**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **23-06-2020** | **Name:** | **Rohan shetty** |
| **Course:** | **python** | **USN:** | **4al17ec079** |
| **Topic:** | **Object oriented programming** | **Semester & Section:** | **6th & ‘B’** |
| **GitHub Repository:** | **rohan-shetty-online-courses** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report:**  **Object oriented programming:**   * Python is a multi-paradigm programming language. Meaning, it supports different programming approach. * One of the popular approach to solve a programming problem is by creating objects. This is known as Object-Oriented Programming (OOP).   An object has two characteristics:   * attributes * behavior   **Class:**   * A class is a blueprint for the object. * We can think of class as an sketch of a parrot with labels. It contains all the details about the name, colors, size etc. Based on these descriptions, we can study about the parrot. Here, parrot is an object. * The example for class of parrot can be:   #empty class  class parrot:  pass   * class Parrot:   # class attribute  species = "bird"  # instance attribute  def \_\_init\_\_(self, name, age):  self.name = name  self.age = age  # instantiate the Parrot class  blu = Parrot("Blu", 10)  woo = Parrot("Woo", 15)  # access the class attributes  print("Blu is a {}".format(blu.\_\_class\_\_.species))  print("Woo is also a {}".format(woo.\_\_class\_\_.species))  # access the instance attributes  print("{} is {} years old".format( blu.name, blu.age))  print("{} is {} years old".format( woo.name, woo.age))  **output:**  Blu is a bird  Woo is also a bird  Blu is 10 years old  Woo is 15 years old |