A. ¿Cuál es el peso del carro1? Argumente su respuesta.
El carro1 fue instanciado con la siguiente línea de código:
carro1 = Carro("Tracker")
En el constructor de la clase Carro, el parámetro peso tiene un valor por defecto de 1 tonelada:
def __init__(self, referencia, peso=1, marca="Ford", motor=None):
Como no se pasa un valor explícito para el parámetro peso al crear carro1, se utiliza el valor por defecto, que es 1 tonelada.
Respuesta: El peso del carro1 es 1 tonelada.
B. ¿Cuál es el motor del carro3? Argumente su respuesta.
El carro3 fue instanciado de la siguiente forma:
carro3 = Carro("Picanto", 2, "Kia")
En este caso, no se especifica un valor para el parámetro motor, por lo que el valor por defecto de motor es None:
def __init__(self, referencia, peso=1, marca="Ford", motor=None):
Respuesta: El motor del carro3 es None.

C. ¿Cuál es la marca del carro0 (En caso de que este no estuviera comentado en la línea 30)? Justifique lo que sucede.
Si descomentamos la línea donde se crea carro0:
carro0 = Carro()
El valor de la marca para carro0 sería el valor por defecto, ya que no se pasa un valor explícito para el parámetro marca. El valor por defecto de marca en el constructor es "Ford".
def __init__(self, referencia, peso=1, marca="Ford", motor=None):
Respuesta: La marca de carro0 sería "Ford".

D. ¿Qué imprime la línea 35? Justifique.
La línea 35 es la siguiente:
print(Carro.carro_mas_pesado([carro1, carro2, carro3]))
El método carro_mas_pesado recibe una lista de carros como argumento. Este método compara los pesos de los carros para encontrar el carro más pesado. Vamos a ver los pasos del método:
Paso 1: Inicializa aux_ref y aux_peso con los valores del primer carro (carro1).
aux_ref = "Tracker"
aux_peso = 1 (peso de carro1)
Paso 2: Compara el peso de carro2 (peso = 3) con el valor de aux_peso (1).
Como 3 > 1, actualiza:
aux_ref = "Sandero Stepway"
aux_peso = 3
Paso 3: Compara el peso de carro3 (peso = 2) con el valor de aux_peso (3).
Como 2 no es mayor que 3, no se actualiza nada.
Finalmente, el carro más pesado es el carro con referencia "Sandero Stepway", que es el carro2.
Respuesta: La línea 35 imprime:
Sandero Stepway

E. ¿Qué imprime la línea 35 después del cambio en el inicializador? Argumente su respuesta.
El cambio propuesto en el inicializador de la clase Carro es:
def __init__(self, referencia, peso=4, marca="Ford", motor=None):

Ahora, el valor por defecto de peso es 4 en lugar de 1.
Esto afectará a la creación de los objetos carro1, carro2 y carro3:
carro1: El valor de peso será ahora 4 (porque no se pasa un valor explícito).
carro2: El peso sigue siendo 3 porque se pasa explícitamente un valor para peso.
carro3: El valor de peso será 4 (porque no se pasa un valor explícito).
Los pesos de los carros serán:
carro1: 4 toneladas
carro2: 3 toneladas
carro3: 4 toneladas
Al ejecutar el método carro_mas_pesado después del cambio:
Paso 1: Inicializa aux_ref y aux_peso con los valores del primer carro (carro1).
aux_ref = "Tracker"
aux_peso = 4 (peso de carro1)
Paso 2: Compara el peso de carro2 (peso = 3) con el valor de aux_peso (4).
Como 3 no es mayor que 4, no se actualiza nada.

Paso 3: Compara el peso de carro3 (peso = 4) con el valor de aux_peso (4).
Como los pesos son iguales, no se actualiza nada.
Finalmente, el carro más pesado será carro1 o carro3 (ya que ambos tienen el mismo peso de 4 toneladas).
Como el primer carro en la lista con ese peso es carro1, se devolverá su referencia.
Respuesta: Después del cambio, la línea 35 imprimirá:
Tracker

F. ¿Cómo modificar el inicializador de Carro, para que pueda recibir un número indefinido de parámetros?
Para permitir que el constructor de la clase Carro reciba un número indefinido de parámetros, se puede usar el parámetro especial *args en el constructor. Esto permite que se pase cualquier cantidad de argumentos adicionales, los cuales serán almacenados en una tupla.
El cambio sería:

```
def __init__(self, referencia, *args):
    self._referencia = referencia
    self._peso = 4  # Valor por defecto
    self._marca = "Ford"  # Valor por defecto
    self._motor = None  # Valor por defecto

    if len(args) > 0:
        self._peso = args[0]  # Si se pasa un peso, lo asigna
    if len(args) > 1:
        self._marca = args[1]  # Si se pasa una marca, la asigna
    if len(args) > 2:
        self._motor = args[2]  # Si se pasa un motor, lo asigna
```

xplicación:
*args captura cualquier número de argumentos adicionales y los almacena en una tupla llamada args.
El constructor comprueba cuántos elementos hay en args y asigna los valores correspondientes a peso, marca, y motor si se pasan.
De esta forma, el constructor puede manejar cualquier número de parámetros adicionales, sin necesidad de especificar todos los posibles en la definición del constructor.