**INFORMATION & COMMUNICATION TECHNOLOGY DEPARTMENT**

**SESI 2: 2022/2023**

**FINAL ASSESSMENT: PRACTICAL TEST**

**SET 1**

|  |  |  |  |
| --- | --- | --- | --- |
| **COURSE CODE:** DFP40203 | **COURSE NAME:** PYTHON PROGRAMMING | | |
| NAME: PUTERA NAQIB KHUSAIRI BIN ASRI | | | |
| **REGISTRATION NO:** 10DDT21F1021 | | **CLASS:** DDT5A | **DURATION:** 2 HOURS |
| |  | | --- | | **CLO1:** Construct Python application based on given scenario (P4,PLO3) | | | | |

**ANSWER :**

|  |  |  |
| --- | --- | --- |
| **Question 1 :**   1. Codes:  |  | | --- | | from parcel\_calculator import calculate\_price  # Get user input for parcel dimensions and weight  length = float(input("Enter parcel length in centimeters: "))  width = float(input("Enter parcel width in centimeters: "))  height = float(input("Enter parcel height in centimeters: "))  weight = float(input("Enter parcel weight in kilograms: "))  # Calculate parcel price using imported function  price = calculate\_price(length, width, height, weight)  # Print parcel price to user  print("The price of your parcel is: $", price) |   Output (Error) :   |  | | --- | |  |  1. Module Not Found Error : No Module named ‘parcel\_calculator’   error found on line 1 , error said The code run did not find a module named 'parcel\_calculator' |

|  |  |
| --- | --- |
| 1. Code (parcel\_calculator.py):  |  | | --- | | def calculate\_price(length, width, height, weight):  volume=length\*width\*height  try:  Parcel=open("Parcel.txt",'a')  Parcel.write("Length : "+str(length)+"\n")  Parcel.write("Width : "+str(width))  Parcel.write("\nHeight : "+str(height)+"\n")  Parcel.write("Weight : "+str(weight)+"\n")  Parcel.write("\n")  except:  print("Fail create and insert data in Parcel.txt")  if weight>=0 or weight<1:  if volume>=0 and volume<=5000:  price=3  elif volume>=5001 and volume<=10000:  price=5  elif volume>10000:  price=7  elif weight>=1 or weight<=5:  if volume>=0 and volume<=5000:  price=5  elif volume>=5001 and volume<=10000:  price=7  elif volume>10000:  price=9  elif weight>=5:  if volume>=0 and volume<=5000:  price=7  elif volume>=5001 and volume<=10000:  price=9  elif volume>10000:  price=11  return price |   **Output:** |

|  |  |  |
| --- | --- | --- |
| **Question 2 :**   1. Codes (gui\_parcel.py):  |  | | --- | | import tkinter as tk  import parcel\_calculator  class ParcelCalculatorGUI:  def \_\_init\_\_(self, master):  self.master = master  master.title("Parcel Calculator")  master.geometry("300x150")  # Create length input label and entry field  self.length\_label = tk.Label(master, text="Length (cm):")  self.length\_label.grid(row=0, column=0)  self.length\_entry = tk.Entry(master)  self.length\_entry.grid(row=0, column=1)  # Create width input label and entry field  self.width\_label = tk.Label(master, text="Width (cm):")  self.width\_label.grid(row=1, column=0)  self.width\_entry = tk.Entry(master)  self.width\_entry.grid(row=1, column=1)  # Create height input label and entry field  self.height\_label = tk.Label(master, text="Height (cm):")  self.height\_label.grid(row=2, column=0)  self.height\_entry = tk.Entry(master)  self.height\_entry.grid(row=2, column=1)  # Create weight input label and entry field  self.weight\_label = tk.Label(master, text="Weight (kg):")  self.weight\_label.grid(row=3, column=0)  self.weight\_entry = tk.Entry(master)  self.weight\_entry.grid(row=3, column=1)  # Create calculate button  self.calculate\_button = tk.Button(master, text="Calculate", command=self.calculate\_price)  self.calculate\_button.grid(row=4, column=0, columnspan=2)  # Create price label  self.price\_label = tk.Label(master, text="")  self.price\_label.grid(row=5, column=0, columnspan=2)  def calculate\_price(self):  # Get parcel dimensions and weight from entry fields  length = float(self.length\_entry.get())  width = float(self.width\_entry.get())  height = float(self.height\_entry.get())  weight = float(self.weight\_entry.get())    # Calculate parcel price using imported function  price = parcel\_calculator.calculate\_price(length, width, height, weight)  # Update price label with calculated price  self.price\_label.config(text="The price of your parcel is: $ " + str(price))  a= tk.Tk()  parcel\_calculator\_gui = ParcelCalculatorGUI(a)  a.mainloop() |   Output:   |  | | --- | |  | |

|  |  |  |
| --- | --- | --- |
| 1. Code (gui\_parcel.py) :  |  | | --- | | import tkinter as tk  import tkinter.messagebox as mbox  import parcel\_calculator  class ParcelCalculatorGUI:  def \_\_init\_\_(self, master):  self.master = master  master.title("Parcel Calculator")  master.geometry("260x290")  master.configure(bg="lightgray")  # Create length input label and entry field  self.id\_label = tk.Label(master, text="Item ID :", bg="lightgray")  self.id\_label.grid(row=0, column=0, padx=10, pady=10, sticky="w")  self.id\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.id\_entry.grid(row=0, column=1, padx=10, pady=10)  # Create length input label and entry field  self.length\_label = tk.Label(master, text="Length (cm):", bg="lightgray")  self.length\_label.grid(row=1, column=0, padx=10, pady=10, sticky="w")  self.length\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.length\_entry.grid(row=1, column=1, padx=10, pady=10)  # Create width input label and entry field  self.width\_label = tk.Label(master, text="Width (cm):", bg="lightgray")  self.width\_label.grid(row=2, column=0, padx=10, pady=10, sticky="w")  self.width\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.width\_entry.grid(row=2, column=1, padx=10, pady=10)  # Create height input label and entry field  self.height\_label = tk.Label(master, text="Height (cm):", bg="lightgray")  self.height\_label.grid(row=3, column=0, padx=10, pady=10, sticky="w")  self.height\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.height\_entry.grid(row=3, column=1, padx=10, pady=10)  # Create weight input label and entry field  self.weight\_label = tk.Label(master, text="Weight (kg):", bg="lightgray")  self.weight\_label.grid(row=4, column=0, padx=10, pady=10, sticky="w")  self.weight\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.weight\_entry.grid(row=4, column=1, padx=10, pady=10)  # Create calculate button  self.calculate\_button = tk.Button(master, text="Calculate", command=self.calculate\_price,  bg="green", fg="white")  self.calculate\_button.grid(row=5, column=0, columnspan=2, padx=10, pady=10)  # Create price label  self.price\_label = tk.Label(master, text="", bg="lightgray")  self.price\_label.grid(row=6, column=0, columnspan=2, padx=10, pady=10)  def calculate\_price(self):  # Get parcel dimensions and weight from entry fields  Item\_id = str(self.id\_entry.get())  length = float(self.length\_entry.get())  width = float(self.width\_entry.get())  height = float(self.height\_entry.get())  weight = float(self.weight\_entry.get())  volume=length\*width\*height  # Calculate parcel price using imported function  price = parcel\_calculator.calculate\_price(length, width, height, weight)  # Display the result in a message box  mbox.showinfo("Parcel Price", "The price of your parcel is: $ " + str(price))  a = tk.Tk()  parcel\_calculator\_gui = ParcelCalculatorGUI(a)  a.mainloop() |   Output:   |  | | --- | |  | |

|  |  |  |
| --- | --- | --- |
| **Question 3 :**   1. Codes (MySQL.py):  |  | | --- | | import mysql.connector  try:  q3db=mydb.cursor()  create="CREATE DATABASE IF NOT EXISTS LelemoveSystem"  q3db.execute(create)  except mysql.connector.Error as x:  print("Fail Creating database : {}".format(ror)) |  1. Codes (MySQL.py):  |  | | --- | | import mysql.connector  try:  q3db=mydb.cursor()  create="CREATE DATABASE IF NOT EXISTS LelemoveSystem"  q3db.execute(create)  try:  mydb.database="LelemoveSystem"  try:  tbl="CREATE TABLE IF NOT EXISTS Customer"\  "(Item\_id VARCHAR(10) PRIMARY KEY, "\  "Item\_height VARCHAR(5), "\  "Item\_width VARCHAR(5), Item\_length VARCHAR(5), "\  "Item\_volume VARCHAR(10), Item\_price VARCHAR(5))"  q3db.execute(tbl)  except mysql.connector.Error as ctb:  print("Fail Creating Table : {}".format(ctb))  except mysql.connector.Error as dnf:  print("Database Not Found : {}".format(dnf))  except mysql.connector.Error as x:  print("Fail Creating database : {}".format(ror)) | |

|  |  |  |
| --- | --- | --- |
| 1. Code (MySQL.py) :  |  | | --- | | import mysql.connector  import tkinter.messagebox as mbox  def InsertDatabase(Item\_id, length, width, height, volume, price):  try:  mydb = mysql.connector.connect(  host="localhost",  user="root",  password="",  database="LelemoveSystem")  mydbse = mydb.cursor()  mydbse.execute("SELECT \* FROM Customer WHERE Item\_id=%s",  (Item\_id,))  sameinpt=mydbse.fetchone()  if sameinpt:  mbox.showinfo("Same ID","Your Item Id alredy in data")  else:  mydbse.execute("INSERT INTO Customer"  "(Item\_id,Item\_height,Item\_width,Item\_length,Item\_volume,Item\_price)"  "VALUES(%s,%s,%s,%s,%s,%s)",  (Item\_id,height,width,length,volume,price))  mydb.commit()  mbox.showinfo("Record Success","Your Application has been submitted")    except mysql.connector.Error as err:  print("Failed Insert data : {}".format(err)+"\n")  mydb=mysql.connector.connect(host="localhost",  user="root",  password="")  try:  q3db=mydb.cursor()  create="CREATE DATABASE IF NOT EXISTS LelemoveSystem"  q3db.execute(create)  try:  mydb.database="LelemoveSystem"  try:  tbl="CREATE TABLE IF NOT EXISTS Customer"\  "(Item\_id VARCHAR(10) PRIMARY KEY, "\  "Item\_height VARCHAR(5), "\  "Item\_width VARCHAR(5), Item\_length VARCHAR(5), "\  "Item\_volume VARCHAR(10), Item\_price VARCHAR(5))"  q3db.execute(tbl)  except mysql.connector.Error as ctb:  print("Fail Creating Table : {}".format(ctb))  except mysql.connector.Error as dnf:  print("Database Not Found : {}".format(dnf))  except mysql.connector.Error as x:  print("Fail Creating database : {}".format(ror)) |   Code (gui\_parcel.py) :   |  | | --- | | import tkinter as tk  import tkinter.messagebox as mbox  import parcel\_calculator  import MySQL  class ParcelCalculatorGUI:  def \_\_init\_\_(self, master):  self.master = master  master.title("Parcel Calculator")  master.geometry("260x290")  master.configure(bg="lightgray")  # Create length input label and entry field  self.id\_label = tk.Label(master, text="Item ID :", bg="lightgray")  self.id\_label.grid(row=0, column=0, padx=10, pady=10, sticky="w")  self.id\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.id\_entry.grid(row=0, column=1, padx=10, pady=10)  # Create length input label and entry field  self.length\_label = tk.Label(master, text="Length (cm):", bg="lightgray")  self.length\_label.grid(row=1, column=0, padx=10, pady=10, sticky="w")  self.length\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.length\_entry.grid(row=1, column=1, padx=10, pady=10)  # Create width input label and entry field  self.width\_label = tk.Label(master, text="Width (cm):", bg="lightgray")  self.width\_label.grid(row=2, column=0, padx=10, pady=10, sticky="w")  self.width\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.width\_entry.grid(row=2, column=1, padx=10, pady=10)  # Create height input label and entry field  self.height\_label = tk.Label(master, text="Height (cm):", bg="lightgray")  self.height\_label.grid(row=3, column=0, padx=10, pady=10, sticky="w")  self.height\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.height\_entry.grid(row=3, column=1, padx=10, pady=10)  # Create weight input label and entry field  self.weight\_label = tk.Label(master, text="Weight (kg):", bg="lightgray")  self.weight\_label.grid(row=4, column=0, padx=10, pady=10, sticky="w")  self.weight\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.weight\_entry.grid(row=4, column=1, padx=10, pady=10)  # Create calculate button  self.calculate\_button = tk.Button(master, text="Calculate", command=self.calculate\_price,  bg="green", fg="white")  self.calculate\_button.grid(row=5, column=0, columnspan=2, padx=10, pady=10)  # Create price label  self.price\_label = tk.Label(master, text="", bg="lightgray")  self.price\_label.grid(row=6, column=0, columnspan=2, padx=10, pady=10)  def calculate\_price(self):  # Get parcel dimensions and weight from entry fields  Item\_id = str(self.id\_entry.get())  Item\_id = Item\_id.upper()  length = float(self.length\_entry.get())  width = float(self.width\_entry.get())  height = float(self.height\_entry.get())  weight = float(self.weight\_entry.get())  volume=length\*width\*height  # Calculate parcel price using imported function  price = parcel\_calculator.calculate\_price(length, width, height, weight)  # Display the result in a message box  mbox.showinfo("Parcel Price", "The price of your parcel is: $ " + str(price))  MySQL.InsertDatabase(Item\_id, length, width, height, volume, price)  a = tk.Tk()  parcel\_calculator\_gui = ParcelCalculatorGUI(a)  a.mainloop() | |

**CODES**

|  |
| --- |
| main.py:  from parcel\_calculator import calculate\_price  # Get user input for parcel dimensions and weight  length = float(input("Enter parcel length in centimeters: "))  width = float(input("Enter parcel width in centimeters: "))  height = float(input("Enter parcel height in centimeters: "))  weight = float(input("Enter parcel weight in kilograms: "))  # Calculate parcel price using imported function  price = calculate\_price(length, width, height, weight)  # Print parcel price to user  print("The price of your parcel is: $", price) |

|  |
| --- |
| parcel\_calculator.py:  def calculate\_price(length, width, height, weight):  volume=length\*width\*height  try:  Parcel=open("Parcel.txt",'a')  Parcel.write("Length : "+str(length)+"\n")  Parcel.write("Width : "+str(width))  Parcel.write("\nHeight : "+str(height)+"\n")  Parcel.write("Weight : "+str(weight)+"\n")  Parcel.write("\n")  except:  print("Fail create and insert data in Parcel.txt")  if weight>=0 or weight<1:  if volume>=0 and volume<=5000:  price=3  elif volume>=5001 and volume<=10000:  price=5  elif volume>10000:  price=7  elif weight>=1 or weight<=5:  if volume>=0 and volume<=5000:  price=5  elif volume>=5001 and volume<=10000:  price=7  elif volume>10000:  price=9  elif weight>=5:  if volume>=0 and volume<=5000:  price=7  elif volume>=5001 and volume<=10000:  price=9  elif volume>10000:  price=11  return price |

|  |
| --- |
| gui\_parcel.py:  import tkinter as tk  import tkinter.messagebox as mbox  import parcel\_calculator  import MySQL  class ParcelCalculatorGUI:  def \_\_init\_\_(self, master):  self.master = master  master.title("Parcel Calculator")  master.geometry("260x290")  master.configure(bg="lightgray")  # Create length input label and entry field  self.id\_label = tk.Label(master, text="Item ID :", bg="lightgray")  self.id\_label.grid(row=0, column=0, padx=10, pady=10, sticky="w")  self.id\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.id\_entry.grid(row=0, column=1, padx=10, pady=10)  # Create length input label and entry field  self.length\_label = tk.Label(master, text="Length (cm):", bg="lightgray")  self.length\_label.grid(row=1, column=0, padx=10, pady=10, sticky="w")  self.length\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.length\_entry.grid(row=1, column=1, padx=10, pady=10)  # Create width input label and entry field  self.width\_label = tk.Label(master, text="Width (cm):", bg="lightgray")  self.width\_label.grid(row=2, column=0, padx=10, pady=10, sticky="w")  self.width\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.width\_entry.grid(row=2, column=1, padx=10, pady=10)  # Create height input label and entry field  self.height\_label = tk.Label(master, text="Height (cm):", bg="lightgray")  self.height\_label.grid(row=3, column=0, padx=10, pady=10, sticky="w")  self.height\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.height\_entry.grid(row=3, column=1, padx=10, pady=10)  # Create weight input label and entry field  self.weight\_label = tk.Label(master, text="Weight (kg):", bg="lightgray")  self.weight\_label.grid(row=4, column=0, padx=10, pady=10, sticky="w")  self.weight\_entry = tk.Entry(master, relief="solid", borderwidth=2)  self.weight\_entry.grid(row=4, column=1, padx=10, pady=10)  # Create calculate button  self.calculate\_button = tk.Button(master, text="Calculate", command=self.calculate\_price,  bg="green", fg="white")  self.calculate\_button.grid(row=5, column=0, columnspan=2, padx=10, pady=10)  # Create price label  self.price\_label = tk.Label(master, text="", bg="lightgray")  self.price\_label.grid(row=6, column=0, columnspan=2, padx=10, pady=10)  def calculate\_price(self):  # Get parcel dimensions and weight from entry fields  Item\_id = str(self.id\_entry.get())  Item\_id = Item\_id.upper()  length = float(self.length\_entry.get())  width = float(self.width\_entry.get())  height = float(self.height\_entry.get())  weight = float(self.weight\_entry.get())  volume=length\*width\*height  # Calculate parcel price using imported function  price = parcel\_calculator.calculate\_price(length, width, height, weight)  # Display the result in a message box  mbox.showinfo("Parcel Price", "The price of your parcel is: $ " + str(price))  MySQL.InsertDatabase(Item\_id, length, width, height, volume, price)  a = tk.Tk()  parcel\_calculator\_gui = ParcelCalculatorGUI(a)  a.mainloop() |

|  |
| --- |
| MySQL.py:  import mysql.connector  import tkinter.messagebox as mbox  def InsertDatabase(Item\_id, length, width, height, volume, price):  try:  mydb = mysql.connector.connect(  host="localhost",  user="root",  password="",  database="LelemoveSystem")  mydbse = mydb.cursor()  mydbse.execute("SELECT \* FROM Customer WHERE Item\_id=%s",  (Item\_id,))  sameinpt=mydbse.fetchone()  if sameinpt:  mbox.showinfo("Same ID","Your Item Id alredy in data")  else:  mydbse.execute("INSERT INTO Customer"  "(Item\_id,Item\_height,Item\_width,Item\_length,Item\_volume,Item\_price)"  "VALUES(%s,%s,%s,%s,%s,%s)",  (Item\_id,height,width,length,volume,price))  mydb.commit()  mbox.showinfo("Record Success","Your Application has been submitted")    except mysql.connector.Error as err:  print("Failed Insert data : {}".format(err)+"\n")  mydb=mysql.connector.connect(host="localhost",  user="root",  password="")  try:  q3db=mydb.cursor()  create="CREATE DATABASE IF NOT EXISTS LelemoveSystem"  q3db.execute(create)  try:  mydb.database="LelemoveSystem"  try:  tbl="CREATE TABLE IF NOT EXISTS Customer"\  "(Item\_id VARCHAR(10) PRIMARY KEY, "\  "Item\_height VARCHAR(5), "\  "Item\_width VARCHAR(5), Item\_length VARCHAR(5), "\  "Item\_volume VARCHAR(10), Item\_price VARCHAR(5))"  q3db.execute(tbl)  except mysql.connector.Error as ctb:  print("Fail Creating Table : {}".format(ctb))  except mysql.connector.Error as dnf:  print("Database Not Found : {}".format(dnf))  except mysql.connector.Error as x:  print("Fail Creating database : {}".format(ror)) |

**ANSWER : OUTPUT**

|  |
| --- |
| main.py : |

|  |
| --- |
| Parcel.txt : |

|  |
| --- |
| gui\_parcel.py : |

|  |
| --- |
| Parcel.txt: |

**ANSWER: DATABASE SCREENSHOTS**

|  |
| --- |
| Parcel.txt: |

|  |
| --- |
| localhost : |