Note: When I'm showing method signatures, I will never include the `self` parameter. Python programmers recognise that this is required so it can go un-said.

Task 1:

Define a `Car` class. Your Car class should store the make ("Toyota", "Ford"), model ("Corolla", "E150"), year, and colour of the car. Your class needs, at a minimum, a `__init__` and `__str__` method.

Test your class by making sure you can construct a couple of objects from it, and then print them out.

Task 2:

Update your `Car` class from above to add a `current_speed` field. ***Note***: This field should *not* be a parameter in the initialiser – you can do `self.current_speed = 0` as a standalone expression in the initialiser.

Then, add a `go_faster(how_much_faster)` method to the `Car` class. Each time this method is called on a `Car` object, the speed of that car should increase by the amount provided, or 5 if no argument was provided.

Now, add a `brake()` method to your `Car` class. This should slow the car by 5, unless doing so isn't possible.

Finally, add a `how_fast()` method to your `Car` class. This should report (return) the current speed of the `Car` object.

Task 3:

Define a function, `filter_cars_by_colour(list_of_cars, colour)`. This function should take in a list of `Car` objects, and filter it to a *new* list of cars that match the specified colour. That is, you should not *modify* the original list, but should create a new one of only cars of the desired colour.

Note – this is a *function*, not a *method*, and thus should not be part of the `Car` class.

The signature of this function is `(list, string) ---> list`.

Task 3.1:

Modify your `filter_cars_by_colour` function so that if the user doesn't specify a colour, it uses blue.

Task 4: (this one is big – don't worry if you don't get through all of it)

We defined a student class together a few minutes ago, but then used Python's built-in List type as our roster. This kind-of works, but it's not great – it's missing things we might expect, like a waitlist, anything that would prevent a student from joining a class twice, and similar things. Create a new `Course` class that tracks:

- The `name` of the course (a string)
- The course `instructor` (a string)
- The class `enrolment_cap` (an int)
- The currently `enrolled_students` (a list)
- The currently `waitlisted_students` (a list)

In addition to creating a suitable constructor and toString method, add methods to let a student enrol in a class and drop a class. Make sure to consider scenarios about what should happen if a class is full, or if a student drops the course while there are students on a waitlist.