

# Progetto di Programmazione di Reti

## Traccia 1

Riccardo Leonelli  
matricola: 0000938493

### Introduzione:

Ho realizzato quattro client UDP corrispondenti alle quattro stazioni meteo. Ognuna delle quali invia i propri dati meteorologici al server UDP del gateway. Una volta che il gateway ha ricevuto i messaggi chiude la connessione con i quattro client, raggruppa i dati e li spedisce al server. Il server dopo aver ricevuto il pacchetto contenente i dati di tutte stazioni lo decodifica e lo stampa sulla console. Tutto il procedimento viene eseguito all'infinito ripetutamente.

### File di riferimento:

- server.py
- gateway.py
- stazione.py
- stazione1.py
- stazione2.py
- stazione3.py
- stazione4.py
- main.py

# Client

Per creare le 4 stazioni meteorologiche ho utilizzato la funzione `Stazione` presente nel file `stazione.py`. Ad essa vengono passati i dati raccolti da una qualsiasi stazione, l'indirizzo ip e mac. Tale funzione provvede ad inviare i dati al gateway utilizzando una connessione UDP. Il messaggio codificato (utf8) che viene inviato contiene il rispettivo Ethernet header e IP header. Inoltre contiene un valore numerico corrispondente al tempo che verrà poi sottratto al tempo calcolato dal gateway al suo arrivo.

```
def Stazione(message,ip,mac):

    gateway_mac="8F:E6:15:B1:42:75"
    udp_gateway_ip="192.168.1.5"

    DIM_BUFFER=1024

    client = sk.socket(sk.AF_INET, sk.SOCK_DGRAM)
    server_address = ('localhost', 8400)#server del gateway
    client.connect(server_address)

    try:

        # Invio il messaggio
        print('Invio dati : "%s" ' % message)

        t=int(time.time()*1000000)#calcolo il tempo in nano secondi trascorsi dal 01/01/1970
        message+='+'
        message+=str(t)

        ethernet_header = mac + gateway_mac
        IP_header = ip + udp_gateway_ip
        packet = ethernet_header + IP_header + message

        sent = client.sendto(packet.encode('utf8'), server_address)

        # Aspetto la risposta dal gateway
        print('Attendo la risposta dal gateway')
        data, server = client.recvfrom(DIM_BUFFER)

        print ('received message "%s"' % data.decode('utf8'))

    except Exception as info:
        print(info)
    finally:
        print ('closing socket')
        client.close()
```

## Gateway

Il gateway possiede due componenti , un server UDP in grado di catturare i messaggi inviati dalle quattro stazioni e un client TCP che invierà tutti i dati raccolti al server di destinazione. Ad ogni messaggio catturato, utilizzando lo slicing di python, vengono tagliati gli Header che verranno poi stampati sulla console , mentre i dati della rispettiva stazione vengono inseriti in un array. Inoltre il tempo calcolato dal client ,che è presente alla fine del messaggio, viene sottratto al tempo calcolato dal gateway al rispettivo arrivo . Così da ottenere il tempo di trasmissione del pacchetto UDP, che successivamente viene stampato a video.

Per la interfaccia TCP del gateway si instaura un collegamento con il server principale che attenderà l'arrivo delle quattro misurazioni effettuate. I dati decriptati contenuti nell'array vengono inseriti in una stringa alla quale verranno aggiunti gli Header mac, ip e un intero corrispondente al tempo in cui il pacchetto è stato spedito. Fatto ciò esso viene criptato (utf8) e spedito al server centrale.

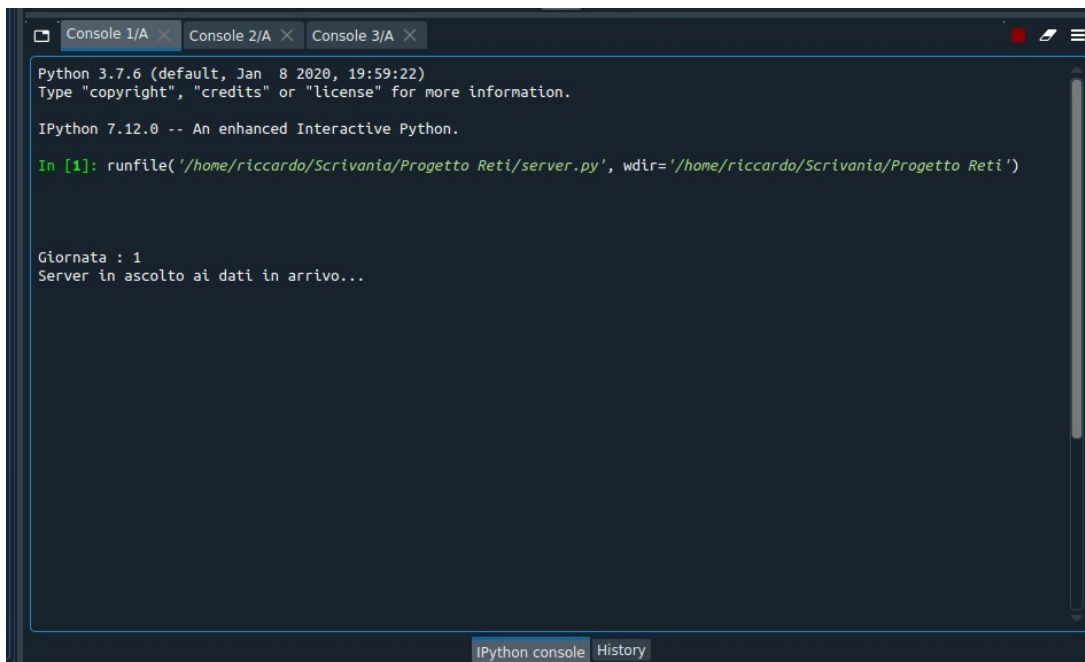
## Server

Il server dopo aver instaurato un collegamento con il gateway di tipo TCP si mette in attesa del messaggio. In seguito decodifica e scorpora il pacchetto, stampa i dati finali , gli header e il tempo di trasmissione del messaggio.

# Esecuzione

Per l' esecuzione occorre aprire tre console ed avviare in sequenza `server.py` , `gateway.py` , `main.py` .

Il server si metterà in attesa.



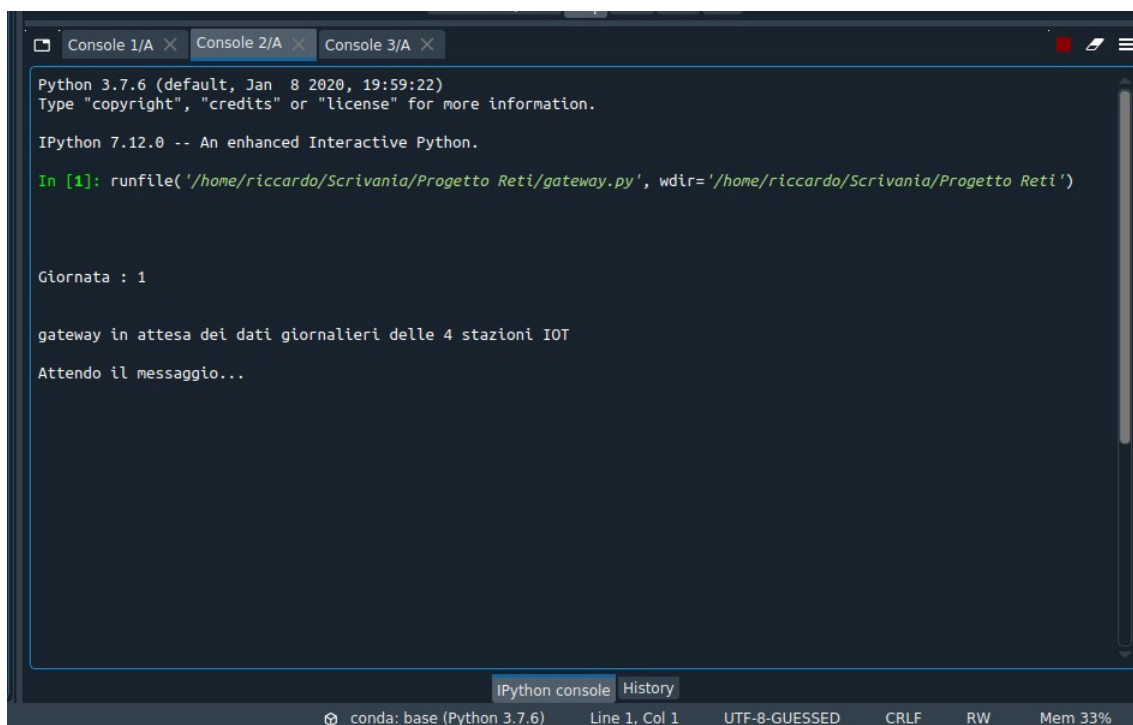
```
Python 3.7.6 (default, Jan 8 2020, 19:59:22)
Type "copyright", "credits" or "license" for more information.

IPython 7.12.0 -- An enhanced Interactive Python.

In [1]: runfile('/home/riccardo/Scrivania/Progetto Reti/server.py', wdir='/home/riccardo/Scrivania/Progetto Reti')

Giornata : 1
Server in ascolto ai dati in arrivo...
```

Il gateway si metterà in attesa.



```
Python 3.7.6 (default, Jan 8 2020, 19:59:22)
Type "copyright", "credits" or "license" for more information.

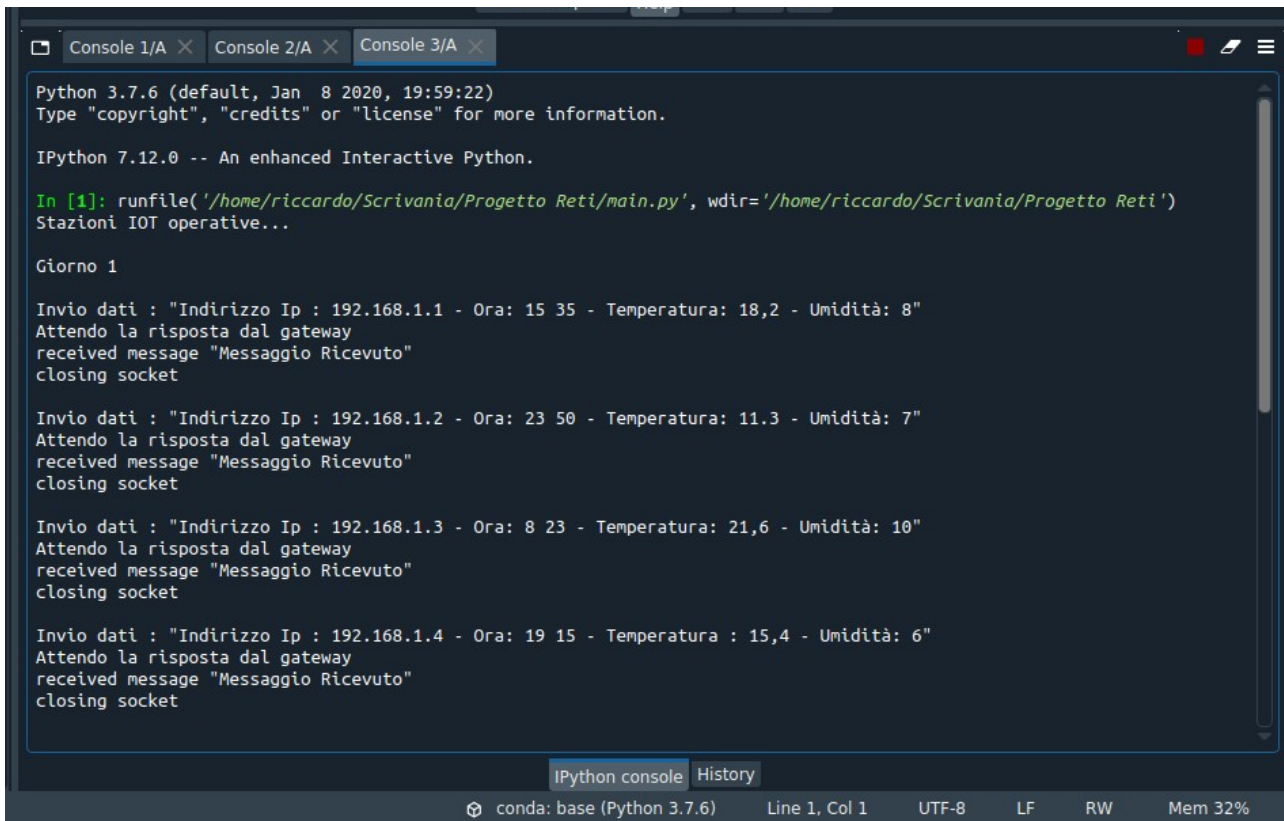
IPython 7.12.0 -- An enhanced Interactive Python.

In [1]: runfile('/home/riccardo/Scrivania/Progetto Reti/gateway.py', wdir='/home/riccardo/Scrivania/Progetto Reti')

Giornata : 1

gateway in attesa dei dati giornalieri delle 4 stazioni IOT
Attendo il messaggio...
```

main.py avvierà la funzione Main che provvederà ad eseguire i quattro client.



```
Python 3.7.6 (default, Jan 8 2020, 19:59:22)
Type "copyright", "credits" or "license" for more information.

IPython 7.12.0 -- An enhanced Interactive Python.

In [1]: runfile('/home/riccardo/Scrivania/Progetto Reti/main.py', wdir='/home/riccardo/Scrivania/Progetto Reti')
Stazioni IOT operative...

Giorno 1

Invio dati : "Indirizzo Ip : 192.168.1.1 - Ora: 15 35 - Temperatura: 18,2 - Umidità: 8"
Attendo la risposta dal gateway
received message "Messaggio Ricevuto"
closing socket

Invio dati : "Indirizzo Ip : 192.168.1.2 - Ora: 23 50 - Temperatura: 11,3 - Umidità: 7"
Attendo la risposta dal gateway
received message "Messaggio Ricevuto"
closing socket

Invio dati : "Indirizzo Ip : 192.168.1.3 - Ora: 8 23 - Temperatura: 21,6 - Umidità: 10"
Attendo la risposta dal gateway
received message "Messaggio Ricevuto"
closing socket

Invio dati : "Indirizzo Ip : 192.168.1.4 - Ora: 19 15 - Temperatura : 15,4 - Umidità: 6"
Attendo la risposta dal gateway
received message "Messaggio Ricevuto"
closing socket
```

IPython console History

conda: base (Python 3.7.6) Line 1, Col 1 UTF-8 LF RW Mem 32%

Il procedimento sarà infinito e automatico.