TSR: First Partial

This exam consists of 10 multiple choice questions. In every case only one answer is correct. You should answer in a separate sheet. If correctly answered, they contribute 1 point to the exam grade. If incorrectly answered, the contribution is negative: -0.33. So, think carefully your answers.

1. Which sentence about LAMP systems is correct?

A	The components in a LAMP system are usually deployed on Windows hosts.
B	A LAMP system usually follows a 3-layered architecture.
C	No component in a LAMP system can be replicated.
D	A complete LAMP system cannot be deployed in a single host.

2. One difference between cloud computing (CC) and highly available (HA) clusters is:

A	HA clusters replicate service components, CC doesn't use replication.
B	HA clusters are mainly used for data science applications, while CC is commonly used for deploying distributed services.
C	From a user point of view, CC is provided by external companies, while HA clusters are purchased and maintained by the company that uses them.
D	HA clusters usually follow an IaaS service model, while CC systems mostly use the SaaS service model.

3. One of the fundamental problems to be solved in distributed systems is to take care of faults. To this end, we should…

A	Send every message only once; otherwise, attackers may easily collect relevant information about our services.
B	Avoid fault detection mechanisms, since those mechanisms need a lot of resources and they are always unreliable.
C	Replicate service components, in order to guarantee that at least one instance remains available.
D	All of the above.

4. The main goal of middleware is…

A	To improve system security.
B	To boost system scalability.
C	To hide and solve many communication problems that may arise among service components.
D	To deal with failure management.

5. Considering this program:

```
// Files.js
const fs = require('fs');
if (process.argv.length<3) {
  console.error('More file names are needed!!');
  process.exit();
}
function handler(name) {
  return function(err,data) {
    if (err) console.error(err);
    else console.log('File '+name+': '+data.length+' bytes.');
  }
}
var files = process.argv.slice(2);
var i=-1;
do {
  i++;
  fs.readFile(files[i], 'utf-8', handler(files[i]))
} while (i<files.length-1);
console.log('We have processed '+files.length+' files.');
```

Let us assume that we have run that program using "node Files A B" and A is a text file with 234781 bytes, and B is a short text file with a length of 430 bytes. What is the output shown in that execution?

A	More file names are needed.
B	File A: 234781 bytes. File B: 430 bytes. We have processed 2 files.
C	"We have processed 2 files.", followed by a line per file, showing their name and size.
D	Nothing is printed, since the program aborts in its line 8 without showing any message.

6. Regarding the program shown in the previous question, it is true that...

A	It is using a promise for processing each file.
B	It aborts its execution in the body of the "handler" function, since functions cannot return functions.
C	It uses the fs.readFile function in a synchronous way in order to avoid problems when the names and sizes of files are shown.
D	The "handler" function provides a closure for adequately printing the name of each file.

7. In the central server mutual exclusion algorithm, it is true that...

A	It is a highly available algorithm, since all its agents are replicated by default.
B	It uses a request-reply synchronous pattern in order to get CS-entry permissions and a one-way asynchronous forwarding for releasing both the CS and its permission.
C	It uses a PUSH-PULL unidirectional asynchronous pattern to get CS-entry permissions and a PUB-SUB for releasing both the CS and its permission.
D	It uses a PUB-SUB asynchronous pattern in order to get CS-entry permissions and a synchronous request-reply for releasing both the CS and its permission.

8. Considering these programs to be run in the same computer...

```
// client.js
var zmq=require('zmq');
var rq=zmq.socket('dealer');
rq.connect('tcp://127.0.0.1:8888');
var i=1;  rq.send(''+i);
rq.on('message',function(req,rep){
  console.log("%s %s",req,rep);
  if (i==100) process.exit(1);
  rq.send((++i)+'');
});
```

```
// server.js
var zmq = require('zmq');
var rp = zmq.socket('dealer');
rp.bindSync('tcp://127.0.0.1:8888');
rp.on('message', function(msg) {
  var j = parseInt(msg);
  rp.send([msg,(j*3).toString()]);
});
```

The following sentences are true:

A	In the client, when the "req" value is printed, it is equal to the value of "i".
B	Each client sends 101 requests to the server before terminating its execution.
C	We cannot have two or more clients of this kind in the same computer. All but the first are aborted in their attempt to connect with the server.
D	None of the above.

9. Considering the programs shown in the previous question...

A	The client cannot send a new request until the reply to the previous request is received and processed.
B	Although DEALER-DEALER communication is possible, in this example the messages do not include a first empty delimiter. Without it, messages aren't delivered.
C	Those programs are useless since it is impossible to intercommunicate two processes using DEALER sockets in both programs.
D	The server can only process the requests sent by a single client.

10. Let us assume that a distributed application requires a unidirectional asynchronous communication channel between two components A and B, forwarding messages from A to B (i.e., A→B). To this end, its developer could use this ZeroMQ pattern...

A	A: PULL, B: PUSH.
B	A: REQ, B: REP.
C	A: SUB, B: PUB.
D	A: DEALER, B: DEALER.