```python
# Created by: Mr. Coxall
# Created on: Sep 2016
# Created for: ICS3U
# This class is used to define a bicycle object

class Bicycle:
    # this class defines a bicycle

    # class variable shared by all instances


    def __init__(self):
        # private fields

        self.__cadence = 0
        self.__speed = 0
        self.__gear = 1

        # public properties
        self.some_property = None

    # properties
    def get_cadence(self):
        # get the cadence property
        return self.__cadence

    def set_cadence(self, new_cadence):
        # set the cadence property
        if new_cadence < 0:
            #this is illegal, so do nothing
            pass
        else:
            self.__cadence_speed_recalculation(new_cadence)
            self.__cadence = new_cadence

    def get_speed(self):
        # get the speed property
        return self.__speed

    def get_gear(self):
        # get the gear property
        return self.__gear

    def set_gear(self, new_gear):
        # set the gear property
        if new_gear < 0 or new_gear > 10:
            # do nothing, this is illegal
```

```
48              pass
49          else:
50              self.__gear_speed_recalculation(new_gear)
51              self.__gear = new_gear
52
53
54      # private methods
55      def __gear_speed_recalculation(self, new_gear):
56          # if you change the gear on a bike, the speed will change
57          old_gear = self.__gear
58
59          if old_gear > new_gear:
60              self.__speed = self.__speed - 5
61          elif old_gear < new_gear:
62              self.__speed = self.__speed + 5
63          else:
64              # same gear!
65              pass
66
67      def __cadence_speed_recalculation(self, new_cadence):
68          # if you change the cadence on a bike, the speed will
            change
69          old_cadence = self.__cadence
70
71          if old_cadence > new_cadence:
72              self.__speed = self.__speed + (1 + (new_cadence-
                old_cadence)/20)
73          elif old_cadence < new_cadence:
74              self.__speed = self.__speed + (1 + (new_cadence-
                old_cadence)/20)
75          else:
76              # same cadence!
77              pass
78
79      # public methods
80
81      def apply_brakes(self, speed_decrease):
82          # decrease the current speed by value passed in
83
84          self.__speed = self.__speed - speed_decrease
85          if self.__speed < 0:
86              self.__speed = 0
87
```

```
New bike1
Set cadence to 40
Current speed: 3
Set gear to 7
Current gear: 7
Current speed: 8
Set cadence to 60
Current cadence: 60
Current speed: 10
Apply breaks by 3
Current speed: 7

New bike2
Set cadence to 40
Current speed: 5
Set gear to 3
Current gear: 3
Current speed: 10
Set cadence to 45
Current cadence: 45
Current speed: 8
Apply break 12
Current speed: 0
```