an object, but let’s subclasses decide which class to instantiate”.
   1. Which design pattern is it?
2. You are writing code to monitor the performance of a nuclear reactor. Temperatures in the reactor’s core are monitored by a big thermometer, which you can read via existing software. In the event of the core overheating, you want an alarm bell to sound, and a warning light to flash.
   1. What design pattern will you use to write this code?
   2. What important methods would you put on the relevant objects? Please do not write any method code.
3. Describe the following design patterns – advantages and disadvantages:
   1. Adapter
   2. Decorator
   3. Strategy
   4. Template
4. Consider the following Python code:



* 1. Which SOLID principles(s) does this code break? What happens if the programmer decides to handle **.pub** files as well?
  2. Explain how the programmer could use the Strategy pattern to improve the code above. Sketch out the classes and methods needed below. It is not necessary to write detailed method implementations. Do not worry about handling unknown file types.
  3. How would you remove the following duplicate code? Note that the code is not the same, but it’s mostly the same.

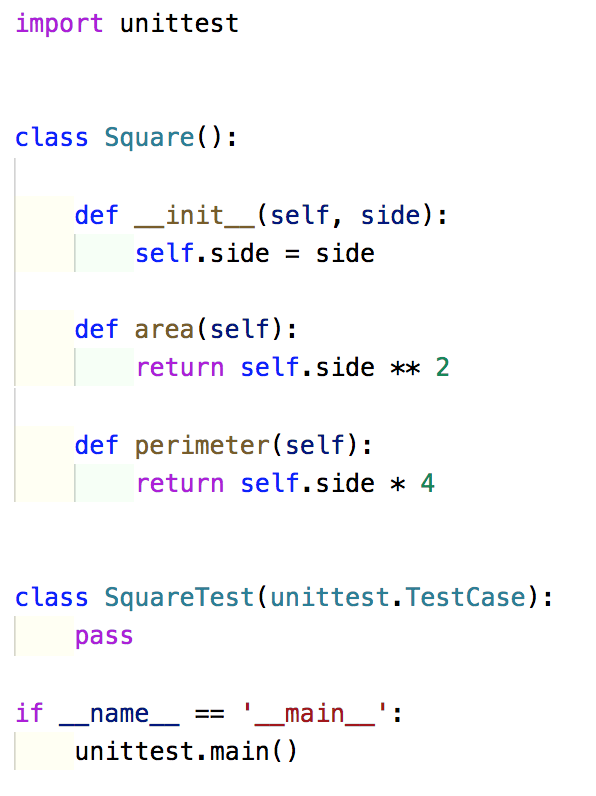
**HINT: It is OK to use a dictionary in the MicrosoftDecoder class to map extensions to strategies.**

1. Consider the following Django models:

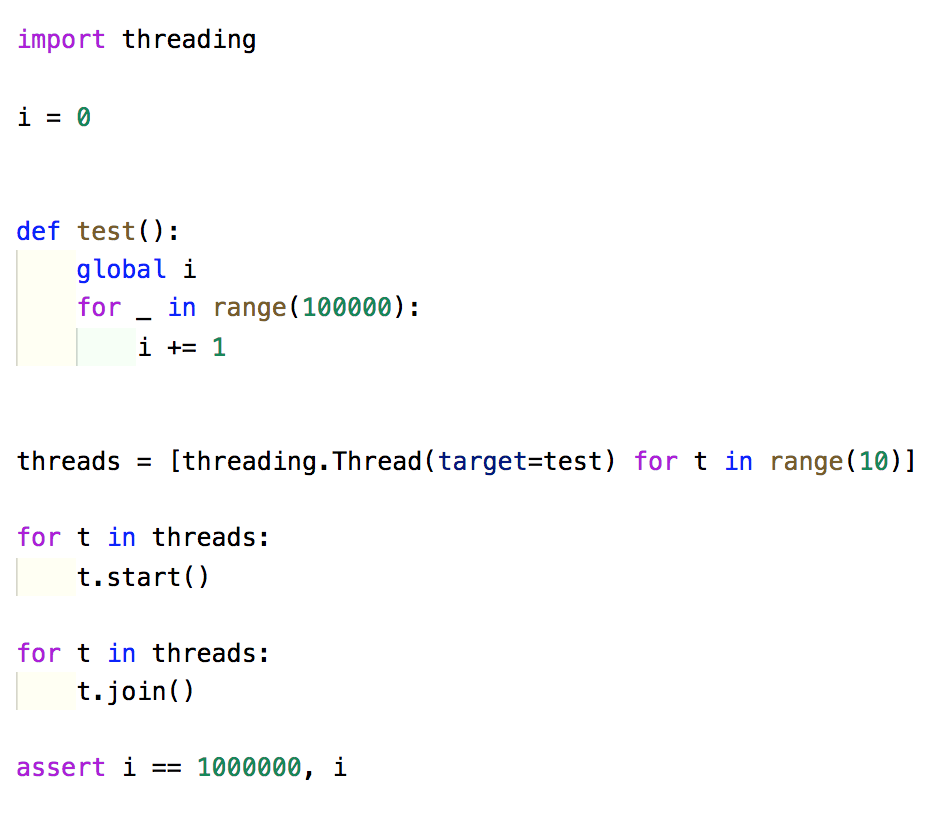


* 1. How do you read a University named “University of Otago” from the database?
  2. Print all students from “University of Otago” (first and last name).

1. Write a test case for the following class:



1. Compare the Template Method Pattern with the Strategy Pattern. What problem is each trying to solve? What is similar about them? What is different about them?
2. The follow multi-threaded code has some potential problems. Can you spot them?



* 1. Describe what you have found.
  2. Fix the identified problems by changing the code. Feel free to strike-through lines to delete and write new ones. There's plenty room for indentation so you don't have to rewrite lines of code to create new blocks.

1. More question types:
   1. Exception handling - try, except and finally
   2. Simplifying existing code using lambda
   3. Queues - producer and consumer
   4. Code smells
   5. End-to-end testing