**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **25/5/2020** | **Name:** | **Chandana.R** |
| **Course:** | **DSP** | **USN:** | **4AL16EC017** |
| **Topic:** | Fourier series and Fourier transform, Hilbert transform, Fourier series using mat lab and python and Gibbs phenomenon using mat lab. | **Semester & Section:** | **8th “A”** |
| **Github Repository:** | **Chandana-shaiva** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report –**  A Fourier transform (FT) is a mathematical transform which decomposes a function (often a function of time, or a signal) into its constituent frequencies, such as the expression of a musical chord in terms of the volumes and frequencies of its constituent notes. The term Fourier transform refers to both the frequency domain representation and the mathematical operation that associates the frequency domain representation to a function of time.  Fourier series is a periodic function composed of harmonically related sinusoids, combined by a weighted summation. With appropriate weights, one cycle (or period) of the summation can be made to approximate an arbitrary function in that interval (or the entire function if it too is periodic). As such, the summation is a synthesis of another function. The discrete-time Fourier transform is an example of Fourier series. The process of deriving the weights that describe a given function is a form of Fourier analysis. For functions on unbounded intervals, the analysis and synthesis analogies are Fourier transform and inverse transform.  Examples for Fourier series.       * Gibbs phenomenon, discovered by Henry Wilbraham (1848) and rediscovered by J. Willard Gibbs (1899), is the peculiar manner in which the Fourier series of a piecewise continuously differentiable periodic function behaves at a jump discontinuity. * The nth partial sum of the Fourier series has large oscillations near the jump, which might increase the maximum of the partial sum above that of the function itself. * The overshoot does not die out as n increases, but approaches a finite limit.This sort of behavior was also observed by experimental physicists, but was believed to be due to imperfections in the measuring apparatus. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:25/5/2020** |  | **Name:CHANDANA.R** |  | |
| **Course: Python** |  | **USN:4AL16EC017** |  | |
| **Topic:** |  | **Semester & Section:** | **8th “A”** | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of sessioon**  C:\Users\akash\Pictures\Screenshots\Screenshot (119).png | | | |
| **Report –**  Object Oriented Programming  Class  User defined objects are created using the class keyword. The class is a blueprint that defines the nature of a future object. From classes we can construct instances. An instance is a specific object created from a particular class. For example, above we created the object lst which was an instance of a list object.  # Create a new object type called Sample  class Sample:  pass  # Instance of Sample  x = Sample()  print(type(x))  <class '\_\_main\_\_.Sample'>  By convention we give classes a name that starts with a capital letter. Note how x is now the reference to our new instance of a Sample class. In other words, we instantiate the Sample class.  Inside of the class we currently just have pass. But we can define class attributes and methods.  An attribute is a characteristic of an object. A method is an operation we can perform with the object.  Homework Assignment  Problem 1  Fill in the Line class methods to accept coordinates as a pair of tuples and return the slope and distance of the line.  class Line(object):    def \_\_init\_\_(self,coor1,coor2):  self.coor1 = coor1  self.coor2 = coor2    def distance(self):  x1,y1 = self.coor1  x2,y2 = self.coor2  return ((x2-x1)\*\*2 + (y2-y1)\*\*2)\*\*0.5    def slope(self):  x1,y1 = self.coor1  x2,y2 = self.coor2  return (y2-y1)/(x2-x1)  coordinate1 = (3,2)  coordinate2 = (8,10)  li = Line(coordinate1,coordinate2)  In [3]:  li.distance()  Out:  9.433981132056603  li.slope()  Out:  1.6  Problem 2  Fill in the class  class Cylinder:    def \_\_init\_\_(self,height=1,radius=1):  self.height = height  self.radius = radius    def volume(self):  return self.height\*3.14\*(self.radius)\*\*2  def surface\_area(self):  top = 3.14 \* (self.radius)\*\*2  return (2\*top) + (2\*3.14\*self.radius\*self.height)  c = Cylinder(2,3)  c.volume()  Out:  56.52  c.surface\_area()  Out:  94.2 | | | |