Università della Calabria
Esame di **Reti di Calcolatori** - Data: 14 luglio 2015 - Durata della prova: 2 ore e 30 minuti

Cognome e Nome		*Matricola*	

Esercizio 1

Utilizzando il package java.net si vuole realizzare un sistema che consenta l'esecuzione di *processi*. Ogni processo è suddiviso in un numero di operazioni elementari sequenziali comprese tra *1* e *k*. Il sistema è composto da tre tipologie di nodi: *n Utenti* che possono sottomettere processi; *k OpServer* che eseguono le operazioni elementari che compongono i processi ed *1 Master Node* che raccoglie i processi sottomessi dagli Utenti, identificando le operazioni elementari ed inoltrando quest'ultime ai relativi OpServer.

Ad esempio, se il Master Node riceve una richiesta di esecuzione di un processo *P1* composto da 3 operazioni *op1*, *op2 e op4*, deve eseguire i seguenti passi: contattare prima l'OpServer 1, chiedendo l'esecuzione di op1 (esecuzione simulata con una stampa su video); attendere il risultato dell'operazione (una stringa), da parte dell'OpServer 1; contattare sequenzialmente OpServer2 e OpServer4 allo stesso modo di OpServer 1; infine, dopo aver ricevuto il risultato da parte di OpServer4 notificare all'utente che ha sottomesso il processo la sua terminazione.

Il particolare, il **Master Node** svolge le sue funzioni restando in ascolto su due porte distinte:

- *Porta TCP 2000*, a cui un nodo utente può connettersi per sottomettere l'esecuzione di un processo P; il master node memorizzerà anche l'indirizzo IP del nodo utente per la successiva notifica di terminazione del processo.
- *Porta TCP 3000*, a cui un OpServer può connettersi per notificare la fine di un'operazione, inviando come parametri l'ID del processo a cui appartiene l'operazione eseguita, l'ID dell'OpServer, e il risultato dell'operazione;

Ogni nodo **Utente** rimane in ascolto sulla porta UDP 4000, sulla quale il master node può notificare l'utente della conclusione di un processo.

Infine, ogni **OpServer** rimane in ascolto sulla porta TCP 5000, sulla quale il master node può inviare l'ID di un processo e l'ID dell'operazione da eseguire.

La soluzione proposta dovrà comprendere i main per avviare il funzionamento dei diversi nodi.

Esercizio 2

Si descrivano, anche mediante figure opportunamente commentate, le differenze tra **crittografia a chiave pubblica e crittografia simmetrica**.

Esercizio 3

Si descriva, anche mediante figure opportunamente commentate, come funzionano i **Proxy HTTP**.

Esercizio 4

Si realizzi un *Web Service* che permette di ottenere alcune informazioni sugli esami di un corso di laurea.

In particolare, il servizio espone:

1. un metodo che, dati il *nome di un esame*, la *data in cui si è tenuto l'esame*, e *la matricola di uno studente* che ha partecipato all'esame, restituisce il *voto conseguito* da quello studente all'esame (un intero compreso tra 1 e 30).
2. un metodo che, data una *data*, restituisce il *nome dell'esame* previsto in quella data (si assuma che si tenga un solo esame al giorno).

come specificato nel file WSDL allegato.

Si implementi in Java una classe che implementa il servizio e si descrivano le principali fasi necessarie alla sua messa in opera.

Allegato all'esercizio 4

```
<wsdl:definitions targetNamespace="http://www.examples.com/wsdl/ExamsService">

   <wsdl:types>
      <schema targetNamespace="http://DefaultNamespace">
        <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>

        <complexType name="Data">
          <sequence>
            <element name="Giorno" type="xsd:int"/>
            <element name="Mese" type="xsd:int"/>
            <element name="Anno" type="xsd:int"/>
          </sequence>
        <complexType>

      <schema>
   <wsdl:types>


   <wsdl:message name="VotoStudenteRequest">
      <wsdl:part name="in0" type="xsd:string"/>
      <wsdl:part name="in1" type="tns1:Data"/>
      <wsdl:part name="in2" type="xsd:int"/>
   </message>

   <wsdl:message name="VotoStudenteResponse">
      <wsdl:part name="in0" type="xsd:int"/>
   </message>

   <wsdl:message name="EsameGiornoRequest">
      <wsdl:part name="in1" type="tns1:Data"/>
   </message>

   <wsdl:message name="EsameGiornoResponse">
      <wsdl:part name="in0" type="xsd:string"/>
   </message>


   <wsdl:portType name="ExamsService">

      <wsdl:operation name="VotoStudente" parameterOrder="in0 in1 in2">
         <wsdl:input message="impl:VotoStudenteRequest" name=VotoStudenteRequest/>
         <wsdl:output message="impl:VotoStudenteResponse" name=VotoStudenteResponse/>
      </operation>

      <wsdl:operation name="EsameGiorno" parameterOrder="in0">
         <wsdl:input message="impl:EsameGiornoRequest" name= EsameGiornoRequest/>
         <wsdl:output message="impl:EsameGiornoResponse" name= EsameGiornoResponse/>
      </operation>

   </wsdl:portType>


   <wsdl:binding ...>
      ...
   </wsdl:binding>

   <wsdl:service ...>
      ...
   </wsdl:service>

</wsdl:definitions>
```