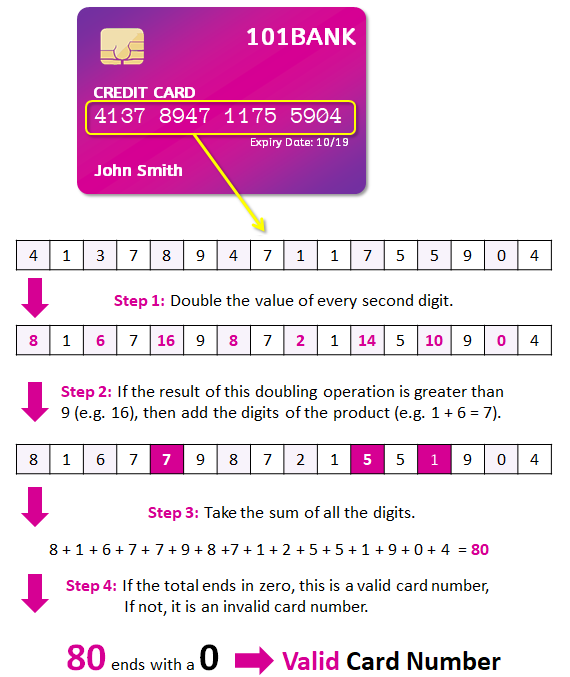
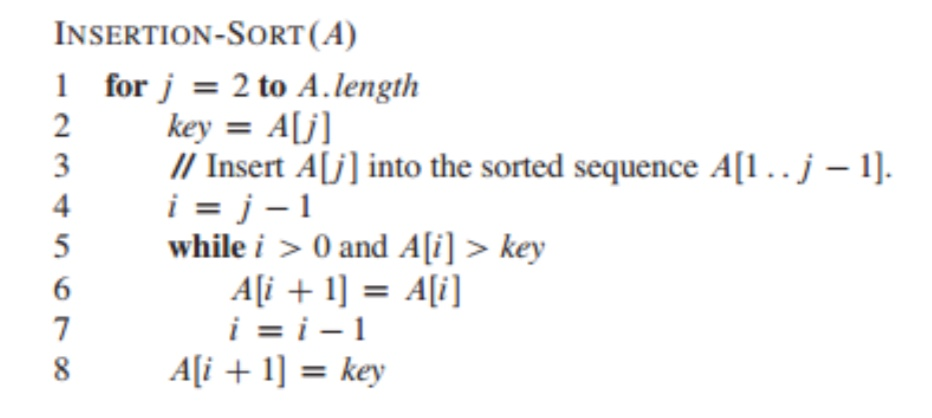
|  |  |
| --- | --- |
|  | DEPARTMENT OF COMPUTER SCIENCEFORMAN CHRISTIAN COLLEGE(A CHARTERED UNIVERSITY),LAHORE |

**Transmission Control Protocol (TCP)**

**Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Roll #: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Algorithms:**



Instructions:

* Complete both tasks given below.
* Submit your work before the deadline.

# TASK1:

## Write a TCP client server program in Python, task1client.py and task1server.py, that sends an unsorted array of integers from clients’ side to server as input and the server returns a sorted array using insertion sorting as output to client.

|  |
| --- |
| #========================  #task1client.py  #---------------------------------------  # Paste your solution below:  from socket import \*  serverName = 'localhost' *# 'servername'*  serverPort = 12001  clientSocket = socket(AF\_INET, SOCK\_STREAM)  clientSocket.connect((serverName, serverPort))  print(clientSocket.getsockname())  sentence = input('Input lowercase sentence: ')  clientSocket.send(sentence.encode())  modifiedSentence = clientSocket.recv(1024)  print('From Server: ', modifiedSentence.decode())  clientSocket.close()  #========================  #task1server.py  #---------------------------------------  # Paste your solution below:  from socket import \*  serverPort = 12001  serverSocket = socket(AF\_INET, SOCK\_STREAM)  serverSocket.bind(('localhost',serverPort))  serverSocket.listen(1)  print('The server is ready to receive')    def insertionSort(arr: list) -> list:  for i in range(0, len(arr)):  key = arr[i]  j = i - 1  while j >= 0 and key < arr[j]:  arr[j + 1] = arr[j]  j -= 1  arr[j + 1] = key  return arr    while 1:  connectionSocket, addr = serverSocket.accept()  print(connectionSocket.getpeername())  array = connectionSocket.recv(1024).decode()  array = [int(x) for x in array.split(', ')]  sorted\_array = insertionSort(array)  sorted\_string = ', '.join(str(x) for x in sorted\_array)  connectionSocket.send(sorted\_string.encode())  connectionSocket.close() |

# TASK2:

## Given below is a program that determines whether a provided credit card number is valid according to Luhn's algorithm. The global e-funds transfer companies such as Visa and Master Card use the checksum formula to facilitate faster online payments and transactions. The Luhn’s algorithm is a simple, public domain checksum algorithm that can be used to validate a variety of identification numbers. For more on Luhn’s Algorithm, [read](https://www.groundlabs.com/blog/anatomy-of-a-credit-card/) this.

## You are given two functions, one of which reads card number from user and other function validates and categorizes the given card number in one of four categories: VISA, MASTERCARD, AMEX and INVALID. Implement a TCP client server program for reading and validating credit card numbers in task2client.py and task2server.py files respectively.

## For testing purposes, here are a [few card numbers](https://developer.paypal.com/api/nvp-soap/payflow/integration-guide/test-transactions/#standard-test-cards).

def get\_card\_num():

    while True:

        card\_num = input("Number: ")

        try:

            if len(card\_num) > 0 and int(card\_num):

                break

        except ValueError:

            continue

    return card\_num

def validate\_card(credit\_card):

    if len(credit\_card) < 13 or 16 < len(credit\_card):

        print("INVALID")

        sys.exit(0)

    even, odd = 0, 0

    card\_len = len(credit\_card)

    if card\_len % 2 == 0:

        for i in range(card\_len):

            num = int(credit\_card[i])

            if i % 2 == 0:

                multiple = num \* 2

                if multiple >= 10:

                    even += multiple // 10

                    even += multiple % 10

                else:

                    even += multiple

            else:

                odd += num

    else:

        for i in range(card\_len):

            num = int(credit\_card[i])

            if i % 2 != 0:

                multiple = num \* 2

                if multiple >= 10:

                    even += multiple // 10

                    even += multiple % 10

                else:

                    even += multiple

            else:

                odd += num

    checksum = (even + odd) % 10

    if checksum == 0:

        first\_digit = int(credit\_card[0])

        second\_digit = int(credit\_card[1])

        if first\_digit == 3 and second\_digit == 4 or second\_digit == 7:

            return "AMEX"

        elif first\_digit == 5 and 1 <= second\_digit <= 5:

            return "MASTERCARD"

        elif first\_digit == 4:

            return "VISA"

        else:

            return "INVALID"

|  |
| --- |
| #========================  #task2client.py  #---------------------------------------  # Paste your solution below:  from socket import \*  serverName = 'localhost' *# 'servername'*  serverPort = 12001  clientSocket = socket(AF\_INET, SOCK\_STREAM)  clientSocket.connect((serverName, serverPort))  print(clientSocket.getsockname())  sentence = input('Input code: ')  clientSocket.send(sentence.encode())  modifiedSentence = clientSocket.recv(1024)  print('From Server: ', modifiedSentence.decode())  clientSocket.close()  #========================  #task2server.py  #---------------------------------------  # Paste your solution below:  from socket import \*  serverPort = 12001  serverSocket = socket(AF\_INET, SOCK\_STREAM)  serverSocket.bind(('localhost',serverPort))  serverSocket.listen(1)  print('The server is ready to receive')  def get\_card\_num(card\_num):  while True:  try:  if len(card\_num) > 0 and int(card\_num):  break  except ValueError:  continue  return card\_num  def validate\_card(credit\_card):  if len(credit\_card) < 13 or 16 < len(credit\_card):  print("INVALID")  sys.exit(0)  even, odd = 0, 0  card\_len = len(credit\_card)  if card\_len % 2 == 0:  for i in range(card\_len):  num = int(credit\_card[i])  if i % 2 == 0:  multiple = num \* 2  if multiple >= 10:  even += multiple // 10  even += multiple % 10  else:  even += multiple  else:  odd += num  else:  for i in range(card\_len):  num = int(credit\_card[i])  if i % 2 != 0:  multiple = num \* 2  if multiple >= 10:  even += multiple // 10  even += multiple % 10  else:  even += multiple  else:  odd += num  checksum = (even + odd) % 10  if checksum == 0:  first\_digit = int(credit\_card[0])  second\_digit = int(credit\_card[1])  if first\_digit == 3 and second\_digit == 4 or second\_digit == 7:  return "AMEX"  elif first\_digit == 5 and 1 <= second\_digit <= 5:  return "MASTERCARD"  elif first\_digit == 4:  return "VISA"  else:  return "INVALID"    while 1:  connectionSocket, addr = serverSocket.accept()  print(connectionSocket.getpeername())  cardCode = connectionSocket.recv(1024).decode()  connectionSocket.send(validate\_card(get\_card\_num(cardCode)).encode())  connectionSocket.close() |