NIST - Rec_First Partial. 2025-01-30

This exam is worth 10 points, and consists of 17 questions. Each question poses 4 alternatives and has only one correct answer. Once discarded the worst answer, each correct answer earns 10/16 points, and each error deducts 10/48 points. You must answer on the answer sheet.

A

1. ***This application can be considered a distributed service when deployed in the mentioned environment:***
   - **A.** A Java compiler when used on all computers in the same laboratory to solve the same practice of a programming subject
   - **B.** Microsoft Word in Office 365 when used by multiple users on their respective computers to edit the same document shared among all of them
   - **C.** The bash command interpreter when used by a user on his computer to launch applications locally, without using the network in them
   - **D.** All the statements are true

2. ***By studying the evolution of software services, we can identify a stage in which its service model automates the monitoring of relevant parameters, allows the expression of elasticity points and automates the reconfiguration of the service based on the existing load. This service model is:***
   - **A.** SaaS
   - **B.** PaaS
   - **C.** Highly available cluster
   - **D.** IaaS

3. ***When analyzing programming paradigms, it is indicated that for a service to be scalable, its servers should not become blocked when handling each request. This can be provided by adopting this paradigm:***
   - **A.** Multi-threaded, since although its threads can share resources, this will not lead to the blocking of any activity
   - **B.** Event-driven, associating a thread with each event and sharing memory between all of them
   - **C.** Multi-threaded, since the multiple threads that could be generated will never share resources
   - **D.** Asynchronous, or event-driven, since the new events generated can be kept in a queue, without interrupting or blocking the activity in progress

4. ***What is displayed on the screen when you run the following program?***

```
const k=2
if (k==1)
  console.log("k=1")
else {
  var j=3
  console.log("j-k=" + (j-k))
}
console.log("j="+j)
```

   - **A.** This content: `j-k=1, ...Reference error: j is not defined...`
   - **B.** An error indicating that a constant cannot be subtracted from a variable
   - **C.** Two lines with: `j-k=1, j=3`
   - **D.** Two lines with: `k=1, j=undefined`

5. ***What is displayed on the screen when you run the following program?***

```
const k=2
if (k==1)
  console.log("k=1")
else {
  let j=3
  console.log("j-k=" + (j-k))
}
console.log("j="+j)
```

   - **A.** An error indicating that a constant cannot be subtracted from a variable
   - **B.** Two lines with: `j-k=1, j=3`
   - **C.** Two lines with: `k=1, j=undefined`
   - **D.** This content: `j-k=1, ...Reference error: j is not defined...`

6. ***What is displayed on the screen when you run the following program?***

```
function greet( x="John", y ) {
  console.log("Hi, "+x); console.log("Hi, "+y)
}
greet(undefined, "Mary", "Peter")
```

A. `Hi, Mary`
`Hi, Peter`

B. `Hi,`
`Hi, Mary`

C. `Hi, John`
`Hi, Mary`

D. An error, since the `greet` function expects two arguments and we are passing it three

7. ***What message does this program display first, how many callbacks are used in it, and how many closures?***

```
function generateF(x) {
  return function () {
    console.log("Writing after "+x+" seconds.")
  }
}
setTimeout(generateF(0), 0)
console.log("End!")
```

A. `Writing after 0 seconds`, with one callback and one closure

B. `End!`, with one callback and one closure

C. `Writing after 0 seconds`, no callbacks and no closures

D. `Writing after 0 seconds`, no callbacks and with one closure

8. ***Consider that this program is started on a computer that already has a net server running that listens for connections on port 9000 and that the latter always returns a response to any request received. Select the true statement about this client program:***

```
const net=require('net')
let counter = 0
let client = net.connect({port:9000},
  function () {
    console.log("client connected!")
    client.write(counter+' world')
  })
client.on('data', function (data) {
  console.log(data.toString())
  if (counter == 9) client.end()
  else client.write((++counter) +' world')
})
client.on('end', function () {
  console.log("client disconnected")
})
```

A. The code used in this client to send its requests guarantees weak persistence in the resulting communication

B. This client terminates after receiving the response to its tenth request

C. If the first argument of the `on` method were replaced in the eighth line by the string `message`, the program would continue to function in the same way

D. This client is unable to interact with any server, since it does not send any message to it

9. ***This is an example of middleware:***

A. Ubuntu 24.10

B. Node.js 22.12.0

C. LibreOffice Writer 24.8

D. Apache ActiveMQ Classic 6.1.4

10. ***Select the correct statement about non-persistent communication systems:***

A. If they use a broker, they are more efficient than when they do not need one.

B. If they do not use a broker, they are more efficient than when they need one.

C. They usually keep messages in the sender's queues when the receiver is not available.

D. They require that the receiver be prepared for the sender to transmit its messages.

11. ***If `so` is a ZeroMQ socket of type REQ, then when performing the operation `so.send("Example")`:***

A. That sending will be blocked until a request is received from another REP socket beforehand, since REQ only serves to respond and cannot initiate communication

B. When transmitting the message to the receiver, it will have two segments: the identity of `so` and `"Example"`

C. The message `"Example"` will necessarily remain in some output queue for a long interval, since the REQ-REP pattern is synchronous and blocking

D. When transmitting the message to the receiver, it will have two segments: `""` and `"Example"`

12. ***The aim is to develop, using ZeroMQ sockets, a news distribution service consisting of three components:***
(a) a correspondent, specialised in a particular topic and who will broadcast his messages using two segments (topic and text of the news),
(b) an agency, which will receive the information from the correspondents and forward it to the interested network or networks, and
(c) a network, which, through a periodic process, will connect to the agency at a certain interval to receive information of interest to its potential users. The same network may be interested in different topics.
Communication will be unidirectional: the agency never responds to the correspondent, nor does the network respond to the agency. The information sent by a correspondent must not be lost if the agency has not yet started its activity. On the contrary, the network is only interested in the information that the agency publishes from the moment it makes its connection to the agency, discarding the messages that the agency has previously published. What types of sockets could be used to develop this service?

A. Correspondent: PUSH; Agency: SUB and PUB; Network: PULL

B. Correspondent: REQ; Agency: REP and PULL; Network: PUSH

C. Correspondent: PUB; Agency: SUB and REQ; Network: REP

D. Correspondent: PUSH; Agency: PULL and PUB; Network: SUB

13. ***Consider the following program:***

```
const zmq = require("zeromq")
const sub = zmq.socket('sub')

sub.connect("tcp://localhost:5555")
sub.on("message", function(msg) {
  console.log("Received: " + msg)
})
```

A subscriber process has been started that runs this program and on that same computer a process that uses a PUB socket `p` has been started previously, on which the operation `p.bind("tcp://*:5555")` has been successfully performed. This publisher process emits a message every second, uninterruptedly, whose content is short text news, between 50 and 80 characters. However, the subscriber process fails to receive or display any message. Why?

A. The management of the connections is wrong, because the SUB sockets must perform `bind` and the PUB sockets `connect`

B. The event to use to receive the messages must be `"data"` instead of `"message"`

C. The publisher program must use `"tcp://localhost:5555"` as the argument to the `bind` operation, instead of the used URL

D. The subscriber program must include a `sub.subscribe("")` statement to receive messages

14. ***A process A sends six messages using a PUSH socket, with a `bind("tcp://localhost:8888")` and `setInterval(...,1000)`, whose content is, respectively, `"1", "2", "3", "4", "5"` and `"6"`, before being terminated by the user. After being started and before the first second has elapsed, three processes B, C and D have connected, in that order, to that port 8888 and receive messages with a PULL socket. What messages does each process receive?***

A. B: "1" and "4"; C: "2" and "5"; D: "3" and "6"

B. B, C and D each receive all the messages

C. The set of messages that each one will receive is unpredictable

D. None, since they should have used another type of socket to enable message reception

15. ***Given the following program:***

```
function f1 (a,b,c) {
  return a+b+c
}
let vector = [1,2,3,4]
console.log ("resultv1: " + f1 (...vector))
```

Choose the message on the screen that shows its execution:

A. `resultv1: 1,2,3,4undefinedundefined`

B. `Uncaught ReferenceError: ...vector is not defined`

C. `resultv1: 6`

D. `resultv1: [1,2,3,4]`

16. ***In the second session of practice 1, we were asked to extend a pair of netClient.js and netServer.js programs so that the second returned to the first a value proportional to its load at that moment. To do this, the extended version of netClient.js needed to receive two arguments from the command line: the server IP and the client IP, in that order. With what instructions could we access those arguments?***

A. All options are correct

B. `let args = process.args.slice(2)`
`let ipServer = args[0]`
`let ipClient = args[1]`

C. `let ipServer = process.argv[2]`
`let ipClient = process.argv[3]`

D. `let pa = argv`
`let ipServer = pa[2]`
`let ipClient = pa[3]`

17. ***In the third session of practice 1, we were asked to review or implement three types of proxies: basic, configurable and programmable. Any of them received the messages issued by a web browser (or any other client process that used TCP connections) to forward them to a certain server (usually a website). What is the main difference between basic and configurable?***
    - **A.** Basic only allows you to manage one server, while configurable can interact with two servers simultaneously.
    - **B.** Basic does not manage any cache, while in configurable we can configure the size of the cache of recent pages that it will maintain.
    - **C.** The basic code keeps the server address and port constant, while configurable receives that information from the command line.
    - **D.** Basic receives the server address and port from the command line, while configurable receives that information in a configuration message.