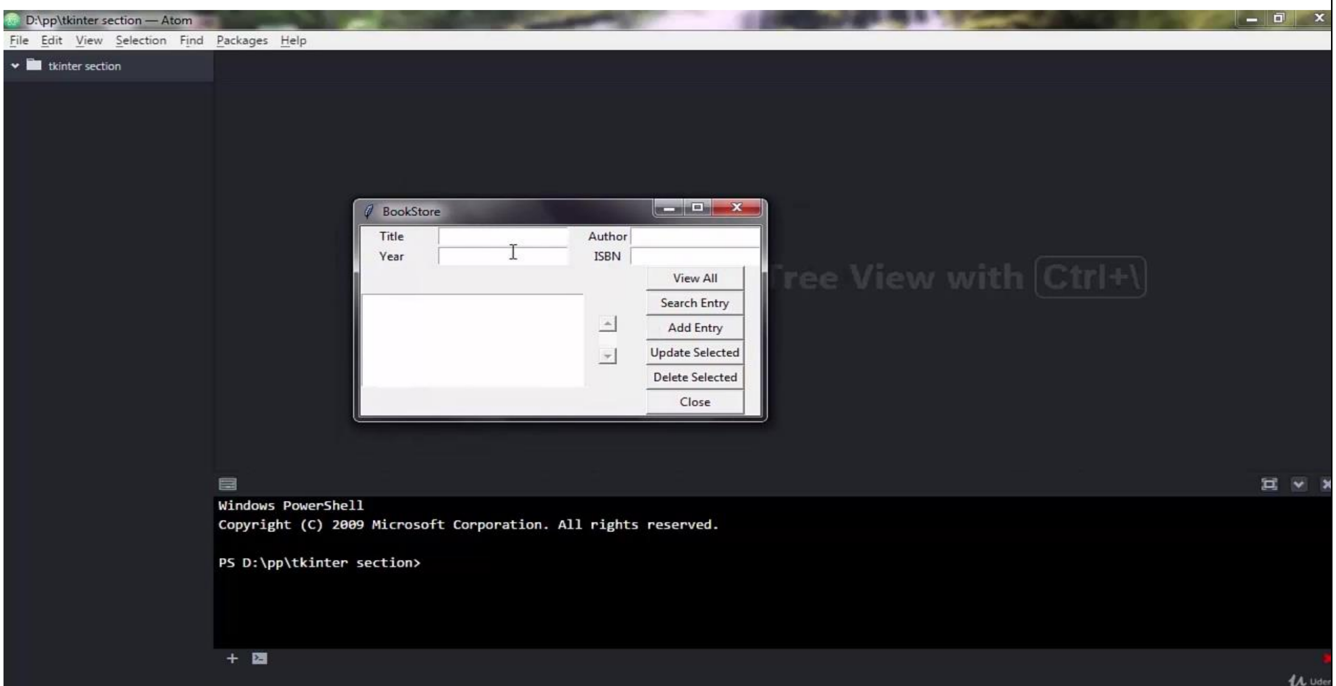
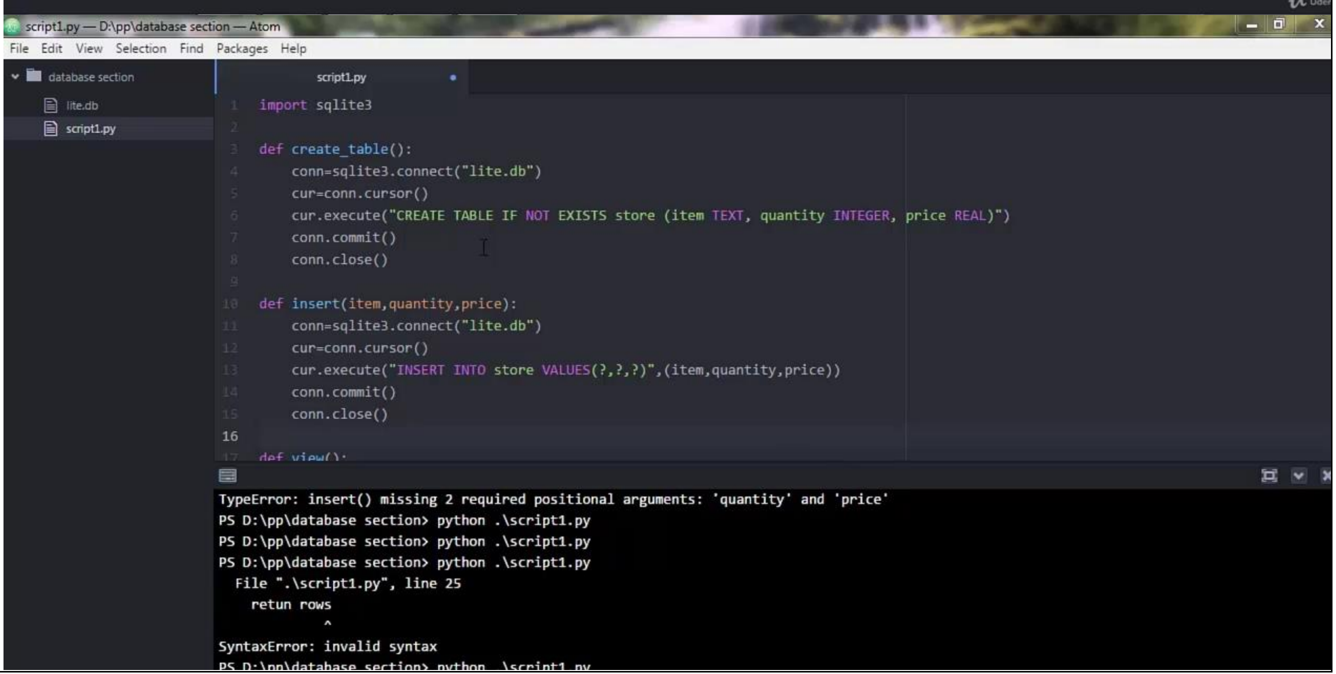
DAILY ASSESSMENT FORMAT

Date:	26th May 2020	**Name:**	Soundarya NA
Course:	UDEMY	**USN:**	4AL16EC077
Topic:	PYTHON: Graphical user interface with Tkinter Interfacing with Databases	**Semester & Section:**	8th - B

AFTERNOON SESSION DETAILS
Image of session





Report:

Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. We would look this option in this chapter.

Tkinter Programming:

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps –

- Import the Tkinter module
- Create the GUI application main window
- Add one or more of the above-mentioned widgets to the GUI application
- Enter the main event loop to take action against each event triggered by the user

E.g.:

```
#!/usr/bin/python
import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
top.mainloop()
```

Code:

```
import tkinter
window = tkinter.Tk()
# to rename the title of the window
window.title("GUI")
# pack is used to show the object in the window
label = tkinter.Label(window, text = "Welcome to DataCamp's Tutorial on Tkinter!").pack()
window.mainloop()
```

Geometry Management:

All widgets in Tkinter have some geometry measurements. These geometry measurements allow you to organize the widgets and throughout the parent frames or parent widget area.

One of the geometry management classes, i.e., pack(), has already been covered here.

For this purpose, Tkinter provides you with three main geometry manager classes:

- pack(): It organizes the widgets in a block manner, and the complete available width is occupied by it. It's a conventional method to show the widgets in the window.
- grid(): It organizes the widgets in a table-like structure. You will learn about it in detail later in this tutorial.
- place(): Its purpose is to place the widgets at a specific position as instructed by the user in the parent widget.

The Python standard for database interfaces is the Python DB-API. Most Python database interfaces adhere to this standard.

Here is the list of available Python database interfaces: Python Database Interfaces and APIs. You must download a separate DB API module for each database you need to access. For example, if you need to access an Oracle database as well as a MySQL database, you must download both the Oracle and the MySQL database modules.

The DB API provides a minimal standard for working with databases using Python structures and syntax wherever possible. This API includes the following −

- Importing the API module.
- Acquiring a connection with the database.
- Issuing SQL statements and stored procedures.
- Closing the connection

Code:

Database Connection:

```
#!/usr/bin/python

import MySQLdb
```

```
# Open database connection
db = MySQLdb.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method
cursor = db.cursor()

# execute SQL query using execute() method.
cursor.execute("SELECT VERSION()")

# Fetch a single row using fetchone() method.
data = cursor.fetchone()
print "Database version : %s " % data

# disconnect from server
db.close()
```

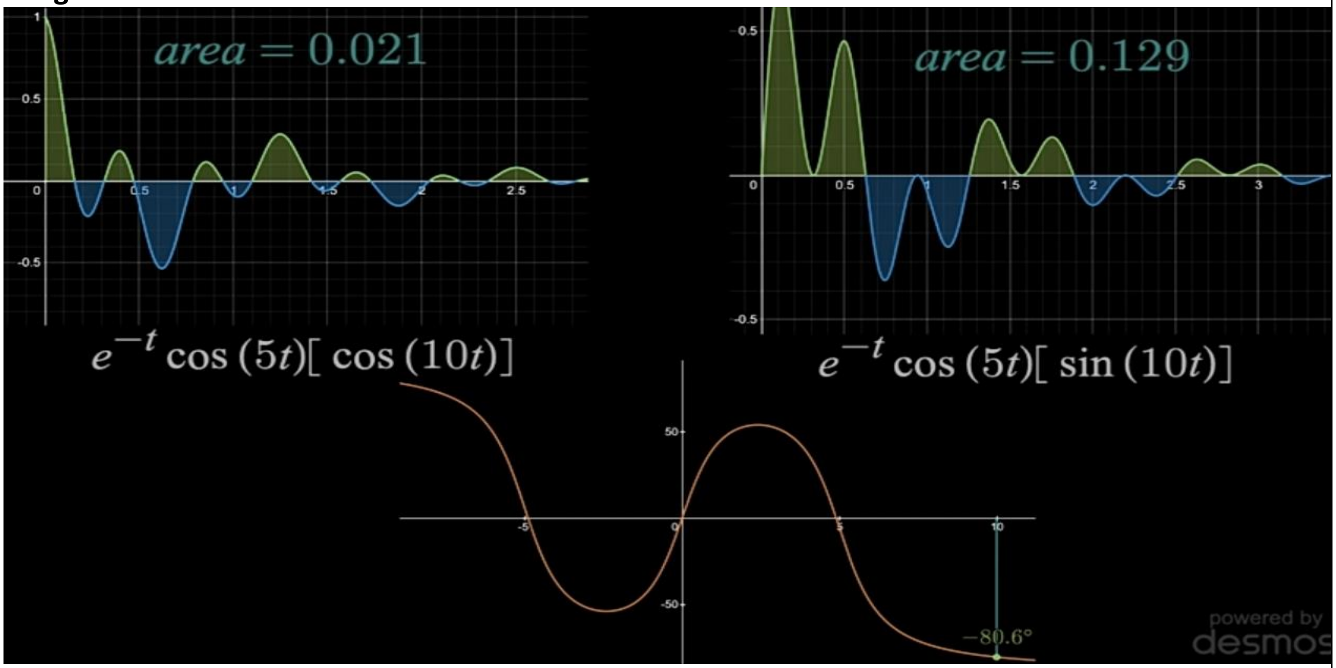
Creating Database table:

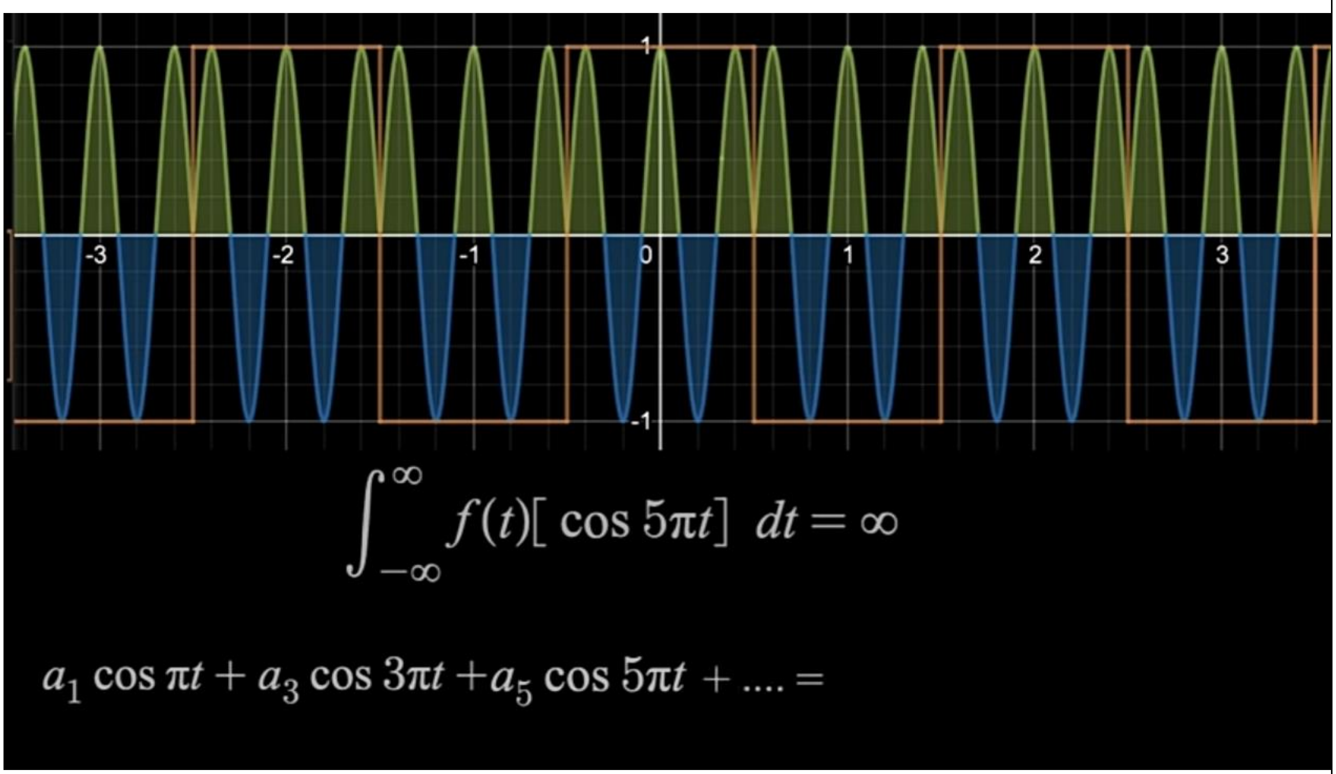
```
#!/usr/bin/python

import MySQLdb

# Open database connection
db = MySQLdb.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method
cursor = db.cursor()

# Drop table if it already exist using execute() method.
cursor.execute("DROP TABLE IF EXISTS EMPLOYEE")

# Create table as per requirement
```

```
sql = """CREATE TABLE EMPLOYEE (
        FIRST_NAME  CHAR(20) NOT NULL,
        LAST_NAME  CHAR(20),
        AGE INT,
        SEX CHAR(1),
        INCOME FLOAT )"""

cursor.execute(sql)

# disconnect from server
db.close()
```

Date:	26th May 2020	**Name:**	Soundarya NA
Course:	DSP	**USN:**	4AL16EC077
Topic:	Digital Signal Processing	**Semester & Section:**	8th - B

Image:





Report:

The problem:

Given the Fourier transform of a general function, find the Fourier transform of its derivative. Use this result to find the Fourier transform of a window function out of the Fourier transform of an antisymmetric pair of delta functions.

The solution:

We are given the following:

$F.T[f(t)] = F(\omega)$,

and we take into account that:

$\lim\ t \rightarrow \pm\infty\ f(t) \rightarrow 0$.

We begin by writing explicitly:

$F.T[f\ 0\ (t)] = Z\ \infty\ -\infty\ f\ 0\ (t)ei\omega tdt$.

Integration by parts gives us:

$f(t)ei\omega t|\ +\infty\ -\infty\ -\ i\omega\ Z\ \infty\ -\infty\ f(t)ei\omega tdt = -i\omega F(\omega)$,

and we get:

$F.T[f\ 0\ (t)] = -i\omega F(\omega)$.

Let us represent a window function in the region $[-a, a]$ as a sum of two step functions:

$\Pi(t) = \Theta(t + a) - \Theta(t - a)$.

We also note that $\delta(t + a) = d\ dt\Theta(t + a)$.

Now, using what we have derived earlier we find:

$F.T[\delta(t + a) - \delta(t - a)] = F.T[\ d\ dt\Theta(t + a) - d\ dt\Theta(t - a)] = -i\omega F.T[\Theta(t + a) - \Theta(t - a)] = -i\omega F.T[\Pi(t)]$, and,

$F.T[\delta(t + a) - \delta(t - a)] = Z\ \infty\ -\infty\ \delta(t + a)ei\omega tdt - Z\ \infty\ -\infty\ \delta(t - a)ei\omega tdt = e{-}i\omega a - e\ i\omega a = -2i\sin(\omega a)$.

$\Rightarrow F.T[\Pi(t)] = 2a\mathrm{sinc}(\omega a)$

In the remainder of the course, we'll study several methods that depend on analysis of images or reconstruction of structure from images:

- Light microscopy (particularly fluorescence microscopy)
- Electron microscopy (particularly for single-particle reconstruction)
- X-ray crystallography

The computational aspects of each of these methods involve Fourier transforms and convolution. These concepts are also important for:

- Some approaches to ligand docking (and protein-protein docking)
- Fast evaluation of electrostatic interactions in molecular dynamics
- (You're not responsible for these additional applications)

Calculate the Laplace Transform using Matlab Calculating the Laplace F(s) transform of a function f(t) is quite simple in Matlab.

First you need to specify that the variable t and s are symbolic ones. This is done with the command >> syms t s Next you define the function f(t).

The actual command to calculate the transform is >> F=laplace(f,t,s) To make the expression more readable one can use the commands, simplify and pretty. here is an example for the function

f(t), t t tf ete 2 2 25.15.325.1)

```
>> syms t s
>> f=-1.25+3.5*t*exp(-2*t)+1.25*exp(-2*t);
>> F=laplace(f,t,s)
F =
-5/4/s+7/2/(s+2)^2+5/4/(s+2)
>> simplify(F)
ans =
(s-5)/s/(s+2)^2
>> pretty(ans)
s - 5 ---------- 2 s (s + 2)
```

which corresponds to

F(s), 2)2( )5( )(+ − = ss s sF

Alternatively, one can write the function f(t) directly as part of the laplace command:

```
>>F2=laplace(-1.25+3.5*t*exp(-2*t)+1.25*exp(-2*t))
```