**Part I**

|  |
| --- |
| **Code :**  import json  # Split data based on ascending part  def split\_data\_into\_part(list\_order):  list\_new = []  my\_list = []    for i in range(len(list\_order)):  if i+1 < len(list\_order):  if list\_order[i] < list\_order[i+1]:  my\_list.append(list\_order[i])  else:  my\_list.append(list\_order[i])  list\_new.append(my\_list)  my\_list = []    return list\_new    # Do it some of math  def doing\_some\_math(list\_part):  dict\_part = {}  median = 0;  number\_data = len(list\_part);  list\_part = sorted(list\_part)  mean = sum(list\_part) / number\_data;    if number\_data % 2 != 0:  median = list\_part[number\_data // 2]  return {  'Our Data' : list\_part,  'Mean' : mean,  'Median' : median  }  else:  median = (list\_part[int((number\_data-1)/2)] + list\_part[int(number\_data / 2)]) / 2  return {  'Our Data' : list\_part,  'Mean' : mean,  'Median' : median  }  output\_data = [] # output  list\_order = [3,4,5,7,12,25,23,29,28,27,31,32] # Input  list\_order\_part = split\_data\_into\_part(list\_order)  for list\_math in list\_order\_part:  output\_data.append(doing\_some\_math(list\_math))    # beautify to be json  print(json.dumps(output\_data, sort\_keys=True, indent=4)) |
| **Output (sample):**  [  {  "Mean": 9.333333333333334,  "Median": 6.0,  "Our Data": [  3,  4,  5,  7,  12,  25  ]  },  {  "Mean": 26.0,  "Median": 26.0,  "Our Data": [  23,  29  ]  },  {  "Mean": 28.0,  "Median": 28,  "Our Data": [  28  ]  }  ] |
| **Screenshoot** |

**Part II**

|  |
| --- |
| **Code:**  import requests  import json  input\_data = [  {"amount" : 15000.0, "currency" : "IDR"},  {"amount" : 3.1, "currency" : "EUR"}  ]  output\_data = []  for data in input\_data:  amount = data["amount"]  currency = data["currency"]  response = requests.get(f'https://api.frankfurter.app/latest?amount={amount}&from={currency}&to=USD')  output\_data.append(response.json()["rates"]["USD"])  # print(json.dumps(response.json(), sort\_keys=True, indent=4))  print("============== Input ===============")  print(json.dumps(input\_data, sort\_keys=True, indent=4))  print("============== Output ===============")  print(json.dumps(output\_data, sort\_keys=True, indent=4)) |
| **Output:**  ============== Input ===============  [  {  "amount": 15000.0,  "currency": "IDR"  },  {  "amount": 3.1,  "currency": "EUR"  }  ]  ============== Output ===============  [  1.0642,  3.5966  ] |
| **Screenshoot:** |

**Part III**

|  |
| --- |
| **Code :**  import itertools  list\_eligible\_money = [1000, 10000, 20000]  dict\_lembar\_uang = {  "17000, 1",  "23000, 4",  "20000, 2",  "15000, 6",  }  for data in dict\_lembar\_uang:  list\_split = data.split(",")  harga\_barang = int(list\_split[0])  lembar\_uang = int(list\_split[1])    if (harga\_barang > 0 and harga\_barang <= 100000) and (lembar\_uang > 0 and lembar\_uang <= 10):  if lembar\_uang <= 1:  index\_search = [value for value in list\_eligible\_money if value <= harga\_barang]  print(f"Output : {[list\_eligible\_money[len(index\_search)]]} dan lembar ({lembar\_uang}) serta jumlah harga barang ({harga\_barang})")  else:  ten\_thousand = (harga\_barang // 10000) \* 10000  thousand = ((harga\_barang - ten\_thousand))    list\_eligible\_ten\_thousand = [value for value in list\_eligible\_money if value <= ten\_thousand]  list\_eligible\_thousand = [value for value in list\_eligible\_money if value <= thousand]  list\_put\_ten\_thousand = []  list\_put\_thousand = []    # Posibilities of ten thousand  for L in range(0, len(list\_eligible\_ten\_thousand)+1):  for subset in itertools.combinations(list\_eligible\_ten\_thousand, L):  if sum(list(subset)) == ten\_thousand:  list\_put\_ten\_thousand.append(list(subset))    count = 0  for vals in list\_eligible\_ten\_thousand:  while sum([vals] \* count) != ten\_thousand:  count+=1  list\_put\_ten\_thousand.append([vals] \* count)  count = 0    # Posibilities of thousand  for L in range(0, len(list\_eligible\_thousand)+1):  for subset in itertools.combinations(list\_eligible\_thousand, L):  if sum(list(subset)) == thousand:  list\_put\_thousand.append(list(subset))    count = 0  for vals in list\_eligible\_thousand:  while sum([vals] \* count) != thousand:  count+=1  list\_put\_thousand.append([vals] \* count)  count = 0    output = []  for i in range(len(list\_put\_ten\_thousand)):  for j in range(len(list\_put\_thousand)):  if lembar\_uang == (len(list\_put\_ten\_thousand[i]) +len(list\_put\_thousand[j])):  if output != list\_put\_ten\_thousand[i]+ list\_put\_thousand[j]:  output = list\_put\_ten\_thousand[i]+ list\_put\_thousand[j]  print(f"Output : {output} dan lembar ({lembar\_uang}) serta jumlah harga barang ({harga\_barang})")  else:  print("Melebihi maximum") |
| **Output :**  Output : [20000] dan lembar (1) serta jumlah harga barang (17000)  Output : [10000, 1000, 1000, 1000, 1000, 1000] dan lembar (6) serta jumlah harga barang (15000)  Output : [20000, 1000, 1000, 1000] dan lembar (4) serta jumlah harga barang (23000)  Output : [10000, 10000] dan lembar (2) serta jumlah harga barang (20000) |
| **Screenshot :** |

**Part IV**

|  |
| --- |
| **Code :**  import sqlite3  from random import randrange  conn = sqlite3.connect('TestCase4.db')  c = conn.cursor()  c.execute(''' SELECT count(name) FROM sqlite\_master WHERE type='table' AND name='STORE\_BOOK' ''')  if c.fetchone()[0]==1 :  print('Table STORE\_BOOK exists.')  else:  conn.execute('''  CREATE TABLE STORE\_BOOK(  ID\_BOOK INT PRIMARY KEY NOT NULL,  NAME\_BOOK TEXT NOT NULL,  DESCRIPTION\_BOOK TEXT NOT NULL,  AUTHOR\_BOOK TEXT NOT NULL,  LIMIT\_BOOK INT  );  ''')  print('Created Table STORE\_BOOK.')  c.execute(''' SELECT count(name) FROM sqlite\_master WHERE type='table' AND name='USERS' ''')  if c.fetchone()[0]==1 :  print('Table USERS exists.')  else:  conn.execute('''  CREATE TABLE USERS(  ID\_USER INT PRIMARY KEY NOT NULL,  NAME\_USER TEXT NOT NULL,  ADDRESS\_USER TEXT NOT NULL  );  ''')  print('Created Table USERS.')  c.execute(''' SELECT count(name) FROM sqlite\_master WHERE type='table' AND name='PEMINJAMAN' ''')  if c.fetchone()[0]==1 :  print('Table PEMINJAMAN exists.')  else:  conn.execute('''  CREATE TABLE PEMINJAMAN(  ID\_PEMINJAMAN INTEGER PRIMARY KEY AUTOINCREMENT,  ID\_USER\_PEMINJAM INT NOT NULL,  ID\_BOOK INT NOT NULL,  STATUS\_PINJAMAN TEXT NOT NULL,  Timestamp DATETIME DEFAULT CURRENT\_TIMESTAMP  );  ''')  print('Created Table PEMINJAMAN.')  SESSION\_ID\_USER = 0  SESSION\_NAME\_USER = ""  while True:  print("System Start... Welcome to Perpustakaan")  print("Pilih Masukkan Perpustakaan")  print("1. Login")  print("2. Register")  print("3. Lihat Buku")  print("4. Peminjaman")  pilihan = int(input())    print()  if pilihan == 0:  print("Insert Seluruh Buku .. hanya untuk admin")  conn.execute("INSERT INTO STORE\_BOOK (ID\_BOOK,NAME\_BOOK,DESCRIPTION\_BOOK,AUTHOR\_BOOK,LIMIT\_BOOK) VALUES (111, 'Harry Potter (Cursed of Black)', 'Bla bla bla...', 'Bla bla bla...', 2)")  conn.execute("INSERT INTO STORE\_BOOK (ID\_BOOK,NAME\_BOOK,DESCRIPTION\_BOOK,AUTHOR\_BOOK,LIMIT\_BOOK) VALUES (222, 'Learning Programming', 'Bla bla bla...', 'Bla bla bla...', 1)")  conn.execute("INSERT INTO STORE\_BOOK (ID\_BOOK,NAME\_BOOK,DESCRIPTION\_BOOK,AUTHOR\_BOOK,LIMIT\_BOOK) VALUES (333, 'Hand Writing Book', 'Bla bla bla...', 'Bla bla bla...', 3)")  conn.commit()  elif pilihan == 1:  print("=== Login Data ===")  print("Masukkan ID User : ")  id\_user = int(input())  cursor = conn.execute("SELECT NAME\_USER FROM USERS WHERE ID\_USER = {}".format(id\_user))  for row in cursor.fetchall():  SESSION\_ID\_USER = id\_user  SESSION\_NAME\_USER = row[0]  print("Welcome ",row[0])  elif pilihan == 2:  print("=== Register Data ===")  generated\_id = randrange(100000, 1000000)  print("Masukkan Nama : ")  nama = input()  print("Masukkan Alamat : ")  alamat = input()  print("Save your ID here... ",generated\_id)  script = "INSERT INTO USERS (ID\_USER, NAME\_USER, ADDRESS\_USER) VALUES (?,?,?)"  conn.execute(script, (generated\_id, nama, alamat))  conn.commit()  elif pilihan == 3:  print("Buku Perpustakaan yang tersediaa...")  cursor = conn.execute("SELECT \* FROM STORE\_BOOK")  for row in cursor.fetchall():  print(f"ID BUKU : {row[0]} -- NAME : {row[1]} -- DESKRIPSI : {row[2]} -- BUKU TERSISA : {row[4]}")  elif pilihan == 4:  print("Masukkan ID Buku yang akan dipinjam")  id\_buku = int(input())  nama\_buku = ""  sisa\_buku = 0  cursor = conn.execute("SELECT ID\_BOOK, NAME\_BOOK ,LIMIT\_BOOK FROM STORE\_BOOK WHERE ID\_BOOK = {}".format(id\_buku))  for row in cursor.fetchall():  print(f"Buku yang anda pilih : {row[1]} -- SISA : {row[2]}")  nama\_buku = row[1]  sisa\_buku = row[2]  print("Anda yakin akan meminjam ? YES/NO ")  persetujuan = str(input())  if persetujuan == 'YES':  # if SESSION\_ID\_USER == 0:  # print("Anda guest, mohon login")  # else:  sisa\_buku -= 1  cursor = conn.execute("UPDATE STORE\_BOOK set LIMIT\_BOOK = {} WHERE ID\_BOOK = {}".format(sisa\_buku, id\_buku))    script\_peminjam = "INSERT INTO PEMINJAMAN (ID\_USER\_PEMINJAM, ID\_BOOK, STATUS\_PINJAMAN) VALUES (?,?,?)"  conn.execute(script\_peminjam, (SESSION\_ID\_USER, id\_buku, "PINJAM"))  print("Anda telah meminjam")    print()  conn.close() |