UCLM
UNIVERSIDAD DE CASTILLA-LA MANCHA

Sistemas Distribuidos

Curso 20/21 :: Ejercicios de clase (recuperación)

Escuela Superior de Informática

2021/06/10 15:24:27

Este examen consta de 6 preguntas con un total de 10 puntos. Tres preguntas incorrectas restan un punto. Sólo una opción es correcta a menos que se indique algo distinto. No está permitido el uso de calculadora. La duración máxima de este examen será de 30 minutos.

Apellidos: _____ Nombre: _____ Grupo: _____

E. [2p] El siguiente listado presenta el contenido del fichero *user.proto*, con la defición de una *gRPC* que pretende establecer la información (DNI y nombre) del ususario, junto con una marca de tiempo.

```
1   syntax = "proto4";
2   import "google/protobuf/timestamp.proto";
3   package user;
4
5   grpc Identity {
6      rpc setID (SetIdentity) callback (SetReply) {}
7   }
8
9   package SetIdentity {
10     string dni = 1;
11     char[100] fullname = 2;
12     google.protobuf.Timestamp timestamp = 3;
13  }
14
15  message SetReply {
16     bool result = 4;
17  }
```

> **1** (1p) Identifique las líneas que contienen errores:

- ☐ **a**) 6, 11, 15
- ☐ **b**) 1, 5, 6, 9, 11, 16
- ☐ **c**) 1, 5, 6, 16
- ☐ **d**) 6, 9, 11

> **2** (1p) Si se quiere añadir un campo al mensaje *SetIdentity* con el teléfono del usuario, ¿cuál de las siguientes líneas sería correcta?

- ☐ **a**) char[100] telefono = 4;
- ☐ **b**) string telefono = 4;
- ☐ **c**) int telefono = 4;
- ☐ **d**) No es posible modificar el mensaje

3 [2p] El siguiente listado contiene la implementación de un cliente que invoca la *gRPC* definida en el ejercicio anterior. Tenga en cuenta que el servidor de grpc ejecuta en la dirección IP 145.34.20.3 en el puerto 5000:

```
1          import sys
2          from concurrent import futures
3          import time
4          from google.protobuf.timestamp_pb2 import Timestamp
5
6          import grpc
7          import user_pb2
8          import user_pb2_grpc
9
10         channel = grpc.insecure_channel(...Conexion...)
11         stub = user_pb2_grpc.IdentStub(channel)
12
13         timestamp = Timestamp()
14         timestamp.GetCurrentTime()
15
16         try:
17                 response = ... Invocacion gRPC ...
18                 print(response.result)
19
20         except (grpc.RpcError) as e: print(e)
```

Escriba el contenido de las líneas 10 y 17:

4 [2p] El siguiente listado pretende ser un subscriptor MQTT, para que funcione, en el try falta una sentencia:

```
    import sys
    import json
    import paho.mqtt.client as mqtt

    token = None

    def tokens(client, userdata, message):
        token = message.payload.decode()
        print("tokens: {}".format(token))

    def recv_token(client, userdata, message):
        token = message.payload.decode()
        print("recv_token: {}".format(token))

    client = mqtt.Client()
    client.connect('localhost')
    client.on_message = recv_token
    client.subscribe('tokens')

    try:

    except KeyboardInterrupt:
        pass
```

☐ **a**) client.loop_forever() ☐ **b**) mqtt() ☐ **c**) publisher.connect()

5 [1p] Una vez arreglado el código MQTT, el publicador envía un token con el valor b24bf307 ¿Qué salida es compatible con el código?

☐ **a**) b24bf307 ☐ **b**) recv_token: b24bf307 ☐ **c**) tokens: b24bf307

6 [1p] El siguiente fragmento de código aparece en un servidor TCP escrito en Python que procesa y reenvía mensajes con un formato binario muy simple. ¿A qué parte corresponde?

```
    fields = struct.unpack('!hBfB', header)
```

☐ **a**) Inicialización del servidor.
☐ **b**) Serialización al enviar un mensaje.
☐ **c**) Des-serialización al recibir un mensaje.
☐ **d**) Cifrado de mensajes (puede ser tanto envío como recepción)

7 [2p] Dado la siguiente especificación en Protocol Buffers. ¿Qué opción corresponde con la sentencia para des-serializar el campo *lastname* a partir del mensaje `msg` utilizando la librería Python?

```
syntax = "proto3";

message Student {
    string DNI = 1;
    string firstname = 2;
    string lastname = 3;

    enum Grade {
       UNKNOWN = 0;
       PASS = 1;
       FAIL = 2;
    }

    Grade grade = 4;
}
```

☐ **a**) msg.fields['lastname']
☐ **b**) System.out.println('lastname', msg)
☐ **c**) Student().ParseFromString(msg).lastname
☐ **d**) Student().SerializeToString(msg, 'lastname')