

**Tracce Progetti corso**  
**PROGRAMMAZIONE DI RETI**  
**ANNO ACCADEMICO 2021-2022**



## Modalità di Esame - Programmazione di Reti

Per accedere allo scritto di Programmazione di Reti è necessario presentare almeno una settimana prima della data d'appello scelta dallo studente, un progetto di programmazione in Python.

Il progetto consiste nello sviluppo di una delle tre tracce pubblicate sul Virtuale al termine del corso (linguaggio utilizzato Python).

Il progetto può essere consegnato direttamente via mail al docente o ricorrendo ad un repository a scelta dello studente.

Il voto finale sarà calcolato come media matematica del voto dello scritto e della valutazione del progetto.

Il voto del progetto ha validità per l'intero anno accademico di riferimento.



# Modalità di svolgimento del Progetto

Il progetto può essere svolto in gruppo. Ogni gruppo deve essere composto di al più tre studenti. Il materiale consegnato deve comprendere:

- il codice di tutte le componenti, con le indicazioni per la loro esecuzione.
- Un documento in formato pdf di una relazione contenente
  - una descrizione generale delle scelte di progetto effettuate.
  - una descrizione delle strutture dati utilizzate
- uno schema generale dei threads attivati sia nei server che nel client

L'organizzazione e la chiarezza dell'esposizione della relazione influiranno sul voto finale dell'esame.

L'utilizzo di metodologie di documentazione del software quali diagrammi UML (delle classi, di sequenza,...) sarà considerato positivamente ai fini della valutazione del progetto.

## INDICARE NEL DOCUMENTO DI PROGETTO:

- **L' INDIRIZZO EMAIL DI CIASCUNO STUDENTE APPARTENENTE AL GRUPPO DI LAVORO**
- **LA MATRICOLA DI CIASCUNO STUDENTE APPARTENENTE AL GRUPPO DI LAVORO**



# Traccia 1

## Progetto DRONI



# TRACCIA 1 – Progetto DRONI

Si ipotizzi di dover gestire una rete di consegne di piccoli pacchi a domicilio tramite l'utilizzo di droni. La rete è composta da:

- un client da cui l'operatore assegna a ciascun drone l'indirizzo di consegna del pacco,
- tre droni e
- un gateway che provvede ad effettuare il relay dei messaggi verso i droni e a fare da concentratore per raccogliere i messaggi provenienti dai droni ed inviarli al client.

Il client deve poter inviare l'indirizzo di consegna del pacco ad un drone, solo dopo che lo stesso drone si sia presentato al gateway come «disponibile». La connessione tra droni e gateway è di tipo UDP mentre la connessione tra client e gateway è di tipo TCP.

Ogni drone ha un suo indirizzo IPv4, e così le due interfacce del gateway e il client.

Sulla console del client si deve poter inserire l'indirizzo di consegna del pacco e l'identificativo (o anche l'ip address) del drone che dovrà effettuare la consegna. Tali informazioni devono essere inviate al gateway che provvederà ad inviare l'informazione sulla destinazione del pacco al drone incaricato. Tali informazioni devono essere visibili sulla console del gateway.

Infine sulla console del Drone dovrà essere visibile l'indirizzo di destinazione del pacchetto. Il drone dovrà quindi inviare al gateway un messaggio di avvenuta consegna del pacchetto (ipotizzare un tempo randomico in un intervallo a scelta) e rendersi nuovamente disponibile.

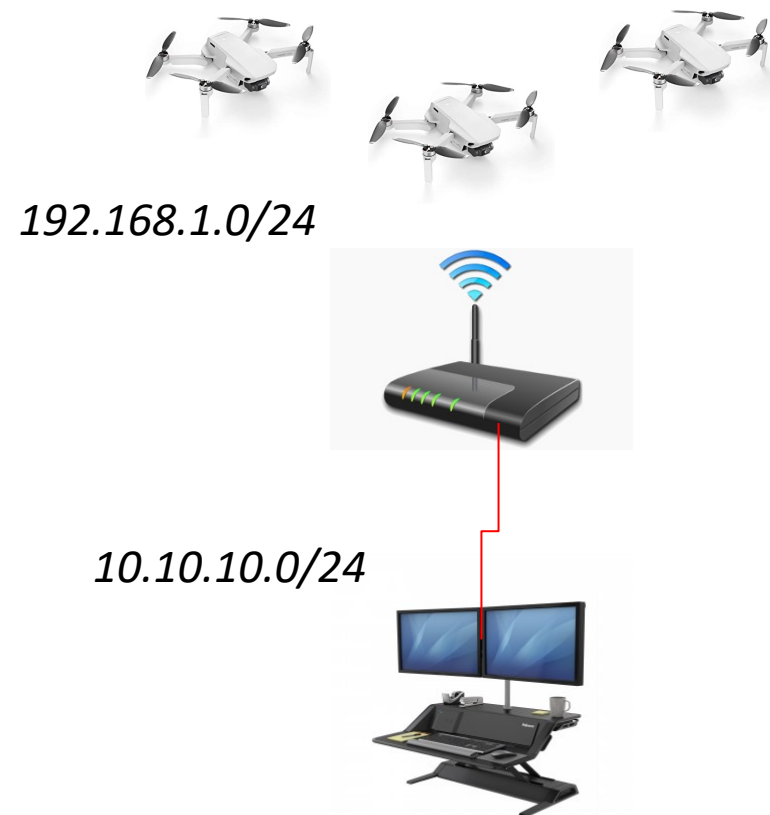
Sulla console del gateway devono comparire tutti i messaggi in transito con sorgente e destinatario.

I 3 droni hanno un indirizzamento appartenente ad una rete di Classe C del tipo 192.168.1.0/24

Il Gateway ha due interfacce di rete: quella verso i droni il cui IP Address appartiene alla stessa network dei dispositivi mentre l'interfaccia verso il client ha indirizzo ip appartenente alla classe 10.10.10.0/24, classe a cui appartiene anche l'IP address del gateway.

Si realizzi un emulatore Python che sfruttando il concetto dei socket visti in laboratorio consenta di simulare, utilizzando l'interfaccia di loopback del proprio PC, il comportamento di questo sistema.

Si devono simulare le connessioni UDP dei device verso il Gateway e la connessione TCP del Gateway verso il Client. Inoltre indicare la dimensione dei buffer utilizzati su ciascun canale trasmissivo, il tempo impiegato per trasmettere il pacchetto UDP ed il tempo impiegato per trasmettere il pacchetto TCP.



# Traccia 2

## Architettura client-server UDP per trasferimento file



## Traccia 2: Architettura client-server UDP per trasferimento file

Lo scopo de progetto è quello di progettare ed implementare in linguaggio Python un'applicazione client-server per il trasferimento di file che impieghi il servizio di rete senza connessione (socket tipo SOCK\_DGRAM, ovvero UDP come protocollo di strato di trasporto).

Il software deve permettere:

- Connessione client-server senza autenticazione;
- La visualizzazione sul client dei file disponibili sul server;
- Il download di un file dal server;
- L'upload di un file sul server;

La comunicazione tra client e server deve avvenire tramite un opportuno protocollo. Il protocollo di comunicazione deve prevedere lo scambio di due tipi di messaggi:

- messaggi di comando: vengono inviati dal client al server per richiedere l'esecuzione delle diverse operazioni;
- messaggi di risposta: vengono inviati dal server al client in risposta ad un comando con l'esito dell'operazione.

**Funzionalità del server:** Il server deve fornire le seguenti funzionalità:

- L'invio del messaggio di risposta al comando list al client richiedente;
- il messaggio di risposta contiene la file list, ovvero la lista dei nomi dei file disponibili per la condivisione;
- L'invio del messaggio di risposta al comando get contenente il file richiesto, se presente, od un opportuno messaggio di errore;
- La ricezione di un messaggio put contenente il file da caricare sul server e l'invio di un messaggio di risposta con l'esito dell'operazione.

**Funzionalità del client:** I client deve fornire le seguenti funzionalità:

- L'invio del messaggio list per richiedere la lista dei nomi dei file disponibili;
- L'invio del messaggio get per ottenere un file
- La ricezione di un file richiesta tramite il messaggio di get o la gestione dell'eventuale errore
- L'invio del messaggio put per effettuare l'upload di un file sul server e la ricezione del messaggio di risposta con l'esito dell'operazione.



# **Traccia 3**

## **Web Server Agenzia di Viaggi**





## Traccia 3: Python Web Server

Si immagini di dover realizzare un Web Server in Python per una agenzia di viaggi.

I requisiti del Web Server sono i seguenti:

- Il web server deve consentire l'accesso a più utenti in contemporanea
- La pagina iniziale deve consentire di visualizzare la lista dei servizi erogati dall'agenzia di viaggi e per ogni servizio avere un link di riferimento ad una pagina dedicata presente nella stessa working directory della pagina principale
- Nella pagina principale dovrà anche essere presente un link per il download di un file pdf da parte del browser
- Come requisito facoltativo si chiede di autenticare gli utenti nella fase iniziale della connessione.
- L'interruzione da tastiera (o da console) dell'esecuzione del web server deve essere opportunamente gestita in modo da liberare la risorsa socket.

