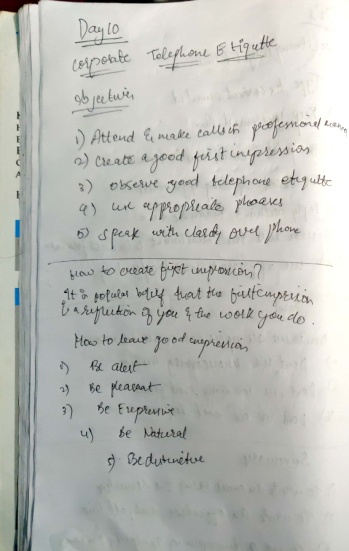
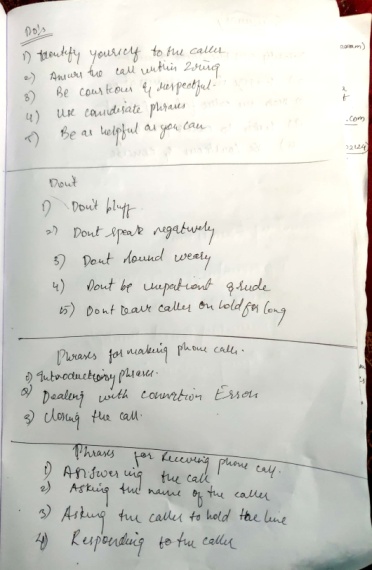
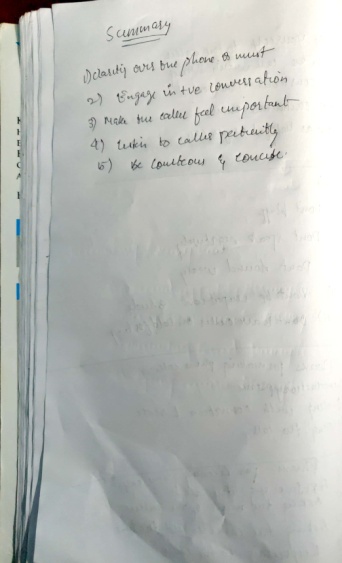
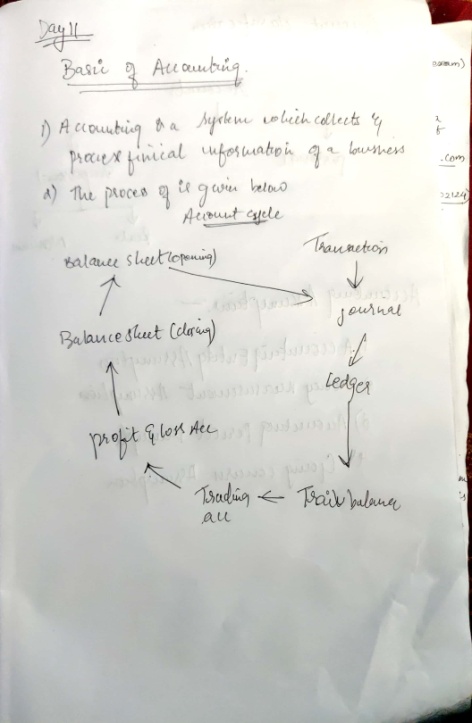
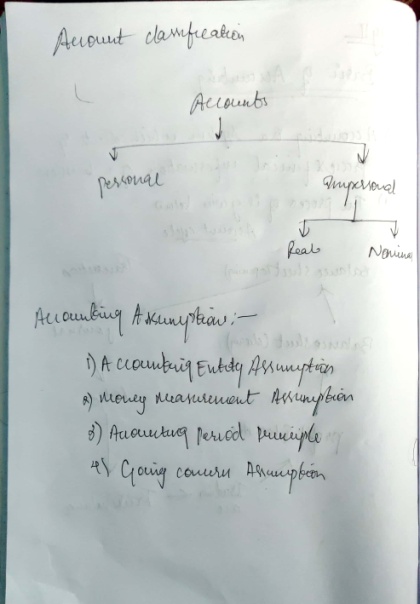
**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **21-May-2020** | **Name:** | **Russell D’souza** |
| **Course:** | **TCS iON** | **USN:** | **4AL15EC023** |
| **Topic:** | **Corporate telephone etiquette,accounting fundamentals,foundation skills in IT** | **Semester & Section:** | **8th sem & ‘A’ section** |
| **Github Repository:** | **Russell1005** |  |  |

|  |
| --- |
| **MORNING SESSION DETAILS** |
| **Image of session** |

****



**DAILY ASSESSMENT FORMAT**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **Date:** | **21-5-2020** | **Name:** | **Russell D’souza** | | **Course:** | **Python programming** | **USN:** | **4AL15EC023** | | **Topic:** | **Data analysis with panda** | **Semester & Section:** | **8th A** | | **Github Repository:** | **Russell1005** |  |  | |
|  |
|  |
| **AFTERNOON SESSION DETAILS** | |
| **Image of session** | |
| Pandas is the most popular python library that is used for data analysis. It provides highly optimized performance with back-end source code is purely written in C or Python.  We can analyze panda by :  1)SERIES  2) DATAFRAME  **Series:**  Series is one dimensional(1-D) array defined in pandas that can be used to store any data type.  # Program to Create series with scalar values  Data =[1, 3, 4, 5, 6, 2, 9] # Numeric data  # Creating series with default index values  s = pd.Series(Data)  # predefined index values  Index =['a', 'b', 'c', 'd', 'e', 'f', 'g']  # Creating series with predefined index values  si = pd.Series(Data, Index)  **Output**:  https://media.geeksforgeeks.org/wp-content/uploads/panda1.png  Panda is a Python library that provides extensive means for data analysis. Data scientists often work with data stored in table formats like .csv, .tsv, or .xlsx. Pandas makes it very convenient to load, process, and analyze such tabular data using SQL-like queries. In conjunction with Matplotlib and Seaborn, Pandas provides a wide range of opportunities for visual analysis of tabular data.  The main data structures in Pandas are implemented with **Series** and **DataFrame** classes. The former is a one-dimensional indexed array of some fixed data type. The latter is a two-dimensional data structure - a table - where each column contains data of the same type. You can see it as a dictionary of Series instances. DataFrames are great for representing real data: rows correspond to instances (examples, observations, etc.), and columns correspond to features of these instances.  In [1]:  import numpy as np  import pandas as pd  pd.set\_option("display.precision", 2) Understand the basic Pandas data structures Pandas has two core data structures used to store data: TheSeries and the DataFrame*.* Series The series is a one-dimensional array-like structure designed to hold a single array (or ‘column’) of data and an associated array of data labels, called an index. We can create a series to experiment with by simply passing a list of data, let’s use numbers in this example:  Copy contents  **import** pandas **as** pd  my\_series = pd.Series([4.6, 2.1, -4.0, 3.0])  **print**(my\_series)  The output should be:  Copy contents  0 4.6  1 2.1  2 -4.0  3 3.0  dtype: float64  Note that printing out our Seriesobject prints out the values and the index numbers. If we just wanted the values, we can add to our script the following line:  Copy contents  **print**(my\_series.values)  Which in addition will print:  Copy contents  array([ 4.6, 2.1, -4. , 3. ])  For a lot of applications, a plain old Series is probably not a lot of use, but it is the core component of the Pandas workhorse, the DataFrame, so it’s useful to know about. DataFrames The DataFrame represents tabular data, a bit like a spreadsheet. DataFrames are organised into colums (each of which is a Series), and each column can store a single data-type, such as floating point numbers, strings, boolean values etc. DataFrames can be indexed by either their row or column names. (They are similar in many ways to R’s data.frame.)  We can create a DataFrame in Pandas from a Python dictionary, or by loading in a text file containing tabular data. First we are going to look at how to create one from a dictionary.  **Setup**  Let’s create a pandas DataFrame with 5 columns and 1000 rows:   * a1 and a2 have random samples drawn from a normal (Gaussian) distribution, * a3 has randomly distributed integers from a set of (0, 1, 2, 3, 4), * y1 has numbers spaced evenly on a log scale from 0 to 1, * y2 has randomly distributed integers from a set of (0, 1).   mu1, sigma1 **=** 0, 0.1 mu2, sigma2 **=** 0.2, 0.2 n **=** 1000df **=** pd**.**DataFrame(  {  "a1": pd**.**np**.**random**.**normal(mu1, sigma1, n),  "a2": pd**.**np**.**random**.**normal(mu2, sigma2, n),  "a3": pd**.**np**.**random**.**randint(0, 5, n),  "y1": pd**.**np**.**logspace(0, 1, num**=**n),  "y2": pd**.**np**.**random**.**randint(0, 2, n),  } )  Readers with Machine Learning background will recognize the notation where a1, a2 and a3 represent attributes and y1 and y2 represent target variables. In short, Machine Learning algorithms try to find patterns in the attributes and use them to predict the unseen target variable — but this is not the main focus of this blog post. The reason that we have two target variables (y1 and y2) in the DataFrame (one binary and one continuous) is to make examples easier to follow.  We reset the index, which adds the index column to the DataFrame to enumerates the rows.  df**.**reset\_index(inplace**=**True)  https://miro.medium.com/max/1624/1*mUL5Q4n0AIsAqVH7sdpOcw.png | |