# *Fondamenti di Internet e Reti*

#### Proff. A. Capone, M. Cesana, I. Filippini, G. Maier

**Laboratorio – 25 Luglio 2016**

|  |  |
| --- | --- |
| **Cognome** |  |
| **Nome** |  |
| **Matricola** |  |

**Script A**

from socket import \*

def get\_GPS\_coord(): return (5,10)

serverName = 'localhost'serverPort = 12000clSock = socket(AF\_INET, SOCK\_STREAM)clSock.connect((serverName, serverPort))

# Invia coordinate GPS al server(latitudine,longitudine) = get\_GPS\_coord()clSock.send(str(latitudine))message = clSock.recv(2)if message == 'OK': clSock.send(str(longitudine))

# Legge dal server il numero di Pokemon nelle vicinanze da richiedere al servernumero\_pokemon = int( clSock.recv(100) )lista\_pokemon = []

# Richiede, uno alla volta, la lista dei pokemon al serverfor i in range(numero\_pokemon): clSock.send('GO') pokemon = clSock.recv(100) lista\_pokemon.append( pokemon )

print lista\_pokemon

clSock.close()

**Script B**

from socket import \*

def get\_list\_from\_GPS(latitudine,longitudine): return ["Pikachu","Charmander"]

serverPort = 12000servSock = socket(AF\_INET, SOCK\_STREAM)

servSock.bind(('', serverPort))servSock.listen(5)

print 'Server Pokemon GO pronto!'

while True: clSock, clAddr= servSock.accept() print "Connection form: ", clAddr

# Riceve dal client le coordinate GPS lat = int( clSock.recv(100) ) clSock.send("OK") long = int( clSock.recv(100) )

# Calcola la lista dei Pokemon

lista\_pkmn = get\_list\_from\_GPS(lat,long)

clSock.send(str(len(lista\_pokemon)))

for pokemon in lista\_pkmn: message = clSock.recv(2) if message == 'GO': clSock.send(pokemon)

clSock.close()

Client

Server

LAT

“OK”

LONG

“2”

“GO”

“Pikachu”

“GO”

“Charmander”

FASE 1: Client invia al server le proprie coordinate GPS

FASE 2: Il Server comunica al client il numero di Pokèmon nelle vicinanze

FASE 3: Il Client richiede iterativamente al Server il nome dei Pokèmon nelle vicinanze

Il codice riportato rappresenta una versione semplificata del videogioco Pokèmon GO.

Periodicamente l’applicazione dell’utente (client) invia le proprie coordinate GPS al server, che risponde con l’elenco dei Pokèmon presenti nelle vicinanze.

Il diagramma in figura mostra il protocollo applicativo, in 3 fasi, tra Client e Server.

**Q1** Completare il codice mancante nel Server e nel Client per implementare la fase 3 del protocollo

**Q2** Quanti utenti si possono accodare nel Server in attesa di essere serviti?

**Q3** Che protocollo di trasporto è utilizzato? Perché la comunicazione tra server e client richiede tale protocollo?