Quiz 19 - Herança e Polimorfismo I (16/10/2014)

1.

A. A herança permite a reutilização de código;
B. A herança permite ao programador modificar código, comum em múltiplas classes, apenas num local;

2.

```
 class Vehicle {}
class Car extends Vehicle {}
class Bike extends Vehicle {}
```

D. Nenhuma das anteriores.
// este troço de código mostra-nos a herança e não o polimorfismo das classes. Neste caso indica-nos que a class Vehicle é a super classe e as classes Bike e Car são classes derivadas.

3. .

```
public class Vehicle {
   void start() {
       System.out.println("Vehicle");
   }
}
public class Car extends Vehicle{ // não é indicado que se trata de uma classe derivada da classe vehicle
   void start(boolean start) {
       System.out.println("Car");
   }
}
public class Bike {
   void start() {
       System.out.println("Bike");
   }
}
public class Garage {
   public static void main(String[] args) {
       Vehicle car = new Car();
       Bike bike = new Bike();
       car.start();
       bike.start();
   }
}
```

Nesta linha de cógigo, `Vehicle car = new Car();` estamos a criar uma variável do tipo Vehicle que referencia um objecto do tipo Car, logo só olhamos para ele enquanto um Vehicle, assim quando invocado, imprime "Vehicle" e não "Car".
Na linha de código `Bike bike = new Bike();`estamos a criar uma variável do tipo Bike que referencia um objecto do tipo Bike, assim quando invocado, imprime "Bike".

C. Vehicle
   Bike

Quiz 19 - Herança e Polimorfismo I (16/10/2014)

4.

```
public class Vehicle {
       void start() {
          System.out.println("Vehicle");
       }
}
public class Car { // não é indicado que se trata de uma classe derivada da classe vehicle
       void start() {
          System.out.println("Car");
       }
}
public class Bike {
       void start() {
          System.out.println("Bike");
       }
}
public class Garage {
       public static void main(String[] args) {
          Vehicle car = new Car();    //dá erro de compilação porque a variável e o
objecto não são de tipos compatíveis
          Vehicle bike = new Bike();//dá erro de compilação porque a variável e o objecto
não são de tipos compatíveis
          car.start();
       }
}
```

D. Erro de compilação.

5.

B. O método deve ser um *override* de um método da classe base;

//Definimos Polimorfismo como um princípio a partir do qual as classes derivadas de uma única classe base são capazes de invocar os métodos que, embora apresentem a mesma assinatura, comportam-se de maneira diferente para cada uma das classes derivadas.

C. A classe desse método deve estender a classe base que tem o método *overriden*;

O polimorfismo só é válido entre as superclasses e as suas classes derivadas.