CA Lab-V LAB on Python Programming

Assignment 1. Develop programs to understand the control structures of python

```python
print("Select your Choice:")

print("1. For Loop")

print("2. While Loop")

choice = int( input() )

if( choice == 1 ):

    for x in range(1,8,2):

        print(x)

elif( choice == 2 ):

  count=1;

  while( count< 8):

    print(count)

    count+=2;

else:

    print( "Ok" )
```

Output: -

Select your Choice:

1. For Loop

2. While Loop

1

1

3

5

7

Assignment 2. Develop programs to learn different types of structures (list, dictionary, tuples) in python

```python
print("Select your Choice:")
print("1. List")
print("2. Dictionary ")
print("3. Tuple")
choice = int( input() )
if( choice == 1 ):
    my_list = [1, 2, 3, 'example', 3.132] #creating list with data
    print(my_list)
    my_list.append([555, 12]) #add as a single element
    print(my_list)
    my_list.extend([234, 'more_example']) #add as different elements
    print(my_list)
    my_list.insert(1, 'insert_example') #add element
    print(my_list)
    del my_list[5] #delete element at index 5
    print(my_list)
    my_list.remove('example') #remove element with value
    print(my_list)
    a = my_list.pop(1) #pop element from list
    print('Popped Element: ', a, ' List remaining: ', my_list)
    my_list.clear() #empty the list
    print(my_list)
    my_list = [1, 2, 3, 'example', 3.132, 10, 30]
    for element in my_list: #access elements one by one
        print(element)
    print(my_list) #access all elements
    print(my_list[3]) #access index 3 element
    print(my_list[0:2]) #access elements from 0 to 1 and exclude 2
    print(my_list[::-1]) #access elements in reverse
    my_list = [1, 2, 3, 10, 30, 10]
    print(len(my_list)) #find length of list
```

```python
        print(my_list.index(10)) #find index of element that occurs first
        print(my_list.count(10)) #find count of the element
        print(sorted(my_list)) #print sorted list but not c
    elif( choice == 2 ):
        my_dict = {1: 'Python', 2: 'Java'} #dictionary with elements
        print(my_dict)
        my_dict = {'First': 'Python', 'Second': 'Java'}
        print(my_dict)
        my_dict['Second'] = 'C++' #changing element
        print(my_dict)
        my_dict['Third'] = 'Ruby' #adding key-value pair
        print(my_dict)
        my_dict = {'First': 'Python', 'Second': 'Java', 'Third': 'Ruby'}
        a = my_dict.pop('Third') #pop element
        print('Value:', a)
        print('Dictionary:', my_dict)
        b = my_dict.popitem() #pop the key-value pair
        print('Key, value pair:', b)
        print('Dictionary', my_dict)
        my_dict.clear() #empty dictionary
        print(my_dict)
        my_dict = {'First': 'Python', 'Second': 'Java'}
        print(my_dict['First']) #access elements using keys
        print(my_dict.get('Second'))
        my_dict = {'First': 'Python', 'Second': 'Java', 'Third': 'Ruby'}
        print(my_dict.keys()) #get keys
        print(my_dict.values()) #get values
        print(my_dict.items()) #get key-value pairs
    elif( choice == 3 ):
        my_tuple = (1, 2, 3) #create tuple
        print(my_tuple)
        my_tuple2 = (1, 2, 3, 'edureka') #access elements
        for x in my_tuple2:
            print(x)
```

```
print(my_tuple2)
print(my_tuple2[0])
print(my_tuple2[:])
print(my_tuple2[3][4])
my_tuple = (1, 2, 3)
my_tuple = my_tuple + (4, 5, 6) #add elements
print(my_tuple)
my_tuple = (1, 2, 3, ['hindi', 'python'])
my_tuple[3][0] = 'english'
print(my_tuple)
print(my_tuple.count(2))
print(my_tuple.index(['english', 'python']))
else:
    print( "Ok" )
```

Output: -

Select your Choice:

1. List

2. Dictionary

3. Tuple

3

(1, 2, 3)

1

2

3

edureka

(1, 2, 3, 'edureka')

1

(1, 2, 3, 'edureka')

e

(1, 2, 3, 4, 5, 6)

(1, 2, 3, ['english', 'python'])

1

3

Assignment 3. Develop programs to learn concept of functions scoping, recursion and list mutability.

```python
total = 0; # This is global variable. # Function definition is here
def sum( arg1, arg2 ):
    # Add both the parameters and return them."
    total = arg1 + arg2; # Here total is local variable.
    print ("Inside the function local total : ", total)
    return total;


def factorial(x):
    if x == 1:
        return 1
    else:
        return (x * factorial(x-1))


print("Select your Choice:")
print("1. Functions Scoping")
print("2. Recursion")
print("3. List Mutability")
choice = int( input() )
if( choice == 1 ):
    # Now you can call sum function
    sum( 10, 20 );
    print ("Outside the function global total : ", total)
elif( choice == 2 ):
    num = 4
    print("The factorial of", num, "is", factorial(num))
elif( choice == 3 ):
    my_list = [1, 2, 3, 'example', 3.132] #creating list with data
    print(my_list)
    my_list.append([555, 12]) #add as a single element
    print(my_list)
    my_list.extend([234, 'more_example']) #add as different elements
```

```python
    print(my_list)
    my_list.insert(1, 'insert_example') #add element
    print(my_list)
    del my_list[5] #delete element at index 5
    print(my_list)
    my_list.remove('example') #remove element with value
    print(my_list)
    a = my_list.pop(1) #pop element from list
    print('Popped Element: ', a, ' List remaining: ', my_list)
    my_list.clear() #empty the list
    print(my_list)
else:
    print( "Ok" )
```

Output: -

Select your Choice:

1. Functions Scoping

2. Recursion

3. List Mutability

3

[1, 2, 3, 'example', 3.132]

[1, 2, 3, 'example', 3.132, [555, 12]]

[1, 2, 3, 'example', 3.132, [555, 12], 234, 'more_example']

[1, 'insert_example', 2, 3, 'example', 3.132, [555, 12], 234, 'more_example']

[1, 'insert_example', 2, 3, 'example', [555, 12], 234, 'more_example']

[1, 'insert_example', 2, 3, [555, 12], 234, 'more_example']

Popped Element: insert_example List remaining: [1, 2, 3, [555, 12], 234, 'more_example']

[]

Assignment 4. Develop programs to understand object oriented programming using python.

```python
class Parent:          # define parent class
    parentAttr = 100
    def __init__(self):
        print ("Calling parent constructor")
    def parentMethod(self):
        print ("Calling parent method")
    def setAttr(self, attr):
        self.parentAttr = attr
    def getAttr(self):
        print ("Parent attribute :", self.parentAttr)


class Child(Parent): # define child class
    def __init__(self):
        print ("Calling child constructor")
    def childMethod(self):
        print ("Calling child method")


c = Child()
c.childMethod()
c.parentMethod()
c.setAttr(200)
c.getAttr()
```

Output:-

Calling child constructor

Calling child method

Calling parent method

Parent attribute : 200

Assignment 5. Develop programs for data structure algorithms using python – searching, sorting and hash tables.

```python
def LinearSearch(lys, element):

    for i in range (len(lys)):

        if lys[i] == element:

            return i

    return -1

def BinarySearch(lys, val):

    first = 0

    last = len(lys)-1

    index = -1

    while (first <= last) and (index == -1):

        mid = (first+last)//2

        if lys[mid] == val:

            index = mid

        else:

            if val<lys[mid]:

                last = mid -1

            else:

                first = mid +1

    return index

def bubblesort(list):

# Swap the elements to arrange in order

    for iter_num in range(len(list)-1,0,-1):

        for idx in range(iter_num):
```

```python
        if list[idx]>list[idx+1]:

            temp = list[idx]

            list[idx] = list[idx+1]

            list[idx+1] = temp

def merge_sort(unsorted_list):

    if len(unsorted_list) <= 1:

        return unsorted_list

# Find the middle point and devide it

    middle = len(unsorted_list) // 2

    left_list = unsorted_list[:middle]

    right_list = unsorted_list[middle:]

    left_list = merge_sort(left_list)

    right_list = merge_sort(right_list)

    return list(merge(left_list, right_list))

# Merge the sorted halves

def merge(left_half,right_half):

    res = []

    while len(left_half) != 0 and len(right_half) != 0:

        if left_half[0] < right_half[0]:

            res.append(left_half[0])

            left_half.remove(left_half[0])

        else:

            res.append(right_half[0])

            right_half.remove(right_half[0])
```

```python
        if len(left_half) == 0:

            res = res + right_half

        else:

            res = res + left_half

    return res

def insertionSort(arr):

        if (n := len(arr)) <= 1:

                return

        for i in range(1, n):

                key = arr[i]

                # Move elements of arr[0..i-1], that are

                # greater than key, to one position ahead

                # of their current position

                j = i-1

                while j >=0 and key < arr[j] :

                        arr[j+1] = arr[j]

                        j -= 1

                arr[j+1] = key

def selectionSort(array, size):

        for ind in range(size):

                min_index = ind

                for j in range(ind + 1, size):

                        # select the minimum element in every iteration

                        if array[j] < array[min_index]:
```

```python
            min_index = j

        # swapping the elements to sort the array

        (array[ind], array[min_index]) = (array[min_index], array[ind])

print("Select your Choice:")

print("1. Linear Search")

print("2. Binary Search")

print("3. Bubble Sort")

print("4. Merge Sort")

print("5. Insertion Sort")

print("6. Selection Sort")

print("7. Hash Table")

choice = int( input() )

if( choice == 1 ):

    print(LinearSearch([1,2,3,4,5,2,1], 5))

elif( choice == 2 ):

    print(BinarySearch([10,20,30,40,50], 40))

elif( choice == 3 ):

    list = [19,2,31,45,6,11,121,27]

    bubblesort(list)

    print(list)

elif( choice == 4 ):

    unsorted_list = [64, 34, 25, 12, 22, 11, 90]

    print(merge_sort(unsorted_list))
```

```python
elif( choice == 5 ):

    arr = [12, 11, 13, 5, 6]

    insertionSort(arr)

    print(arr)

elif( choice == 6 ):

    arr = [-2, 45, 0, 11, -9,88,-97,-202,747]

    size = len(arr)

    selectionSort(arr, size)

    print(arr)

elif( choice == 7 ):

    # Declare a dictionary

    dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

    # Accessing the dictionary with its key

    print ("dict['Name']: ", dict['Name'])

    print ("dict['Age']: ", dict['Age'])

    dict['Age'] = 8; # update existing entry

    dict['School'] = "DPS School"; # Add new entry

    print ("dict['Age']: ", dict['Age'])

    print ("dict['School']: ", dict['School'])

    dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

    del dict['Name']; # remove entry with key 'Name'

    dict.clear();     # remove all entries in dict

    del dict ;        # delete entire dictionary

    print ("dict['Age']: ", dict['Age'])
```

```
    print ("dict['School']: ", dict['School'])

else:

    print( "Ok" )
```

Output:-

Select your Choice:

1. Linear Search

2. Binary Search

3. Bubble Sort

4. Merge Sort

5. Insertion Sort

6. Selection Sort

7. Hash Table

3

[2, 6, 11, 19, 27, 31, 45, 121]

Assignment 6. Develop programs to learn regular expressions using python.

```python
import re
#Return a list containing every occurrence of "ai":
txt1 = "The rain in Spain"
x1 = re.findall("ai", txt1)
print(x1)


txt2 = "The rain in Spain"
x2 = re.search("\s", txt2)
print("The first white-space character is located in position:", x2.start())


#Split the string at every white-space character:
txt3 = "The rain in Spain"
x3 = re.split("\s", txt3)
print(x3)


#Replace all white-space characters with the digit "9":
txt4 = "The rain in Spain"
x4 = re.sub("\s", "9", txt4)
print(x4)


txt5 = "The rain in Spain"
x5 = re.search("ai", txt5)
print(x5)


#Search for an upper case "S" character in the beginning of a word, and print the word:
txt6 = "The rain in Spain"
x6 = re.search(r"\bS\w+", txt6)
print(x6.group())
```

Output:-

['ai', 'ai']

The first white-space character is located in position: 3

['The', 'rain', 'in', 'Spain']

The9rain9in9Spain

<re.Match object; span=(5, 7), match='ai'>

Spain

Assignment 7. Demonstrate the concept of exception handling using try/except/else Statement, Unified try/except/finally, try/finally Statement, raise Statement, assert Statement, catch multiple specific exceptions

```python
# Python code to illustrate
# working of try()
def divide(x, y):
    try:
        # Floor Division : Gives only Fractional
        # Part as Answer
        result = x // y
    except ZeroDivisionError:
        print("Sorry ! You are dividing by zero ")
    else:
        print("Yeah ! Your answer is :", result)
    finally:
        # this block is always executed
        # regardless of exception generation.
        print('This is always executed')

# Look at parameters and note the working of Program
divide(3, 2)
divide(3, 0)

# A python program to create user-defined exception
# class MyError is derived from super class Exception
class MyError(Exception):

    # Constructor or Initializer
    def __init__(self, value):
        self.value = value

    # __str__ is to print() the value
    def __str__(self):
        return(repr(self.value))
try:
    raise(MyError(3*2))

# Value of Exception is stored in error
except MyError as error:
    print('A New Exception occurred: ', error.value)
x = "hello"
#if condition returns False, AssertionError is raised:
assert x == "goodbye", "x should be 'hello'"
```
Output: -
Yeah ! Your answer is : 1
This is always executed
Sorry ! You are dividing by zero
This is always executed
A New Exception occurred: 6

```
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
<ipython-input-1-dfedc9c43237> in <module>
    38 x = "hello"
    39 #if condition returns False, AssertionError is raised:
---> 40 assert x == "goodbye", "x should be 'hello'"

AssertionError: x should be 'hello'
```

Assignment 8. Demonstrate the concept of String-Based Exceptions, Class-Based Exceptions and Nesting Exception handlers.

```python
#String-Based Exceptions
try:
    print(1 + '3')
except Exception as e:
    error_message = str(e)
    print(error_message)
    print(type(error_message))

#Class-Based Exceptions
class LowAgeError(Exception):
    def __init__(self):
        pass

    def __str__(self):
        return 'The age must be greater than 18 years'

class Employee:
    def __init__(self, name, age):
        self.name = name
        if age < 18:
            raise LowAgeError
        else:
            self.age = age

    def display(self):
        print('The name of the employee: ' + self.name + ', Age: ' + str(self.age) +' Years')

try:
    e1 = Employee('Subhas', 25)
    e1.display()

    e2 = Employee('Anupam', 12)
    e1.display()
except LowAgeError as e:
    print('Error Occurred: ' + str(e))

 #Nested
x = 10
y = 0

try:
    print("outer try block")
    try:
        print("nested try block")
        print(x / y)
    except TypeError as te:
        print("nested except block")
```

```
        print(te)
except ZeroDivisionError as ze:
    print("outer except block")
    print(ze)
```

Output: -
unsupported operand type(s) for +: 'int' and 'str'
<class 'str'>
The name of the employee: Subhas, Age: 25 Years
Error Occurred: The age must be greater than 18 years
outer try block
nested try block
outer except block
division by zero

Assignment 9. Demonstrate implementation of the Anonymous Function Lambda.

 # Finding the area of a triangle

triangle = lambda m,n : 1/2 * m * n

res=triangle(34,24)

print("Area of the triangle: ",res)

Output: -
Area of the triangle: 408.0

Assignment 10. Demonstrate implementation functional programming tools such as filter and reduce

series = [23,45,57,39,1,3,95,3,8,85]
result = filter (lambda m: m > 29, series)
print('All the numbers greater than 29 in the series are :',list(result))

Output: -
All the numbers greater than 29 in the series are : [45, 57, 39, 95, 85]

from functools import reduce
series = [23,5,1,7,45,9,38,65,3]
sum = reduce (lambda m,n: m+n, series)
print('The total sum of all the elements in the list is :',sum)

Output: - The total sum of all the elements in the list is: 196

Assignment 11. Demonstrate the Module Creation, Module usage.

Step1: Click on New-Text File

Step2: Rename as module2.py

Step3:Write following code

```
def show(name):
        print("Hello",name)
```

Step4: Click on New-Python3

Step5: Rename as Assignment11

Step6: Write following code

```
import module2
module2.show("Manoj")
```

Step7: Run

Output: - Hello Manoj

Assignment 12. Demonstrate image insertion in python.

```
from PIL import Image
myImage = Image.open("C:\\Users\\Admin\\Pictures\\d.png");
myImage.show();
```

Output: -

Assignment 13. Demonstrate use of DataFrame method and use of .csv files.

Step1: Upload .csv file
Step2: Write code as follows

```
import pandas as pd
df=pd.read_csv("Ecommerce_Customers.csv")
df.head()
df.tail()
em = df["Email"]
print(em)
print(df.loc[1])
print(df.info())
```

Output:-

Assignment 14. Develop programs to learn GUI programming using Tkinter

```
from tkinter import *
import tkinter
def helloCallBack():
    print("Name=",t1.get())
    i=radio.get()
    if i==1:
        print("Gender=Male")
    else:
        print("Gender=Female")
    j=chk1.get()
    k=chk2.get()
    l=chk3.get()
    str=""
    if j==1:
        str=str+" "+"Red"
    if k==2:
        str=str+" "+"Green"
    if l==3:
        str=str+" "+"Blue"
    print("Color=",str)
    for i in lb1.curselection():
        print(lb1.get(i))


base = Tk()
base.geometry('600x600')
base.title("Registration Form")
l1 = Label(base, text="Enter Name",width=20,font=("bold", 10))
l1.place(x=90,y=55)
t1 = Entry(base)
t1.place(x=220,y=55)

radio=IntVar()
l2= Label(base, text="Select Gender",width=20,font=("bold", 10))
l2.place(x=94,y=93)
r1 = Radiobutton(base, text="Male", value=1,variable=radio)
r1.place(x=220,y=93)
r2 = Radiobutton(base, text="Female", value=2,variable=radio)
r2.place(x=280,y=93)

l3= Label(base, text="Select Class",width=20,font=("bold", 10))
l3.place(x=90,y=123)
lb1 = Listbox(base)
lb1.insert(1, "MCA-I")
lb1.insert(2, "MCA-II")
lb1.insert(3, "MCA-III")
lb1.place(x=230,y=123)
```

```
chk1=IntVar()
chk2=IntVar()
chk3=IntVar()
l4= Label(base, text="Select Color",width=20,font=("bold", 10))
l4.place(x=90,y=300)
c1 = Checkbutton(base,text = "Red",width = 20, onvalue=1,variable=chk1)
c1.place(x=220,y=300)
c2 = Checkbutton(base,text = "Green",width = 20,onvalue=2,variable=chk2)
c2.place(x=340,y=300)
c3 = Checkbutton(base,text = "Blue",width = 20,onvalue=3,variable=chk3)
c3.place(x=460,y=300)

b1 = Button(base, text ="Click", command = helloCallBack)
b1.place(x=190,y=380)
base.mainloop()
```

Output:-

Name= abc
Gender=Male
Color= Red Green
MCA-I

Assignment 15. Create a simple web application using Flask.

from flask import Flask

app = Flask (__name__)

@app.route("/")

@app.route("/home")

def home():

   return "<h2>This Is My Home Page</h2>"

if __name__=='__main__':

   app.run()

OUTPUT:-
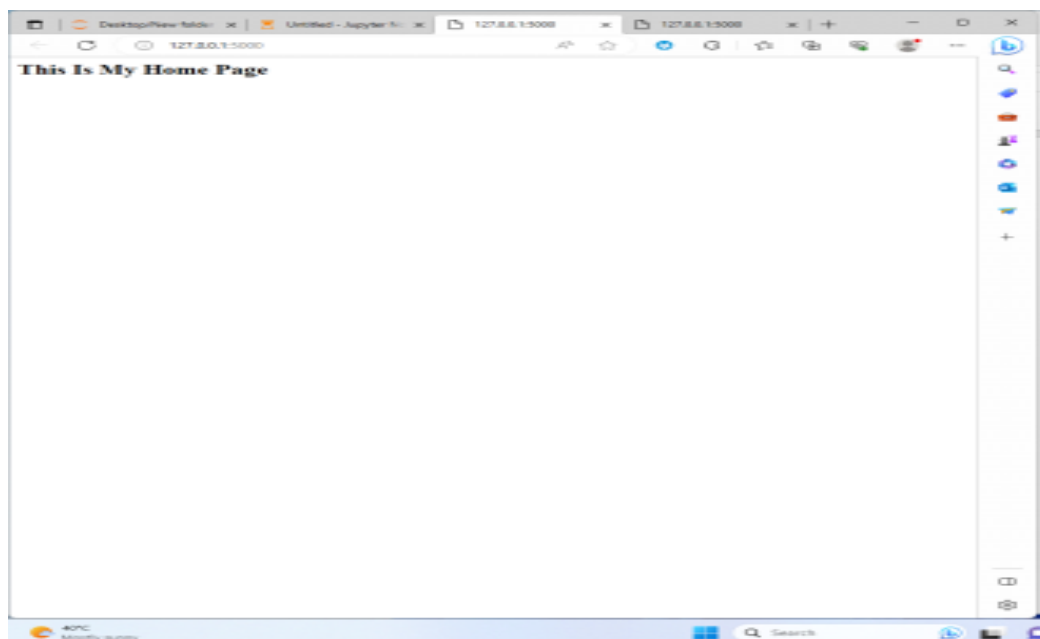
Assignment 16. Create Simple Django Framework

Step1: Open Anaconda Navigator, Click on Environments, Click on Create, Give name Django2
Step2: Tick on Python package and select version 3.7 or above, Click on Create button.
Step3: Click on Home, select Applications on as Django2, Launch Jupyter Notebook.
Step4: Take New Python3-Install following packages one by one and every time restart kernel.
pip install django
pip install django-bootstrap4
pip install django-crispy-forms

```
In [1]:   1 pip install django

Requirement already satisfied: django in c:\programdata\anaconda3\envs\django2\lib\site-packages (3.2.19)Note: you may need to
restart the kernel to use updated packages.
Requirement already satisfied: asgiref<4,>=3.3.2 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django) (3.6.
0)
Requirement already satisfied: pytz in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django) (2023.3)
Requirement already satisfied: sqlparse>=0.2.2 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django) (0.4.4)
Requirement already satisfied: typing-extensions in c:\programdata\anaconda3\envs\django2\lib\site-packages (from asgiref<4,>=
3.3.2->django) (4.1.1)
```

```
In [1]:   1 pip install django-bootstrap4

Requirement already satisfied: django-bootstrap4 in c:\programdata\anaconda3\envs\django2\lib\site-packages (23.1)Note: you may
need to restart the kernel to use updated packages.

Requirement already satisfied: beautifulsoup4>=4.8.0 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django-bo
otstrap4) (4.10.0)
Requirement already satisfied: importlib-metadata<3 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django-boo
tstrap4) (2.1.3)
Requirement already satisfied: Django>=3.2 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django-bootstrap4)
(3.2.19)
Requirement already satisfied: soupsieve>1.2 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from beautifulsoup4>=
4.8.0->django-bootstrap4) (2.3.1)
Requirement already satisfied: pytz in c:\programdata\anaconda3\envs\django2\lib\site-packages (from Django>=3.2->django-bootst
rap4) (2023.3)
Requirement already satisfied: sqlparse>=0.2.2 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from Django>=3.2->dj
ango-bootstrap4) (0.4.4)
Requirement already satisfied: asgiref<4,>=3.3.2 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from Django>=3.2->
django-bootstrap4) (3.6.0)
Requirement already satisfied: typing-extensions in c:\programdata\anaconda3\envs\django2\lib\site-packages (from asgiref<4,>=
3.3.2->Django>=3.2->django-bootstrap4) (4.1.1)
Requirement already satisfied: zipp>=0.5 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from importlib-metadata<3-
>django-bootstrap4) (3.8.0)
```

```
In [1]:   1 pip install django-crispy-forms

Requirement already satisfied: django-crispy-forms in c:\programdata\anaconda3\envs\django2\lib\site-packages (2.0)Note: you ma
y need to restart the kernel to use updated packages.
Requirement already satisfied: django>=3.2 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django-crispy-form
s) (3.2.19)
Requirement already satisfied: sqlparse>=0.2.2 in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django>=3.2->dj
ango-crispy-forms) (0.4.4)
Requirement already satisfied: pytz in c:\programdata\anaconda3\envs\django2\lib\site-packages (from django>=3.2->django-crispy
-forms) (2023.3)
```

Step5: Select Django2-select Installed-Update Index and see installed packages as follows.

Assignment 17. Demonstrate Database connectivity using MySql.

Step 1: Search MySQL Workbench

Step 2: Click on localhost

Step 3: Right Click in Schemas then create Schema and Give Name

Step 4: Open schema then right click on tables and create table

Step 5: Write Following Code

```python
import tkinter as tk

from mysql.connector import connect, Error

# Database connection configuration

db_config = {

    'host': '127.0.0.1',

    'user': 'root',

    'password': 'manager',

    'database': 'sys'

}

def execute_query(query, values=None):

    try:

        with connect(**db_config) as connection:

            cursor = connection.cursor()

            if values:

                cursor.execute(query, values)

            else:

                cursor.execute(query)

            connection.commit()

            return cursor.lastrowid

    except Error as e:

        print(f"Error executing query: {e}")


def insert_data():

    id = entry_id.get()
```

```python
    name = entry_name.get()
    email = entry_email.get()
    query = "INSERT INTO student (id,name, email) VALUES (%s, %s,%s)"
    values = (id,name, email)
    execute_query(query, values)


def update_data():
    id = entry_id.get()
    name = entry_name.get()
    email = entry_email.get()
    query = "UPDATE student SET name = %s, email = %s WHERE id = %s"
    values = (name, email,id)
    execute_query(query, values)


def delete_data():
    name = entry_name.get()
    query = "DELETE FROM student WHERE name = %s"
    values = (name,)
    execute_query(query, values)


def select_data():
    try:
        with connect(**db_config) as connection:
            query = "SELECT * FROM student"
            cursor = connection.cursor()
            cursor.execute(query)
            rows = cursor.fetchall()
            for row in rows:
                print(row)  # You can modify this to display the data in your Tkinter application
    except Error as e:
        print(f"Error executing query: {e}")
```

```python
# Tkinter application setup
root = tk.Tk()
label_id = tk.Label(root, text="ID")
label_id.grid(row=0, column=0)

entry_id = tk.Entry(root)
entry_id.grid(row=0, column=1)

label_name = tk.Label(root, text="Name")
label_name.grid(row=1, column=0)

entry_name = tk.Entry(root)
entry_name.grid(row=1, column=1)

label_email = tk.Label(root, text="Email")
label_email.grid(row=2, column=0)

entry_email = tk.Entry(root)
entry_email.grid(row=2, column=1)

button_insert = tk.Button(root, text="Insert", command=insert_data)
button_insert.grid(row=3, column=0)

button_update = tk.Button(root, text="Update", command=update_data)
button_update.grid(row=3, column=1)
button_delete = tk.Button(root, text="Delete", command=delete_data)
button_delete.grid(row=3, column=2)
button_select = tk.Button(root, text="Select", command=select_data)
button_select.grid(row=3, column=3)
root.mainloop()
```
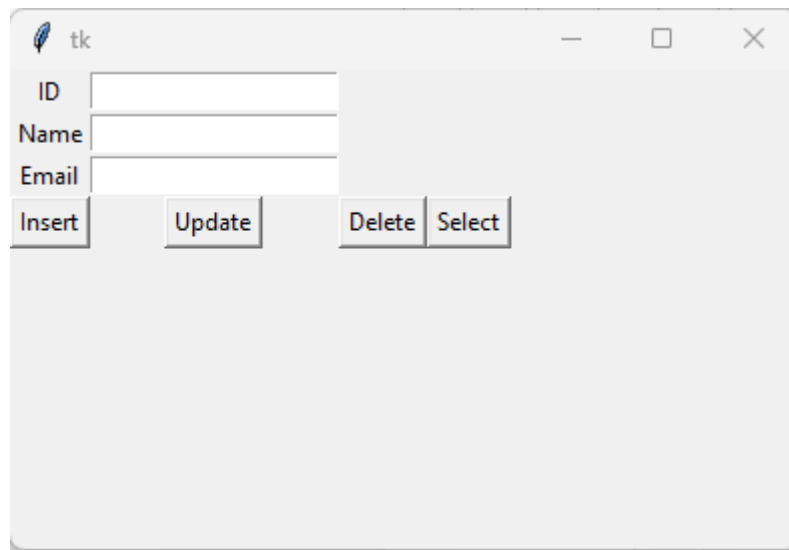
OUTPUT:-