NIST - First Partial. 2024-11-04

This exam is worth 10 points, and consists of 22 questions. Each question poses 4 alternatives and has only one correct answer. Once discarded the two worst answers, each correct answer earns 0.5 points, and each error deducts 1/6 points. You must answer on the answer sheet.

A

ALUMN@

1. ***In regard to the advantages provided by a distributed system, select the correct statement.***
   - **A.** It improves performance
   - **B.** It allows to improve availability
   - **C.** It facilitates resource sharing
   - **D.** All other statements are true

2. ***In regard to the concept of highly available clusters, these clusters:***
   - **A.** Sacrifice the availability of servers to maintain the integrity of the managed information
   - **B.** Require dedicated software, hardware, and specialized personnel, and are therefore expensive for companies.
   - **C.** None of the other statements are true.
   - **D.** Sacrifice the integrity of the managed information to maintain the availability of the servers

3. ***In regard to cloud computing***
   - **A.** Ideally, it presents the following tiered structure: SaaS, PaaS, IaaS
   - **B.** All other statements are true
   - **C.** It is possible thanks to the use of HW virtualization technology
   - **D.** It introduces a "pay as you go" model

4. ***Let us assume the following program, called copy.js***

```
const fs=require('fs')
function copyFile(source, dest) {
  console.log("reading")
  fs.readFile(source, (error, data)=> {
    if (!error) {
    console.log("writing")
    fs.writeFile(dest, data, error=>{
      if (!error) console.log("done")
    })
    }
  })
}
copyFile("copy.js","copy2.js")
console.log("start")
```

Assuming that reading and writing files do not generate errors (the file to be read exists, we have the necessary permissions, there is space to write the new file, etc.), choose the expected result when running the program

- **A.** Syntax error when trying to run the program
- **B.** It writes to the screen (in that order) the lines

```
reading
start
writing
done
```

- **C.** It writes to the screen (in that order) the lines

```
start
reading
writing
done
```

- **D.** It writes on the screen (`start`, `reading`, `writing`, `done`), but we cannot know the order (it may be a different order in each execution)

5. ***Let us assume the following program, called pcopy.js***

```
const fs=require('fs').promises
function copyFile(source, dest) {
  console.log("reading")
  fs.readFile(source)
  .then(data=>{console.log("writing");
         return fs.writeFile(dest, data)})
  .then(()=>console.log("done"))
}
copyFile("pcopy.js","pcopy2.js")
console.log("start")
```

Assuming that reading and writing files do not generate errors (the file to be read exists, we have the necessary permissions, there is space to write the new file, etc.), choose the expected result when running the program

A. It writes to the screen (in that order) the lines

```
reading
start
writing
done
```

B. It writes to the screen (in that order) the lines

```
start
reading
writing
done
```

C. It writes on the screen (`start`, `reading`, `writing`, `done`), but we cannot know the order (it may be a different order in each execution)

D. Syntax error when trying to run the program

6. ***Let us consider these client and server programs:***

```
***cliente
let net=require('net');
let sock=net.connect({port:3000}, ()=>{
  for (let i=1; i<=10; i++) sock.write('hello');
});
sock.on('data', (x)=>{console.log(""+x)});

***servidor
const net=require('net');
const server=net.createServer( (sock)=>{
   sock.on('data', (x)=>sock.write('OK '+x));
}).listen (3000);
```

Regarding these two programs, select the correct option:

A. The server finishes by responding to all messages sent by the client.

B. Neither the client program nor the server program terminates

C. None of the other statements are true.

D. The client terminates upon receiving all messages sent by the server.

7. ***Consider the following JavaScript program:***

```
function f(x) {
  return function (y) {
    x++
    return x+y
  }
}
g=f(5, 10)
console.log(g(3))
```

Indicate what that program would display on the screen during its execution.

A. 14

B. 63

C. 9

D. An error occurred because f() was called with more than one argument.

8. ***Consider the following JavaScript program:***

```
setTimeout(()=>{console.log("1")},10)
let k=Math.abs(parseInt(process.argv[2]))
let j=0
for (let i=0; i<k; i++) j+=i
setTimeout(()=>{console.log("2")}, 1)
```

Indicate what that program would display on the screen during execution if it is passed an integer numeric value as the first argument on the command line and none of the arithmetic operations generate an error.

A. The values 1 and 2, each on a line, and their order will depend on the value provided in the argument

B. Always a first line with a 1 and a second line with a 2

C. Always a first line with a 2 and a second line with a 1

D. An error, since you cannot use `setTimeout()` more than once in the same program

9. ***Consider the following JavaScript program:***

```
const net=require('net')
let server=net.createServer(
  function(c){
    console.log('server connected')
    c.on('end',function(){
         console.log('server disconnected')
    })
    c.on('data', function(data){
         console.log('data from client: '
                     + data.toString())
         c.write(data)
    })
})// End of net.createServer()
server.listen(9000,
  function(){
    console.log('server bound')
})
```

This program shows how a very simple server using a TCP socket could be written. How does the process running this program manage to service multiple connections from its clients if it only has a single thread to do so?

A. Allowing each established connection to be managed by a call to the function defined as the unique argument of `createServer()`

B. Managing message arrivals on those connections using `"data"` events

C. Using the event queue to sequence the execution of all the "listeners" that are activated during the use of those connections

D. All other statements are true

10. ***Consider the following JavaScript program:***

```
const net=require('net')
let server=net.createServer(
  function(c){
    console.log('server connected')
    c.on('end',function(){
         console.log('server disconnected')
    })
    c.on('data', function(data){
         console.log('data from client: '
                     + data.toString())
         c.write(data)
    })
})// End of net.createServer()
server.listen(9000,
  function(){
    console.log('server bound')
})
```

This program illustrates how a very simple server using a TCP socket could be written. What message will be displayed first (if there are no errors) by a process running this program?

A. `"server connected"`

B. `"server bound"`

C. Either `"server bound"` or `"server connected"`

D. Any of the following: `"data from client..."`, `"server connected"` or `"server bound"`

11. ***Consider the following client program that uses ZeroMQ to connect to two servers:***

```
const zmq=require('zeromq')
const rq=zmq.socket('req')
rq.connect('tcp://127.0.0.1:8888')
rq.connect('tcp://127.0.0.1:8889')
rq.send('Hello!')
rq.send('Hello again!')
rq.on('message', function (msg) {
  console.log('Response: '+msg)
})
```

Select the correct statement about how the program works:

A. If there is no server listening on port 8889, this client will not display anything on the screen, since its execution will block before the `rq.on()`

B. The code in this client sends the string `"Hello!"` to the server that manages the local port 8888 and the string `"Hello again!"` to the server that manages the local port 8889

C. This program will not work correctly, as it would need a second `rq.on('message'...)` to process the response from the second server

D. The execution of the process that runs this program ends as soon as the first response has been received from some server.

12. ***The components of a distributed system:***
   A. All other statements are true
   B. ... must be able to be specified and developed independently.
   C. ... must be able to be deployed autonomously
   D. ... must be able to interact in a simple and useful way

13. ***Standards make it easier to overcome complexities, but they do NOT include***
   A. Specifying the functionality of services
   B. The format of the transmitted information
   C. Synchronization between client and server
   D. Differentiation of roles played in a computer system

14. ***Regarding messaging systems:***
   A. In persistent versions, the receiver has a buffer for messages, which can be sent even after the sender has finished
   B. Non-persistent versions with a broker have no overhead for using that broker
   C. In non-persistent versions without a broker, the sender and receiver must keep the messages in memory
   D. In non-persistent versions, the message can only be transmitted if the receiver is active.

15. ***Using ZeroMQ, the code for two components has been developed, which allows two computers on an IP network to communicate. Changes are going to be made, and most of the previous code can be reused... :***
   A. If the change does not imply that those components will run on the same computer
   B. None of the other statements are correct.
   C. Only if the programming language of both components is the same
   D. If the change does not imply that these components become threads of the same process

16. ***One of the advantages of using ZeroMQ to program communicating components is:***
   A. You can choose different languages to program those components
   B. Sockets and the semantics of the operations that accompany them are no longer needed.
   C. You can communicate ZeroMQ components with others that use the NodeJS `net` module
   D. You can communicate ZeroMQ components with others using the NodeJS `http` module

17. ***Suppose that on a computer X we have three components A, B and C that use ZeroMQ. Choose which statement is true:***
   A. If A and B are processes that communicate with each other, two ZeroMQ libraries are needed on machine X
   B. If A and B communicate with each other, and C communicates with a component on another computer Y, then two ZeroMQ libraries are needed on computer X.
   C. If A, B and C are threads of the same process communicating with each other, three ZeroMQ libraries are needed on machine X
   D. Only one ZeroMQ library is needed on machine X, regardless of the number of communicating components

18. ***Consider the following JavaScript program:***

```
for (let i=0; i<10; i++) {
  var j=i;
  setTimeout(()=>console.log("j = "+j), j*1000);
}
// Final del programa
```

   In this code, 10 events are programmed, using the variables `i` and `j`. About these variables we can say that:
   A. At the beginning of the program, no variable is declared
   B. At the end of the program, j has the value 0, and this sequence is printed: `"j = 0"`, `"j = 1"`, `"j = 2"`... `"j = 9"` (on separate lines)
   C. None of the other statements are true.
   D. At the end of the program, no variable is declared (present in scope), and `"j = 9"` is printed several times.

19. ***Given the following code, choose the correct statement:***

```
let fg
{
  let x=1
  fg= (y)=>{console.log("hello "+y+" "+ x++)}
}
for (let i=0; i<10; i++) fg("pp")
```

   A. The code is incorrect because function `fg` cannot be used outside its block
   B. When running the program, 10 identical lines will be printed on the console.
   C. The variable `x` is part of a closure
   D. There are no closures, as there are no functions that return functions.

20. ***Consider the following program, similar to the emitterX.js programs used in Lab 1:***

```
const ev=require('events')
const emitter=new ev.EventEmitter()
const e1='e1', e2='e2', e3='e3'
let inc=1

function handler(e,n) {
  return (inc)=> {
    n+=inc
    console.log(e + ' --> ' + n)
  }
}

emitter.on(e1, handler(e1,1))
emitter.on(e2, handler(e2,'2'))
emitter.on(e3, handler(e3,3))
emitter.on(e3, handler(e3,'3'))

function phase() {
  emitter.emit(e1,inc)
  emitter.emit(e2,0)
  inc++
}

setInterval(phase,1000)
setTimeout(process.exit,2500)
```

What is the output on the screen when running the program?

A.
```
e1 --> 2
e2 --> 20
e3 --> 4
e3 --> 31
e1 --> 4
e2 --> 200
e3 --> 6
e3 --> 312
```

B.
```
e1 --> 2
e2 --> 20
e1 --> 4
e2 --> 200
```

C.
```
e1 --> 2
e2 --> 21
e1 --> 4
e2 --> 212
```

D.
```
e1 --> 2
e2 --> 21
e3 --> 4
e3 --> 31
e1 --> 4
e2 --> 212
e3 --> 6
e3 --> 312
```

21. ***In the final part of practice 1, a question is formulated in which, given a pool of servers, the proxy must send the request to the least loaded server in that pool.***

A. It can be achieved by simply modifying the programmer.js code

B. It can be achieved without adding or modifying code on the server computers

C. It can be achieved by simply modifying the code in `ProxyProg.js` (or `ProxyConf.js`)

D. Code will need to be added (or modified) on the server machines

22. ***To implement the programmable proxy from Practice 1...***

A. It is no longer necessary to create a specific socket to communicate with the server (`serverSocket`)

B. We use `JSON.stringify` and `JSON.parse` to encode/decode messages between the client and the proxy

C. In the proxy we must call `net.createServer` twice: once to define the code that serves the client, and another to define the code that serves the server.

D. In the proxy we must call `net.createServer` twice: once to define the code that serves the client, and another to define the code that serves the programmer.