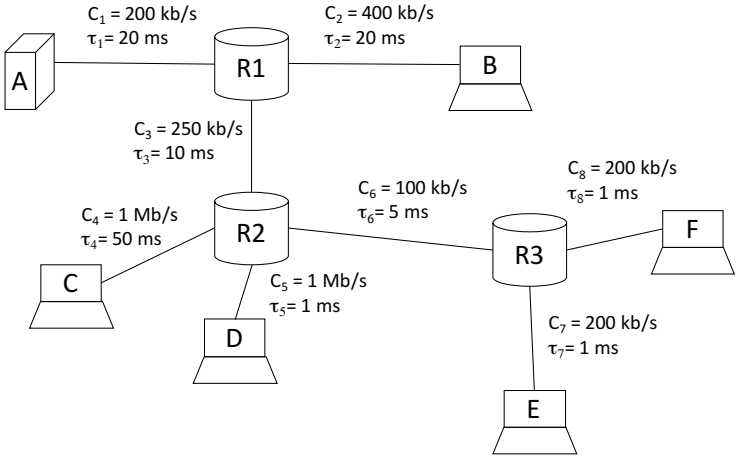
Prova in itinere – 4 Maggio 2016

Cognome	**STUDENTE**
Nome	**BRAVO**
Matricola	**SOLUZIONI**

Tempo complessivo a disposizione per lo svolgimento: 1h45m
Usare lo spazio dopo ogni Esercizio/Quesito per la risposta.

Es1 (10 pt)	Es2 (8 pt)	Ques (9 pt)	Lab (6pt)

Esercizio 1 (10 punti)



a) Una connessione TCP tra l'host A e l'host B nella rete in figura è caratterizzata dai seguenti parametri:

- Link bidirezionali e simmetrici
- MSS = 200 byte
- Lunghezza header complessivo (tutti i livelli), H = 50 byte
- Lunghezza ACK e segmenti di apertura, L_{ACK} = 250 byte
- RCWND = 1000 byte, SSTHRESH = 1600 byte

a.1) Si *calcoli* il tempo necessario a trasferire un file di dimensione F = 5 kbyte (dall'apertura della connessione alla ricezione dell'ultimo ACK)
a.2) Si *indichi* il rate medio di trasferimento del file da A a B

b) Nella rete a commutazione di pacchetto in figura, al tempo t=0 sono presenti 5 pacchetti in A diretti rispettivamente alle seguenti destinazioni: C, D, E, F, E. *Calcolare* l'istante di fine ricezione degli ultimi 3 pacchetti a destinazione assumendo che i pacchetti abbiamo le seguenti dimensioni: pacchetti verso C, L_C = 375 byte; pacchetti verso D, L_D = 250 byte; pacchetti verso E, L_E= 375 byte; pacchetti verso F, L_F= 125 byte.

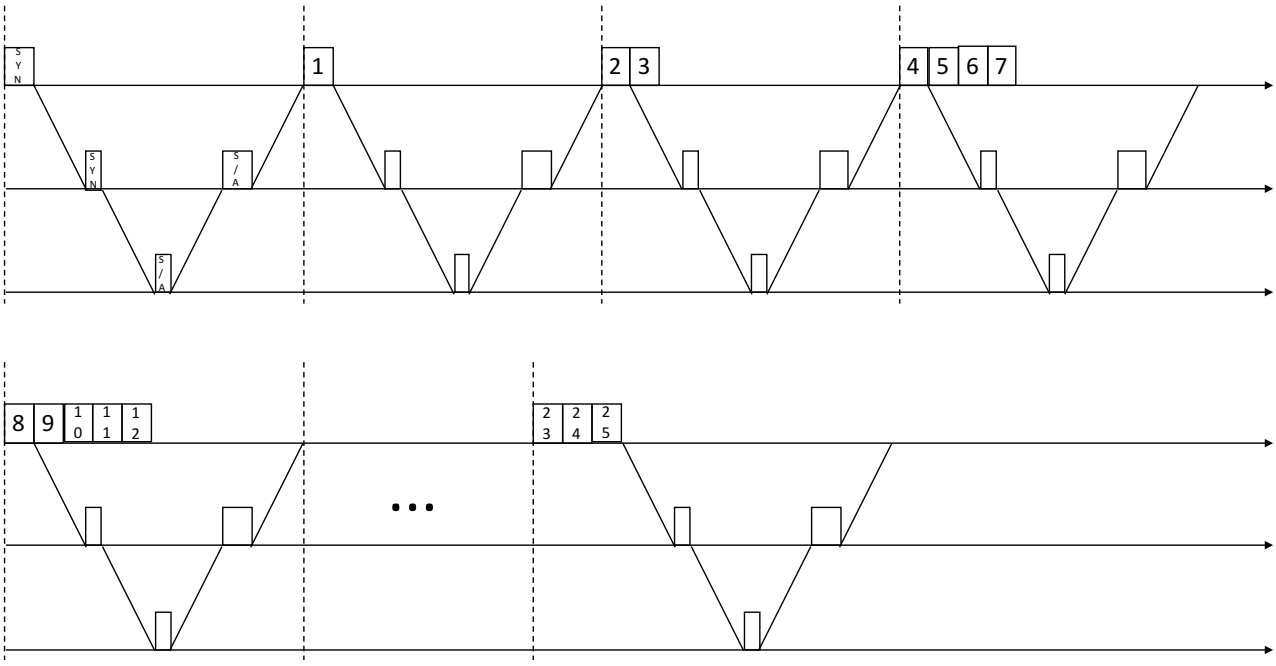
Soluzione

Punto a
RCWND = 1000 byte / 200 byte = 5 MSS
SSTHRESH = 1600 byte / 200 byte = 8 MSS

File = 5000 byte / 200 byte = 25 MSS
L = MSS + H = 250 byte

T_1 = L / C_1 = 250 * 8 / 200000 = 10 ms
T_2 = L / C_2 = 250 * 8 / 400000 = 5 ms
$T_1^{ACK} = T_1$; $T_2^{ACK} = T_2$
RTT = $2(T_1 + \tau_1 + T_2 + \tau_2)$ = 110 ms
Wc = RTT / T_1 = 11 MSS

T_{setup} = RTT = 110 ms



Dopo i segmenti di apertura della connessione, il TCP parte in modalità Slow Start. Tuttavia, prima di raggiungere la SSTHRESH, la finestra è limitata dalla RCWND a 5MSS. Quindi, una volta raggiunto tale valore, la finestra non aumenterà. Inoltre, dato che Wc > RCWND, la trasmissione non sarà mai continua.

Il tempo totale di trasferimento è dato da:

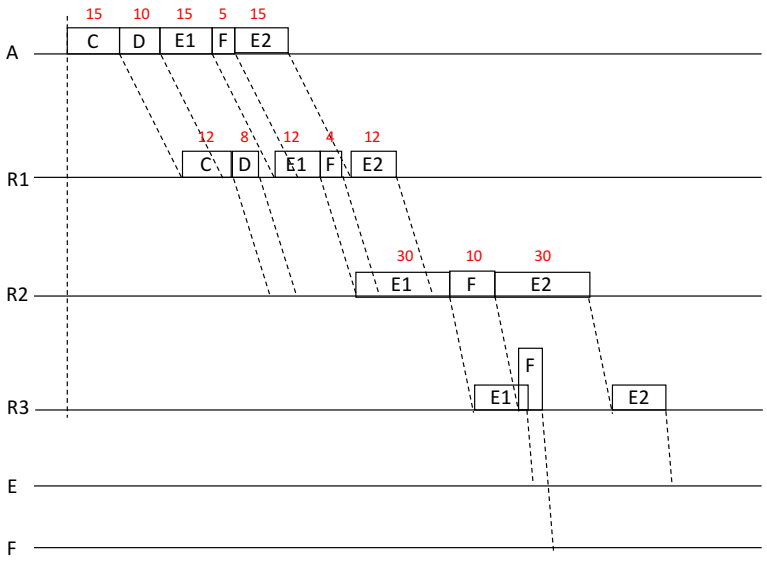
$$T_{tot} = T_{setup} + 6RTT + 2T_1 + RTT = 8RTT + 2T_1 = 900\ ms$$

Il rate medio di trasferimento è dato da:

$$R_{medio} = \frac{F}{T_{tot}} = 5000 * \frac{8}{0.9} = 44{,}44\ kbit/s$$

Punto b

$$T_1^C = \frac{L_C}{C_1} = \frac{375 * 8\ bit}{200\ kbps} = 15\ ms$$
$$T_3^C = \frac{L_C}{C_3} = \frac{375 * 8\ bit}{250\ kbps} = 12\ ms$$
$$T_1^D = \frac{L_D}{C_1} = \frac{250 * 8\ bit}{200\ kbps} = 10\ ms$$
$$T_3^D = \frac{L_D}{C_3} = \frac{250 * 8\ bit}{250\ kbps} = 8\ ms$$
$$T_1^E = \frac{L_E}{C_1} = \frac{375 * 8\ bit}{200\ kbps} = 15\ ms$$
$$T_3^E = \frac{L_E}{C_3} = \frac{375 * 8\ bit}{250\ kbps} = 12\ ms$$
$$T_6^E = \frac{L_E}{C_6} = \frac{375 * 8\ bit}{100\ kbps} = 30\ ms$$
$$T_7^E = \frac{L_E}{C_7} = \frac{375 * 8\ bit}{200\ kbps} = 15\ ms$$
$$T_1^F = \frac{L_F}{C_1} = \frac{125 * 8\ bit}{200\ kbps} = 5\ ms$$
$$T_3^F = \frac{L_F}{C_3} = \frac{125 * 8\ bit}{250\ kbps} = 4\ ms$$
$$T_6^F = \frac{L_F}{C_6} = \frac{125 * 8\ bit}{100\ kbps} = 10\ ms$$
$$T_8^F = \frac{L_F}{C_8} = \frac{125 * 8\ bit}{200\ kbps} = 5\ ms$$



$$T_{E1} = T_1^C + T_1^D + T_1^E + \tau_1 + T_3^E + \tau_3 + T_6^E + \tau_6 + T_7^E + \tau_7 = 15 + 10 + 15 + 20 + 12 + 10 + 30 + 5 + 15 + 1$$
$$= 133\ ms$$

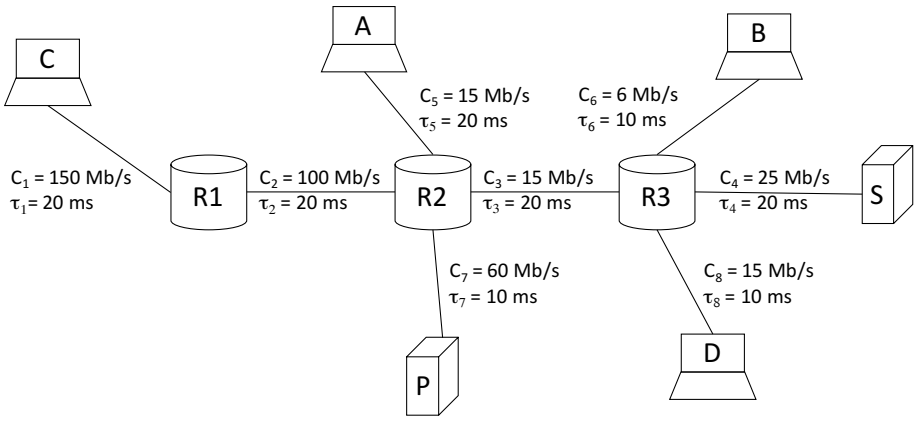
$$T_F = T_1^C + T_1^D + T_1^E + \tau_1 + T_3^E + \tau_3 + T_6^E + T_6^F + \tau_6 + T_8^F + \tau_8$$
$$= 15 + 10 + 15 + 20 + 12 + 10 + 30 + 10 + 5 + 5 + 1 = 133\ ms$$

$$T_{E2} = T_{E1} + T_6^E + T_6^F = 133 + 30 + 10 = 173\ ms$$

Esercizio 2 (8 punti)

Nella rete in figura sono rappresentati 4 client (A, B, C e D), 3 router (R1, R2 e R3), un server HTTP (S) e un proxy HTTP (P). Il client C vuole trasferire un documento formato da una pagina HTML di dimensione L_{HTML} = 30 kbyte che richiama 11 oggetti di dimensione L_{OGG} = 75 kbyte. Nella rete sono presenti anche 4 flussi interferenti di lunga durata: 2 da A a D, 2 da A a B. Si supponga che: i) i messaggi di apertura della connessione e di richiesta HTTP siano di lunghezza trascurabile, ii) la connessione TCP tra proxy HTTP e server HTTP sia sempre aperta e non occorra mandare una richiesta di apertura.
Si calcoli:

a) il tempo di trasferimento del documento nel caso in cui il client C senza proxy configurato apra in parallelo in modalità non-persistente tutte le connessioni TCP necessarie
b) il tempo di trasferimento del documento nel caso in cui il client C con proxy configurato apra un'unica connessione TCP persistente per scaricare tutti gli oggetti, ipotizzando che la pagina HTML e solo i primi 6 degli 11 oggetti siano presenti nella cache del proxy
c) nel caso b), il numero minimo di oggetti che occorre trovare nella cache del proxy per avere un tempo di trasferimento minore di 2 s.



Soluzione

Punto a

Il client manda/riceve messaggi HTTP direttamente al/dal server.
Durante il trasferimento della pagina HTML (1 flusso tra C e S) sul link R2-R3 ci sono 5 flussi che condividono 15 Mb/s. Il link è collo di bottiglia per tutti i flussi che ottengono 3 Mb/s ciascuno. Quindi la capacità vista dal trasferimento C-S sarà $C_{eff-5} = 3Mb/s$.

Il tempo di trasferimento della pagina HTML è: $T_{HTML} = \frac{L_{HTML}}{C_{eff-5}} = \frac{30*8\ kbit}{3\ Mbps} = 80\ ms$

Durante il trasferimento degli oggetti (11 flussi in parallelo tra C e D) sul link R2-R3 ci sono 15 flussi che condividono 15 Mb/s. Il link è collo di bottiglia per tutti i flussi che ottengono 1 Mb/s ciascuno. Quindi la capacità vista dal trasferimento C-S sarà $C_{eff-15} = 1Mb/s$.

Il tempo di trasferimento di un oggetto è: $T_{OGG} = \frac{L_{OGG}}{C_{eff-15}} = \frac{75*8\ kbit}{1\ Mbps} = 600\ ms$

Il RTT tra C e S è: $RTT_{C-S} = 2(\tau_1 + \tau_2 + \tau_3 + \tau_4) = 2 * 80 = 160\ ms$

Il tempo totale di trasferimento sarà:

$$T_{tot} = RTT + RTT + T_{HTML} + RTT + RTT + T_{OGG} = 4RTT + T_{HTML} + T_{OGG} = 1320\ ms$$

Punto b

Il client manda/riceve messaggi HTTP al/dal proxy. Per i primi 6, riceve l'oggetto direttamente dal proxy, per gli ultimi 5, il proxy scarica l'oggetto dal server e poi lo invia al client.
La capacità del trasferimento tra client e proxy è determinata dal link P-R2 e dunque pari a $C_{eff} = 60\ Mb/s$.

Il tempo di trasferimento della pagina HTML (diretta al proxy) è: $T_{HTML} = \frac{L_{HTML}}{C_{eff}} = \frac{30*8\ kbit}{60\ Mbps} = 4\ ms$

Il tempo di trasferimento di un oggetto (in cache) è: $T_{OGG-cache} = \frac{L_{OGG}}{C_{eff}} = \frac{75*8\ kbit}{60\ Mbps} = 10\ ms.$

La capacità del trasferimento tra proxy e server è determinata dal link R2-R3, su cui transitano 5 flussi, dunque pari a $C_{eff-5} = 3\ Mb/s$.

Il tempo di trasferimento di un oggetto (nel server) è: $T_{OGG-nocache} = \frac{L_{OGG}}{C_{eff-5}} = \frac{75*8\ kbit}{3\ Mbps} = 200\ ms.$

Il RTT tra C e P è: $RTT_{C-P} = 2(\tau_1 + \tau_2 + \tau_7) = 2 * 50 = 100\ ms$, mentre il RTT tra P e S è: $RTT_{P-S} = 2(\tau_7 + \tau_3 + \tau_4) = 2 * 50 = 100\ ms$

Il tempo totale di trasferimento è:

$$T_{tot} = RTT_{C-P} + RTT_{C-P} + T_{HTML} + 6(RTT_{C-P} + T_{OGG-cache}) + 5(RTT_{P-S} + T_{OGG-nocache} + RTT_{C-P} + T_{OGG-cache})$$
$$= 100 + 100 + 4 + 6(100 + 10) + 5(100 + 200 + 100 + 10) = 204 + 660 + 2050 = 2914\ ms$$

Punto c

Dall'ultima espressione del punto b) si può scrivere

$$T_{tot} = 204 + x(110) + (11 - x)(410)$$

dove *x* è il numero di oggetti trovati in cache.
Quindi da $T_{tot} = 204 + x(110) + (11 - x)(410) < 2000\ ms$, si ottiene $x > 9.046$. Occorre trovare in cache almeno 10 oggetti.

Quesiti (9 punti)

Q1

Nell'ambito del servizio DNS, *si illustrino* le differenze tra la risoluzione di un nome simbolico (*www.polimi.it*) in modalità iterativa ed in modalità ricorsiva.

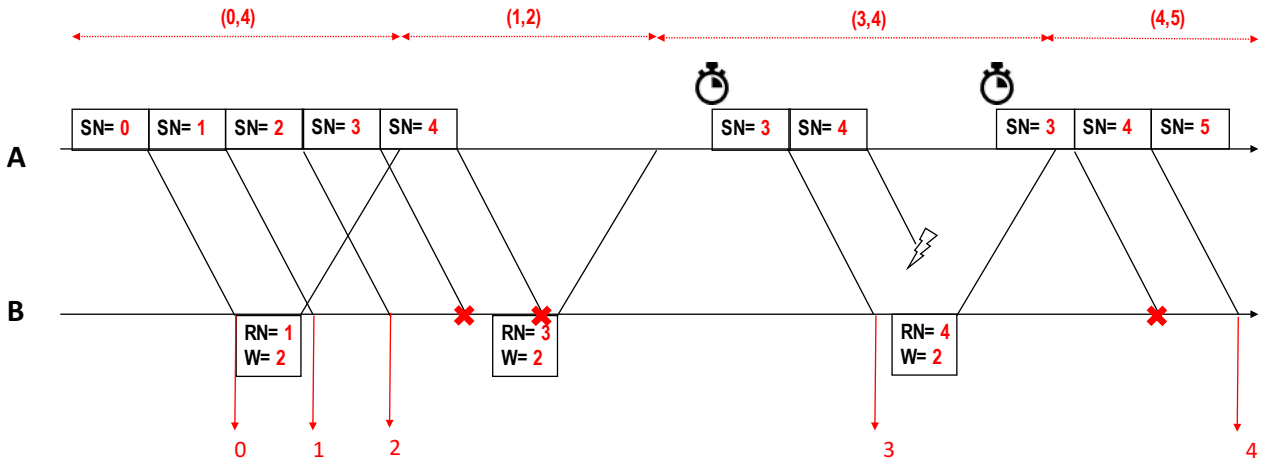
Modalità iterativa:
il client contatta il local NS,
il local NS contatta un root NS ed ottiene l'indirizzo di NS per il TLD .it,
il local NS contatta il NS per il TLD .it ed ottiene un NS per il dominio polimi.it,
il local NS contatta il NS per il dominio polimi.it ed ottiene l'indirizzo della macchina www.polimi.it,
il local NS resituisce al client l'indirizzo per la macchina www.polimi.it

Modalità ricorsiva:
il client contatta il local NS,
il local NS contatta un root NS
il root NS contatta un NS per il TLD .it
il NS per il TLD .it contatta un NS per il dominio polimi.it e riceve l'indirizzo della macchina www.polimi.it
l'indirizzo della macchina percorre catena di NS al contrario fino ad arrivare al local NS
il local NS resituisce al client l'indirizzo per la macchina www.polimi.it

Q2

Si completi la figura in cui è rappresentato un colloquio governato dai meccanismi Go-Back-N e controllo di flusso con il campo W. Il buffer in ricezione ha una capacità massima di 2 pacchetti, l'applicazione svuota il buffer ad ogni lettura ed invia un ACK. Si assuma una finestra iniziale al trasmettitore di 5 pacchetti.
Nel completamento si indichino con chiarezza:

- i valori corretti di SN, RN e W,
- gli estremi della finestra al trasmettitore, precisando gli instanti in cui essi cambiano sulla linea tratteggiata,
- i pacchetti accettati ed eventuali pacchetti scartati



Q3

Durante una sessione TCP, l'algoritmo di Jacobson stima valor medio e deviazione standard del RTT come $SRTT^0 = 10$ ms e $SDEV^0 = 2$ ms. I due segmenti successivi registrano un RTT di $RTT^1 = 16$ ms e $RTT^2 = 32$ ms. *Si indichino* nella tabella i valori di SRTT, SDEV, DEV e del Timeout alla ricezione di ciascuno dei due segmenti considerando $(1 - \alpha) = 7/8$ come peso della stima precedente di RTT e $(1 - \beta) = 3/4$ come peso della stima precedente di SDEV.
Si usi la tabella per indicare i risultati finali e lo spazio sottostante per mostrare i conti fatti.

	RTT	SRTT	DEV	SDEV	Timeout
$SRTT^0 = 10$ $SDEV^0 = 2$	$RTT^1 = 16$	$SRTT^1 = 10.75$	$DEV^1 = 6$	$SDEV^1 = 3$	$T^1 = 22.75$
	$RTT^2 = 32$	$SRTT^2 = 13.40$	$DEV^2 = 21.25$	$SDEV^2 = 7.56$	$T^2 = 43.46$

$SRTT^1 = \frac{7}{8}SRTT^0 + \frac{1}{8}RTT^1 = 10.75;\ SRTT^2 = \frac{7}{8}SRTT^1 + \frac{1}{8}RTT^2 = 13.40$
$DEV^1 = RTT^1 - SRTT^0 = 6;\ DEV^2 = RTT^2 - SRTT^1 = 21.25$
$SDEV^1 = \frac{3}{4}SDEV^0 + \frac{1}{4}DEV^1 = 3;\ SDEV^2 = \frac{3}{4}SDEV^1 + \frac{1}{4}DEV^2 = 7.56$
$T^1 = SRTT^1 + 4SDEV^1 = 22.75;\ T^2 = SRTT^2 + 4SDEV^2 = 43.64$

Laboratorio (6 punti

Q1

L'utente con indirizzo di posta *studente@labfir.lan* si connette al proprio server POP per scaricare la posta ricevuta con credenziali username: *studente* e password: *bravo*. *Si completi* la seguente sequenza di comandi

```
user@pc:~$ telnet localhost 110
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
+OK Hello Jedi, may the Force be with
you!
USER studente
+OK
PASS bravo
+OK Logged in.
LIST
1 370
2 458
3 498
.
RETR 3
+OK 498 octects
Return-Path: professore@corsofir.it
Delivered-To: studente@labfir.lan
Received: from server_corsofir.it
(localhost [127.0.0.1])
      by www.labfir.lan (Postfix) with
SMTP id E99C683BB
      for <destination>; Thu, 28 Apr
2016 13:55:33 +0200 (CEST)
Subject: Esame FIR
Message-Id: 201604.E99@www.labfir.lan
Date: Thu, 28 Apr 2016 13:55:33 +0200
(CEST)
From: professore@corsofir.it

In teoria non c'è differenza tra teoria e
pratica, in pratica c'è.
.
DELE 3
+OK Marked to be deleted.
LIST
1 370
2 458
.
RSET
+OK
LIST
1 370
2 458
3 498
.
QUIT
+OK Logging out.
Connection closed by foreign host.
```

Q2

Si vuole scrivere un'applicazione client/server UDP per il calcolo dei quadrati. Il client chiede all'utente di inserire un numero, il server risponde con il quadrato del numero e infine il client stampa la risposta. (**Hint!** Utilizzare le funzioni int() e str(): int() converte una stringa ricevuta in un numero intero per effettuare operazioni aritmetiche, str() converte un intero in una stringa da trasmettere). *Si completi* il codice del server

UDP client

```
from socket import *

serverName = 'localhost'
serverPort = 12000

clientSocket = socket(AF_INET, SOCK_DGRAM)

message = raw_input('Inserisci un numero')
clientSocket.sendto(message, (serverName,
serverPort))

reply, serverAddress =
clientSocket.recvfrom(2048)
print reply

clientSocket.close()
```

UDP server

```
from socket import *

serverPort = 12000

serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))

print "The server is ready to receive"

...DA COMPLETARE
```

```
while 1:
   message, clientAddress = serverSocket.recvfrom(2048)
   print "Datagram from: ", clientAddress
   modifiedMessage = str(int(message)*int(message))
   serverSocket.sendto(modifiedMessage, clientAddress)
```

Q3

Data la seguente coppia di script si indichi:

a) quale è il client e quale il server
b) cosa fa l'applicazione implementata

Script A

```
from socket import *

serverName = 'localhost'
serverPort = 12000

clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))

clientSocket.send('Ho vinto?')

reply = clientSocket.recv(1024)
print 'From Server:', reply

clientSocket.close()
```

Script B

```
from socket import *

serverPort = 12000

serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind(('', serverPort))
serverSocket.listen(1)
print 'The server is ready to receive'
client_num = 0
while True:
    connectionSocket,  clientAddress =
serverSocket.accept()
    client_num = client_num + 1
    print "Connection from: ", clientAddress
    sentence = connectionSocket.recv(1024)
    if client_num == 1337:
        connectionSocket.send('Hai vinto!')
        client_num = 0
    else:
        connectionSocket.send('Ritenta! :(')
connectionSocket.close()
```

a) Lo script A implementa il client, lo script B implementa il server

b) Il client invia al server la stringa 'Ho vinto?', il server restituisce 'Hai vinto!' a un visitatore ogni 1337, altrimenti restituisce 'Ritenta! :('.