DIMES

Esame di **Reti di Calcolatori** del Corso di Laurea in Ingegneria Informatica
Prova scritta del 04 settembre 2019 – Durata: 2 ore e 30 minuti

Cognome e nome:	Matricola:

Seconda parte

Esercizio 3 (fino a 15 punti)

Si realizzi un'applicazione di rete in Java per la gestione delle prenotazioni presso una clinica sanitaria. Il sistema è composto dalle seguenti entità:

- 1 nodo ***Server***, che gestisce le richieste di prenotazioni per alcuni esami clinici;
- n nodi ***Paziente*** che inviano le loro richieste al server.
- 1 nodo ***Direzione***, che gestisce le statistiche sugli esami clinici svolti in una giornata.

Il Server gestisce le prenotazioni per diversi esami clinici. Ogni esame è identificato da un codice. Per ogni esame, la clinica mette a servizio 2 diversi *medici*, i quali si dividono i pazienti. Ogni *Medico* è indentificato da una matricola. Per ogni esame vengono accettati al massimo 20 pazienti al giorno (10 per ogni medico).

Il server offre il servizio di prenotazione degli esami clinici per un periodo di 4 ore nell'arco di una giornata, dalle ore 8 alle ore 12. Per tutte le richieste giunte al di fuori di questo intervallo temporale viene inviato un apposito messaggio "*service not available*" al *Paziente* che tenta la prenotazione.

Il *Paziente* che intende prenotare un esame clinico invia una richiesta di prenotazione sulla **porta TCP 3000** con una stringa contenente il codice dell'esame richiesto. Il Server controlla se nella giornata odierna ci sono posti disponibili per quel tipo di esame e, in caso affermativo, invia una stringa di *prenotazione* contenente: il codice dell'esame, il numero progressivo della prenotazione per quell'esame e la matricola del medico al quale è stato assegnato.
Il *Paziente* può in qualsiasi momento decidere di annullare la propria prenotazione inviando una richiesta col codice della prenotazione e il codice dell'esame sulla **porta TCP 4000** del Server.

Se non ci sono più posti disponibili per l'esame clinico richiesto, il Server inserisce il paziente in una coda di attesa. Il paziente in coda attende che si liberi un posto per un tempo massimo di 1 ora, trascorso il quale la connessione socket (instaurata inizialmente col server) viene interrotta automaticamente. Se un posto per quell'esame si rende disponibile, il server invia i dettagli della prenotazione al primo paziente in coda per quell'esame, usando la stessa connessione socket instaurata in precedenza.

Non appena il servizio di prenotazione del server viene terminato (dopo le ore 12), vengono inviati alla direzione (hostname *direzione.unical.it*) sulla **porta UDP 5000** una serie di messaggi contenenti un oggetto *Statistica* per ciascun esame clinico, che contiene: il codice dell'esame, il numero di pazienti prenotati e il numero di pazienti rimasti in coda senza ottenere una prenotazione.

È richiesto di gestire le connessioni multiple da parte del Server. Si realizzino le classi Server, Paziente e Direzione che implementino le funzionalità sopra descritte. Inoltre, si realizzino due main: 1) il primo main crea e avvia il Server (con hostname *clinica.unical.it*); 2) il secondo main crea e avvia un Client che si collega al Server per richiedere la prenotazione di un esame clinico.

Esercizio 4 (fino a 5 punti)

Si realizzi un Web Service che permette di ottenere informazioni sui prodotti venduti da una catena di magazzini, identificati da un codice (univoco) e caratterizzati da nome, produttore e prezzo di vendita. Per ogni magazzino, identificato da un id univoco, vengono gestire le informazioni sui prodotti venduti. In particolare, il servizio espone:

1. un metodo che, dato il nome del produttore, restituisce il *Prodotto* più venduto di quel produttore tra i diversi magazzini.
2. un metodo che, dato un *id* magazzino, restituisce una lista contenente i 3 prodotti che hanno prodotto più incassi per quel magazzino.

come specificato nel file WSDL allegato.

Allegato all'esercizio 4

```
<wsdl:definitions targetNamespace="http://www.examples.com/wsdl/ProdottiService">
<wsdl:types>
<schema targetNamespace="http://DefaultNamespace">
<import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
<complexType name="Prodotto">
        <sequence>
                <element name="codice" type="xsd:string"/>
                <element name="nome" type="xsd:string"/>
                <element name="produttore" type="xsd:string"/>
                <element name="prezzo" type="xsd:double"/>
        </sequence>
<complexType>
<complexType name="ListaProdotti">
    <sequence>
        <element name="item" type="tns1:Prodotto" maxOccurs="unbounded" minOccurs="0" />
    </sequence>
</complexType>
<schema>

<wsdl:types>
        <wsdl:message name="ProdottoPiuVendutoRequest">
                <wsdl:part name="in0" type="tns1:string"/>
        </message>
        <wsdl:message name="ProdottoPiuVendutoResponse">
                <wsdl:part name="in0" type="tns1:Prodotto"/>
        </message>
        <wsdl:message name="ProdottiMaxIncassoRequest">
                <wsdl:part name="in0" type="xsd:string"/>
        </message>
        <wsdl:message name="ProdottiMaxIncassoResponse">
                <wsdl:part name="in0" type="tns1:Prodotto"/>
        </message>
</wsdl:types>

<wsdl:portType name="ProdottiService">
<wsdl:operation name="ProdottoPiuVenduto" parameterOrder="in0">
        <wsdl:input message="impl:ProdottoPiuVendutoRequest" name=ProdottoPiuVendutoRequest />
        <wsdl:output message="impl:ProdottoPiuVendutoResponse" name=ProdottoPiuVendutoResponse />
</operation>
<wsdl:operation name="ProdottiMaxIncasso" parameterOrder="in0">
        <wsdl:input message="impl:ProdottiMaxIncassoRequest" name=ProdottiMaxIncassoRequest />
        <wsdl:output message="impl:ProdottiMaxIncassoResponse" name=ProdottiMaxIncassoResponse />
</operation>
</wsdl:portType>

<wsdl:binding ...>
...
</wsdl:binding>
<wsdl:service ...>
...
</wsdl:service>
</wsdl:definitions>
```