# DMP02 FAQs

1. What is the purpose of the keyword **self** in python?

   The **self**-keyword represents the instance of the class. Methods and attributes defined within the class recognize which instance of the class is accessing them using the **self**-keyword. See this for more information on it.

2. What is the difference between class and instance attributes?

   An instance attribute is a variable that belongs to a particular object (instance of a class). This variable is only accessible using the object to which it belongs to. A class attribute, on the other hand, is a variable that belongs to a class rather than a particular object (instance of that class). It is shared between all the objects of this class and it is defined outside the **__init__()** constructor.

3. What is the interpretation of "double underscore" in a method name inside a class?

   **D**ouble **under**score methods are called the dunder methods. These methods let you emulate the behavior of built-in types. The __init__() is called automatically as soon as you instantiate an object.

4. Is **__init__()** function optional in a class definition?

   Yes, we can opt to not have this method. The implementation of the **__init__()** depends on the objective of the class. Usually, it works as a constructor of the class. If we want to perform certain operations as soon as the class object is created, we do it using the **__init__()** method.

5. Are methods and variables defined in the parent class available in the child class?

   Yes, all the methods and variables of the parent class are available to the child class. Depending on the requirement, we can override methods that are inherited from the parent class and alter its behaviour.