TSR **Recovery** **2023-02-03** **A**

Deliver the answers to the parts you need. Leave the others empty.

1. *16 multiple choice questions for partial 1*
2. *2 short open-ended exercises for Lab 2*
3. *16 multiple choice questions for partial 2*

Each multiple-choice question raises 4 alternatives and has a single correct answer. Each correct answer contributes 10/16 points, and each error deducts 10/48 points. You must answer the multiple-choice questions on the answer sheet.

PARTIAL 1

1 *Highly available clusters are an example application area of distributed systems where…:*

- **a** Replication is used to improve service throughput and availability.
- **b** Service responsibility is distributed among 'client' nodes, making it cooperative and easily scalable.
- **c** All other options are true.
- **d** Its service model automates the deployment of distributed applications.

2 *In cloud computing, the main goal of the SaaS service model is:*

- **a** Under a pay-per-use model, provide the necessary infrastructure and manage it properly.
- **b** Automate the tasks of service deployment and infrastructure management to its clients.
- **c** Offer a set of distributed applications already developed, for customers to acquire and deploy them wherever they prefer.
- **d** Offer remote computing services to its clients, so that the latter do not have to worry about application deployment or infrastructure management.

3 *Wikipedia is a good case study in Unit 1 because:*

- **a** It is a distributed service that illustrates all stages of service evolution, e.g. it used 'mainframes' in its early stage and is a SaaS in its current stage.
- **b** It has been programmed in Node.js (JavaScript) as well as the projects to be developed in the TSR labs.
- **c** All other options are true.
- **d** It uses multiple mechanisms to make a distributed service scalable and provides some documentation on them.

Let us consider this set of JavaScript programs:

```
// Program: ex1.js
const ev = require('events')
const emitter = new ev.EventEmitter()
const e1 = "print"
emitter.on(e1, function(num) {
  return () =>
    console.log("Event " + e1 + ": " + ++num)
}(0))
emitter.emit(e1)
```

```
// Program: ex2.js
function f(x){
  return function (y) {
    let z = x + y
    return z
  }
}
const f2 = f(10)
```

4 *In the execution of 'ex1.js', if the initial thread of execution is not considered, how many times will there be any 'listener' of e1 in the event queue?*

- **a** None of the other options are true.
- **b** An indefinite number of times, since event e1 is generated repeatedly.
- **c** Once.
- **d** None.

5 *What is the scope of the variable* `z` *in the program 'ex2.js'?*

- **a** None, because its declaration is wrong.
- **b** Global.
- **c** Local to the anonymous function contained in the function f.
- **d** Local to function f.

6 *Is any closure defined in the programs 'ex1.js' and 'ex2.js'?*

- **a** None.
- **b** Yes, but only in 'ex2.js'.
- **c** Yes, one in each program.
- **d** Yes, but only in 'ex1.js'.

7 *The following code snippet writes some content to 3 files and then reads them using different versions of the read file API.*

Assuming that we execute the program and that the writing of the 3 files is completed without errors, indicate the correct statement:

```
const fs=require('fs')
const fsp=fs.promises;

// Write the content "fb", "fc" and "fd" to
// files "fb", "fc" and "fd" respectively
let write = (x) => fs.writeFileSync (x, x)
write("fb"); write("fc"); write("fd")

// Read from the files
fsp.readFile("fb").then ( data =>
  console.log(data + ""))
fs.readFile("fc", (err,data) =>
  console.log(data + ""))
console.log(""+fs.readFileSync("fd"))
```

- **a** It is possible that when executing the program, we see the contents of the files in this order: fb, fc, fd
- **b** When executing the program we will only see the content of the files 'fc' and 'fd'. The content of 'fb' will not be seen.
- **c** It is possible that when executing the program, we see the contents of the files in this order: fd, fc, fb
- **d** When executing the program we will only see the content of the files 'fb' and 'fc'. The content of 'fd' will not be seen.

8 *In Unit 3, it is stated that 'The content of messages is transparent to ØMQ', but…*

- **a** It is not possible to send information of several types in the same message
- **b** It is possible to send information of various types even in the same segment
- **c** The responsibility for interpreting the types and content of a message rests with the participants
- **d** You can only use multiple segments in a message if you need to include data of multiple types

9 *In ØMQ, how could two different publisher processes broadcast information to multiple subscribers if the latter only use one SUB socket each?*

a This is not possible because there would be two bind operations for the same endpoint that subscribers connected to.

b It is a facility exclusively applicable to the case in which a single subscriber binds and all publishers connect (to the subscriber's url).

c Each subscriber should perform multiple connects: one per publisher.

d This can be achieved if the url used by subscribers includes a regular expression (e.g. an asterisk) in order to reference all publishers involved.

10 *In ØMQ:*

a REQ sockets only have a message send queue.

b REP sockets only have a queue for receiving messages.

c Both REQ and REP sockets have message send and receive queues.

d Of the other three responses, only two are correct.

Let us consider these two programs:

```
// File: sender.js
const zmq = require('zeromq')
const push = zmq.socket('push')
const NUM_MSGS = 10
const DELAY = 1000
push.connect('tcp://127.0.0.1:8000')
let count = 0
function sender() {
  push.send("Message number " + ++count+
    " from sender " + process.pid)
  if (count < NUM_MSGS)
    setTimeout( sender, DELAY )
  else push.close()
}
sender()
```

```
// File: receiver.js
const zmq = require('zeromq')
const pull = zmq.socket('pull')
pull.bind('tcp://127.0.0.1:8000', (err) => {
  if (err) {
    console.log("Error on bind()")
    process.exit(1)
  }
})
pull.on('message', (msg) => {
  console.log(msg+"")
})
```

11 *Let us consider the original sender.js and receiver.js programs. Let us assume that port 8000 is not bound.*

We start a sender process that runs the program sender.js and after five seconds another receiver process (receiver.js) is started on that same computer.

What happens to the messages sent on that communication channel?

a All are lost, since the sender never connects with the receiver.

b They are all delivered to the receiver and the receiver displays the contents of those ten messages on the screen as they are received.

c The first five or six messages are lost, depending on the specific moment in which the connection between both processes can be established.

d None of the other options are correct.

12 *In the programs sender.js and receiver.js.*

How many messages would the receiver display on the screen if we replaced the sender's PUSH socket with a REQ and the receiver's PULL socket with a REP, without making any other changes to both programs?

Assume that the new sender and receiver are started at the same time on the same machine and that port 8000 was not bound before starting both processes.

a None, since the sender never connects with the receiver.

b The receiver shows the content of those ten messages on the screen, since it gets to receive each one in due time.

c Surely it shows from the second to the tenth message and maybe the first one too, but this depends on the instant the processes connect.

d None of the other options are correct.

13 *Consider the following JavaScript program:*

```
for (var i=0; i<5; i++) {
  console.log("i: " + i)
}
console.log("end --> i=" + i)
```

What changes would there be in the execution of the program if we replaced the keyword 'var' with 'let' in its first line?

a The program would behave the same way, since 'i' is still a global variable.

b The program would generate an error on its last line.

c The program would generate an error on its first line, since 'let' cannot be used in 'for' loops.

d None of the other options are correct.

14 *What would the execution of the following JavaScript program display on the screen?*

```
function f1 (a,b,c) {
  console.log(arguments.length +
    "arguments")
  return a+b+c;
}
console.log("result: " + f1(1,"a",[3,0],4,5))
```

a An error.

b 5 arguments
result: 1a3,0

c 6 arguments
result: 1a3

d None of the other options are correct.

15 *Consider the following JavaScript program:*

```
const fs=require("fs")
console.log("Call to readFile")
fs.readFile("a.txt",(e,d)=> {
  if (e) console.error("readFile error")
  else console.log(d+'')
})
console.log("End of readFile")

console.log("Call to readFileSync")
try {
  console.log( fs.readFileSync("a.txt")+ "")
} catch(e) {}
console.log("End of readFileSync")
```

What will be the last string that this program will display, if the file it is trying to read does not exist?

a 'End of readFile'

b 'readFile error'

c 'End of readFileSync'

d None of the other options are correct.

16 *This is the initial program to complete in the 'emisor3.js' example from Lab 1:*

```
...
const e1='e1', e2='e2'
let inc=0, t
function rand() { // will return random value
// in range [2000,5000) (ms)
... // Math.floor(x) integer part of x-value
... // Math.random() random value range [0,1)
}
function handler (e,n) { // e is the event, n
  // the associated value
  return (inc) => {..} // listener receives a value
}
emitter.on(e1, handler(e1,0))
emitter.on(e2, handler(e2,''))
function stage() {
  ...
}
setTimeout(stage,t=rand())
```

In this exercise, an indefinite series of stages had to be programmed, with a random duration between 2 and 5 seconds. If instead of having infinite stages, it was requested to generate 10 stages and each stage did not need to display its duration on the screen,

Would it be valid to replace the last line of the program with this block?

```
t = 0
for (let i=0; i<10; i++) {
  t = t + rand()
  setTimeout(stage, t)
}
```

a Yes.

b No, because the first argument to setTimeout, both in the original and in this new code, should be stage() instead of stage.

c No, because it is not possible to make the duration of each stage different and random.

d No, because the variable i should have been declared with var instead of let.

PARTIAL 2

17 *We have a client.js program that uses an s1 socket of type DEALER to interact,* **using messages with a single segment**, *(e.g.,* `s1.send('request')`*) with another server.js program that uses an s2 socket of type ROUTER.*

Communication is carried out without problems if we run a process of each type.

Some time later it was decided to make a single change in those programs, specifically in client.js: replace `s1=zmq.socket('DEALER')` *with* `s1=zmq.socket('REQ ')`.

When checking the operation of the new client.js, it is observed that server.js only receives apparently empty messages.

It behaves in that way because"

a When creating a REQ-ROUTER channel, the REQ socket should have large sending queues. Otherwise, the message content is lost at sending time.

b The ROUTER socket, if connected with a REQ socket, automatically discards the first segment of each received message.

c The REQ socket prepends an empty segment when sending messages.

d None of the other statements is true.

18 *You want to develop and deploy a distributed application in which there will be multiple instances of component A and at least one instance of component B.*

A will initiate communication, send requests, and wait for a response from B. B will never initiate communication.

Communication between A and B is intended to be bidirectional and asynchronous, using a single ØMQ socket in each component. Which sockets could A and B use?

a ROUTER in A and ROUTER in B.

b DEALER in A and ROUTER in B.

c DEALER at A and DEALER at B.

d All other options are valid.

19 *Consider an image 'a' in which there is no Node.js support and this Dockerfile:*

```
FROM a
COPY myPrg.js /
CMD node /myPrg
```

By using the appropriate command, a docker image named new-a has been generated using that Dockerfile.

Select the true statement among the following:

a When generating the new-a image, the node interpreter was installed in it.

b So that there are no errors in that generation of the new-a image, there must be a file myProg.js in the same directory where the Dockerfile is.

c If there were no errors generating the new-a image, executing the docker run new-a command on that same host will certainly not cause any errors.

d All other statements are true.

20 *Consider this Dockerfile:*

```
FROM a
COPY myPrg.js /
CMD node /myPrg
```

To generate a new-a image with this Dockerfile you should use this command in its same directory:

a docker run Dockerfile

b docker commit new-a

c docker build -t new-a .

d All other statements are true.

21 *There are two alternatives to create Docker images: create them interactively in a container that we will later save as an image, or generate them using a Dockerfile.*

Why is it often preferred to use Dockerfiles?

a Because they allow the use of environment variables whose value can be provided when starting the containers, thus simplifying dependency resolution.

b There are a larger number of images to be based on the Dockerfile than generating the image interactively.

c Generated images take up less space and require fewer resources than interactively created ones.

d Interactively generated images could not be used as a base image to generate other images via Dockerfiles.

22 *Why, among other reasons, is the deployment of a distributed application easier if containers are used than if the different programs that make up the application are installed and configured manually?*

a Because the dependencies related to the necessary libraries and configuration of each program have been largely resolved when creating their images.

b Because containers automate the intercommunication of components, no matter which deployment plan is used.

c Because containers, due to their more precise configuration, will always provide greater security guarantees.

d All other statements are false.

23 *Consider this file docker-compose.yml:*

```
version: '2'
services:
  one:
    image: zzz
    links:
     - two
    environment:
     - SERVER_IP=two
     - SERVER_PORT=80
  two:
    image: yyy
    expose:
- "80"
```

Select the true statement, about the functionality of the 'docker-compose up' command, in case it is used wherever that docker-compose.yml file is:

a Among other things, checks image 'yyy' for a bind() or listen() on port 80. Only start instances of 'two' if that is true.

b Among other things, checks the image 'zzz' for a connect() on the SERVER_PORT of a machine with IP address SERVER_IP. It only starts instances of 'one' if that is true.

c All other statements are true.

d Among other things, assign the IP address of service instance 'two' to the SERVER_IP variable of each service instance 'one'.

24 *Consider this file docker-compose.yml:*

```
version: '2'
services:
  one:
    image: zzz
    links:
- two
    environment:
     - SERVER_IP=two
     - SERVER_PORT=8080
  two:
    image: yyy
    expose:
     - "8080"
     - "8443"
```

Select the true statement if there is no 'zzz' image, but there is a 'yyy' image, on the host computer when using the 'docker-compose up' command where the file resides:

a A Dockerfile will be searched in the zzz subdirectory and with it the zzz image will be generated to start an instance of one.

b The image zzz will be checked for not existing locally and will be downloaded from the global repository, if it exists, to start an instance of one.

c It will check that the image zzz does not exist locally, but the links: clause indicates that if so we can use the image yyy instead.

d None of the other statements is true.

25 *Select the correct statement about consistency models:*

a The FIFO model is stronger than the cache model.

b The cache model is stronger than the FIFO model.

c The causal model is stronger than the sequential model.

d None of the other statements are true.

26 *Active replication is rarely used in relational databases because:*

a Each query usually requires a long processing interval and it is common to perform many more queries than modifications.

b UPDATEs tend to be more frequent than SELECTs and can change a large amount of state.

c There is no guarantee that the interconnection network of the replicas will be extremely fast and a fast network is a must in the active model.

d None of the other statements is true, since all relational databases use active replication, as we already saw when analyzing Wikipedia.

27 *Which statement is true for a scalable distributed service?*

a The service will be able to tolerate network partitioning situations, maintain its availability and respect relaxed consistency simultaneously.

b The service will not be able to respect sequential consistency.

c The service simultaneously tolerates network partitions, provides availability, and respects strict consistency among all subgroups that may be formed.

d For the service to respect FIFO consistency and to be available, partitions in the network must be avoided.

28 *In modern cloud systems, service elasticity is managed by level:*

a SaaS.

b PaaS.

c IaaS.

d Every level.

29 *The following 'docker-compose.yml' snippet was used in the first session of Lab 3 to configure the 'bro' and 'wor' components:*

```
version: '2'
services:
  # Clients are not relevant here.
  wor:
    image: worker
    build: ./worker/
    links:
     - bro
    environment:
     - BROKER_HOST=bro
     - BROKER_PORT=9999
  bro:
    image: broker
    build: ./broker/
    expose:
     - "9998"
     - "9999"
```

If the 'worker' and 'broker' images used in that session were not modified,

what changes should be made to the above snippet so that the 'wor' component starts before the 'bro' component and both components continue to work correctly?

a Remove the 'links' section from 'wor', and add a 'links' section with the value '- wor' in 'bro'.

b None. The requested startup order cannot be implemented, as the worker code depends on a previous startup of the broker.

c Move the 'links' and 'environment' sections from 'wor' to 'bro', replacing any 'bro' values in them with 'wor'.

d There is no need to make any changes, since the start order configured in that fragment already matches the one requested.

30 *The following 'docker-compose.yml' snippet was used in the first session of Lab 3 to configure the 'wor' component:*

```
version: '2'
services:
  # Clients and broker are not relevant here.
  wor:
    image: worker
    build: ./worker/
    links:
     - bro
    environment:
     - BROKER_HOST=bro
     - BROKER_PORT=9999
```

How does the worker.js program, used to create the 'worker' image, get the value of the BROKER_HOST variable?

a The 'environment:' section of a 'docker-compose.yml' is for informational purposes only and has no practical effect. The program does not use that variable at all.

b The program has a variable called BROKER_HOST and Docker will assign the value associated with 'bro' to it when the container starts.

c The BROKER_HOST variable was used in the Dockerfile associated with the 'worker' image and allows worker.js to get its value in one of its arguments.

d None of the other alternatives is true.

31 *The following 'docker-compose.yml' snippet was used in the second session of Lab 3 to partially configure the 'bro' component:*

```
version: '2'
services:
# Clients(cli), workers(wor) logger(log) not relevant here
  bro:
    image: broker
    build: ./broker/

    links:
     - log
    expose:
     - "9998"
     - "9999"
    ports:
     - "9998:9998"
```

What is the 'ports' section used for in that snippet?

a To tell the system administrator that port 9998 is the most important in this deployment. It is an informational section, just like expose.

b So that the 'log' component, when deployed on other machines, can interact with the 'bro' component.

c So that workers deployed on other machines can interact with the 'bro' component.

d To make the port 9998 of the 'bro' container correspond to the port 9998 of the host computer.

32 *Why does the WordPress deployment exercise, based on Bitnami, show no Dockerfile?*

a Because WordPress is pre-installed in the LINUX distribution of the portal virtual machines.

b Because having two images cannot build a Dockerfile that builds them at the same time.

c Those Dockerfiles are not downloaded because the docker-compose.yml file causes them to be executed on the remote repository.

d The Dockerfiles are not needed because the images have already been built and docker-compose.yml is based on them

NIST Retake 2022-002-03 **A**

Fill and deliver this answer sheet. Each question has a single correct answer. Don't forget to correctly write your personal data.

Don't fix a wrong answer with another mark: erase it or cover it with Tipp-Ex
A question with more than an answer will be discarded

0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9

DNI/NIE/passport: _____

Surname: _____

First Name: _____

PARTIAL 1

1	A	B	C	D
2	A	B	C	D
3	A	B	C	D
4	A	B	C	D
5	A	B	C	D
6	A	B	C	D
7	A	B	C	D
8	A	B	C	D
9	A	B	C	D
10	A	B	C	D
11	A	B	C	D
12	A	B	C	D
13	A	B	C	D
14	A	B	C	D
15	A	B	C	D
16	A	B	C	D

PARTIAL 2

17	A	B	C	D
18	A	B	C	D
19	A	B	C	D
20	A	B	C	D
21	A	B	C	D
22	A	B	C	D
23	A	B	C	D
24	A	B	C	D
25	A	B	C	D
26	A	B	C	D
27	A	B	C	D
28	A	B	C	D
29	A	B	C	D
30	A	B	C	D
31	A	B	C	D
32	A	B	C	D