**Object-oriented Programming Spot Check Electronic Answer Document (EAD)**

Use the following document to record your answers to the assignment spot check. You should then submit the completed EAD to the link provided on Moodle by your teacher.

|  |
| --- |
| **Question 1, Part 3** |
|  |
| **Question 1, Part 4** |
| class Vehicle():  #constructor  def \_\_init\_\_(self,speed,top\_speed,braking,acceleration,amount,driver):  #private  self.\_speed = speed  self.\_top\_speed = top\_speed  self.\_braking = braking  self.\_acceleration = acceleration  self.\_fuel = amount  #public  self.driver = driver  def accelerate(self):  self.\_speed += self.\_acceleration  return self.\_speed    def braking(self):  self.\_speed = self.\_speed - self.\_braking  return self.\_speed  def add\_fuel(self, amount):  self.\_fuel += amount  return self.\_fuel  #test program  def main():  driver = input("Please enter your name: ")  vehicle\_one = Vehicle(5,50,2,6,2,driver)  print(vehicle\_one.driver, vehicle\_one.\_speed, vehicle\_one.\_top\_speed, vehicle\_one.\_braking, vehicle\_one.\_acceleration, vehicle\_one.\_fuel)  print("Accelerating...")  vehicle\_one.accelerate()  print("New speed is: {0}, Top speed is:{1}, Braking is: {2}, Acceleration is: {3}, Fuel is {4}".format(vehicle\_one.\_speed, vehicle\_one.\_top\_speed, vehicle\_one.\_braking, vehicle\_one.\_acceleration, vehicle\_one.\_fuel))  print()  print("Braking...")  vehicle\_one.braking()  print("New speed is: {0}, Top speed is:{1}, Braking is: {2}, Acceleration is: {3}, Fuel is {4}".format(vehicle\_one.\_speed, vehicle\_one.\_top\_speed, vehicle\_one.\_braking, vehicle\_one.\_acceleration, vehicle\_one.\_fuel))  print()  amount = int(input("Please enter an amount of fuel(1-10): "))  vehicle\_one.add\_fuel(amount)    if \_\_name\_\_ == "\_\_main\_\_":  main() |

|  |
| --- |
| **Question 2, Part 1** |
| Encapsulation is the process of making attributes of a Class private |
| **Question 2, Part 2** |
| Some attributes are private to indicate that they should not be changed during the running of the program, or by another programmer. |

|  |
| --- |
| **Question 3, Part 4** |
|  |
| **Question 3, Part 5** |
| from Vehical import \*  class RaceCar(Vehicle):  #constructor  def \_\_init\_\_(self,driver,boost\_energy,race\_number):  super().\_\_init\_\_(5,120,2,6,2,driver)  #private  self.\_boost\_energy = boost\_energy  #public  self.race\_number = race\_number  def accelerate(self):  if self.\_speed < 30:  self.\_speed += self.\_acceleration  elif self.\_speed >= 30 and self.\_speed <= self.\_top\_speed:  self.\_speed += self.\_acceleration \* 1.2  elif self.\_speed < self.\_top\_speed:  self.\_speed += self.\_acceleration \* 0.9  if self.\_speed > self.\_top\_speed:  self.\_speed = self.\_top\_speed  def braking(self):  recovered\_energy = (self.\_speed - (self.\_speed - self.\_braking)) \* 0.1  self.\_speed = self.\_speed - self.\_braking  self.\_boost\_energy += recovered\_energy  return self.\_boost\_energy  def boost(self):  if self.\_boost\_energy > 5:  self.\_boost\_energy -= 0.1  self.\_speed += 10  #test program  def main():  driver = input("Please enter your name: ")  print()  boost\_energy = int(input("Please enter a boost value (1-10): "))  print()  race\_number = int(input("Please enter a car number (1-12): "))  print()  race\_car\_one = RaceCar(driver, boost\_energy, race\_number)  print()  print("Driver: {0}, Car number: {1}, Speed is: {2}, Boost energy level is :{3}, Top speed is:{4}, Braking is: {5}, Acceleration is: {6}, Fuel is {7}".format(race\_car\_one.driver, race\_car\_one.race\_number, race\_car\_one.\_speed, race\_car\_one.\_boost\_energy, race\_car\_one.\_top\_speed, race\_car\_one.\_braking, race\_car\_one.\_acceleration, race\_car\_one.\_fuel))  print()  print("Accelerating...")  race\_car\_one.accelerate()  print("New speed is: {0}, Boost energy level is :{1}, Top speed is:{2}, Braking is: {3}, Acceleration is: {4}, Fuel is {5}".format(race\_car\_one.\_speed, race\_car\_one.\_boost\_energy, race\_car\_one.\_top\_speed, race\_car\_one.\_braking, race\_car\_one.\_acceleration, race\_car\_one.\_fuel))  print()  print("Braking...")  race\_car\_one.braking()  print("New speed is: {0}, Boost energy level is :{1}, Top speed is:{2}, Braking is: {3}, Acceleration is: {4}, Fuel is {5}".format(race\_car\_one.\_speed, race\_car\_one.\_boost\_energy, race\_car\_one.\_top\_speed, race\_car\_one.\_braking, race\_car\_one.\_acceleration, race\_car\_one.\_fuel))  print()  print("Boost...")  race\_car\_one.boost()  print("New speed is: {0}, Boost energy level is :{1}, Top speed is:{2}, Braking is: {3}, Acceleration is: {4}, Fuel is {5}".format(race\_car\_one.\_speed, race\_car\_one.\_boost\_energy, race\_car\_one.\_top\_speed, race\_car\_one.\_braking, race\_car\_one.\_acceleration, race\_car\_one.\_fuel))  print()  amount = int(input("Please enter an amount of fuel(1-10): "))  race\_car\_one.add\_fuel(amount)  if \_\_name\_\_ == "\_\_main\_\_":  main() |

|  |
| --- |
| **Question 4, Part 1** |
| Inheritance is when a child class inherits from the parent class |
| **Question 4, Part 2** |
| In relation to the RaceCar class, the RaceCar inherits the attributes from the Vehicle class in the parent program, the attributes, it inherits are :  self.\_speed = speed  self.\_top\_speed = top\_speed  self.\_braking = braking  self.\_acceleration = acceleration  self.\_fuel = amount |
| **Question 4, Part 3** |
| Polymorphism is when a child class can use parent classes’ methods and can change them to suit the child classes’ individual needs or requirements. This can be useful, as it means that we don’t have to re-write the same code over again every time we create a new child class. |