1.what is clean coding program?

A)int functions(String str){

If str!=null{

Return str.length();

}

}

B) int functions(String str){

If (str==””){

Return 0;

}

}

c)int functions(String str){

Return str.length();

}

d)All

ANS: a)

2.node – v8- option|grep?

a)error,

b)error in unix command

c)v8 features in progress

d)all the versions features with command

ANS:b)

3)Exit code in node js

In NodeJS, syntax to exit is this.

process.exit([code])

Ends the process with the specified code. If omitted, exit uses the 'success' code 0.

To exit with a 'failure' code:

process.exit(1);

The shell that executed node should see the exit code as 1.

a)code with exceptional handler

b)error at runtime without exception handler

c)unused code

d)error

ANS: d) Only Not sure about the options

4).Which of the following is schedulers method?

a)Archaius (This is Netflix Configuration management – Netflix version of COnfig Server)

b)Nebula

c)Fenzo

d)—

ANS: c) Should be Fenzo. However note it is not a function. It is an open source **scheduler from Netflix** for Apache Mesos frameworks.

5.How to change the default settings of junit test?

a)Cant able to override

b)@testpropertysource

c)@runwith

d)custom annotation in bean

ANS: Not sure what this question means

6.On creating new project in angular which files will be created?

a)some file

b)environment.prod.ts

c)src/app/app.component.css

d)src/app/app.component.spec.ts

ANS: c and d only

7.How to connect data access layer (ORM) and core?

a)ORM

b)OXM

c)JMS

d)All the above

ANS:

8.Which of the following mutator(SETTER) method is correct?

a)function(){

ISResigned

--

}

b)function(){

Resigned

}

c)function(){

ResignedInt

}

d)All the above

ANS:

Setter method convention is setXXX().

Please note that for boolean it is isXXX(). For example:

public void isLegalAge(Boolean val){

this.legalAge=val;  
}

9)which autowire mode will be suitable for this code

@autowire

Class employee {

Methodname{@qualifier employee emp }{

this.employee=emp;

}}

a)ById

b)ByName

c)Constructor

d)Value

ANS: Not clear the question and options. Suggest you study all the types

10).Common things between json and xml?

* ANS: Both are data exchange format.
* Both supports unicode. So internationalization is supported by JSON and XML both.
* Both represents self describing data.
* Both are interoperable or language-independent.

11).Which of the json format is correct:

ANS:a)

{“customers”:[

{“job”:”xxx”,

“Name”:”ben’s shell”

},

{“job”:”xxx”,

“Name”:”ben’s Martein”

}

]

}

12).Which bean post processor method is called before initialization method of bean instance invoked ?

a)before()

b)after()

c)postprocessbeforeInitialization()

d)postbeforeinizialation

ANS: c)

13).Which of the following is true about dependency injection?

a).DI decoupling of object

b).DI (….)

c).DI (…..)

d).DI not useful in testing

A) A and D only

B) A,B,C Only

C)….

d)….

Not sure. A is 100% correct (DI decoupling of object)

D is incorrect (DI not useful in testing). In fact DI makes testing much easier

ANS:

14).Why we are using java configuration?

a).Easy to get dependency at runtime

b).easy to inject dependency with annotation

c)….

d)….

ANS: I assume this is related to Spring but not 100% sure from question. @Configuration **pring Configuration** annotation Indicates that a class declares one or more [@Bean](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/context/annotation/Bean.html) methods and may be processed by the Spring container to generate bean definitions and service requests for those beans at runtime, for example:

@Configuration

public class AppConfig {

@Bean

public MyBean myBean() {

// instantiate, configure and return bean ...

}

}

**Bootstrapping @Configuration classes**

***Via AnnotationConfigApplicationContext***

@Configuration classes are typically bootstrapped using either [AnnotationConfigApplicationContext](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/context/annotation/AnnotationConfigApplicationContext.html" \o "class in org.springframework.context.annotation) or its web-capable variant, [AnnotationConfigWebApplicationContext](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/web/context/support/AnnotationConfigWebApplicationContext.html" \o "class in org.springframework.web.context.support). A simple example with the former follows:

AnnotationConfigApplicationContext ctx = new AnnotationConfigApplicationContext();

ctx.register(AppConfig.class);

ctx.refresh();

MyBean myBean = ctx.getBean(MyBean.class);

// use myBean ...

See the [AnnotationConfigApplicationContext](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/context/annotation/AnnotationConfigApplicationContext.html" \o "class in org.springframework.context.annotation) javadocs for further details, and see [AnnotationConfigWebApplicationContext](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/web/context/support/AnnotationConfigWebApplicationContext.html" \o "class in org.springframework.web.context.support) for web configuration instructions in a Servletcontainer.

***Via Spring <beans> XML***

As an alternative to registering @Configuration classes directly against an AnnotationConfigApplicationContext, @Configuration classes may be declared as normal <bean>definitions within Spring XML files:

<beans>

<context:annotation-config/>

<bean class="com.acme.AppConfig"/>

</beans>

In the example above, <context:annotation-config/> is required in order to enable [ConfigurationClassPostProcessor](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/context/annotation/ConfigurationClassPostProcessor.html" \o "class in org.springframework.context.annotation) and other annotation-related post processors that facilitate handling @Configuration classes.

***Via component scanning***

@Configuration is meta-annotated with [@Component](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/stereotype/Component.html), therefore @Configuration classes are candidates for component scanning (typically using Spring XML's <context:component-scan/> element) and therefore may also take advantage of [@Autowired](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/beans/factory/annotation/Autowired.html)/[@Inject](https://docs.oracle.com/javaee/7/api/javax/inject/Inject.html?is-external=true) like any regular @Component. In particular, if a single constructor is present autowiring semantics will be applied transparently for that constructor:

@Configuration

public class AppConfig {

private final SomeBean someBean;

public AppConfig(SomeBean someBean) {

this.someBean = someBean;

}

// @Bean definition using "SomeBean"

}

@Configuration classes may not only be bootstrapped using component scanning, but may also themselves *configure* component scanning using the [@ComponentScan](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/context/annotation/ComponentScan.html) annotation:

@Configuration

@ComponentScan("com.acme.app.services")

public class AppConfig {

// various @Bean definitions ...

}

See the [@ComponentScan](https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/context/annotation/ComponentScan.html) javadocs for details.

15.By using which of the following method you will bind the event or handle event to attach

a)eventbind

b)attach

c)eventattach

d)event

ANS: Assuming this is on JQUery

The .bind() method is used for attaching an event handler directly to elements.

A basic usage of .bind() is:

|  |  |
| --- | --- |
| 1  2  3 | $( "#foo" ).bind( "click", **function**() {  alert( "User clicked on 'foo.'" );  }); |

**As of jQuery 3.0, .bind() has been deprecated.** It was superseded by the [.on()](https://api.jquery.com/on/) method

$( "#dataTable tbody tr" ).on( "click", **function**() {

console.log( $( **this** ).text() );

});

16.Annotation used one an only for attributes of setter method?

a)@required

b)@scope

c)…

d)….

ANS: @Autowired

17.Which of the following is useful for catch storage?

a)time and date ….

b)age get the time when instance created

c)………………………….

d)catche control

ANS: Obviously we will not cache time and date as it keep changing. Here I’m not clear on the other options. Use common sense. We cache things that doesn’t changes.

18.Netflix gateway => ANS:zuul

Netflix Discovery Service=>Eureka

Netflix Scheduler->Fenzo

Netflix COnfig Server🡺archaius

19.ANS:ApplicationContext Extends BeanFactory

20.ANS:<context:property-placeholder>tag tag is used to externalize **properties** in a separate file.

21. Image can be embedded to \_\_\_\_\_\_\_\_\_\_\_\_ tag?

a)<dd>

b)<dl>

c)<dt>

d)<df>

ANS: There is no <df> tag

22.a->a.toLowerCase();

String::toLowerCase();

State whether both are equal or not?

ANS:Equal

Not sure …. What tech this syntax is on. Why answer is true?

23).AutoDiscovery

Not sure what this means….

24).Jumbotron in bootstrap?

ANS:<div class=”jumbotron”>

25){{expression}} -> ANS:OneWay binding

[(ngModel)] -> ANS:TwoWay Binding

26)pager styling =>.previous & .next

27)set background (color,width) -> ANS:background-size

28)Transaction in xml?

ANS:<tx:annotation-driven>

29)Standard interface in java ?=>@serializable

30)OPTIONS:

a)<maven:implementation>install

b)<maven:implementation>copy

c) <maven:implementation>deploy

ANS:a) only

Not sure what these options mean. Out of these install and deploy are Maven specific. They are Maven default lifecycle phases.

31)Spring mvc controller= > how to move from ……..

a)archive

b)copy

c)play

d)………..

ANS: move WHAT?? Abmbiguous question. Try to recall the question

32)In spring virtual if some part is non-invasive what will be the progress,whther it will be working or not?

A)true

B)False

ANS: Couldn’t understand question. Thjere is nothing like spring virtual

33)Key terminals in aop?

A)covers

b)float

c)advice

d)aspect ^/

ANS: C and D

34)The tag to specify illustration

a)<mark>

b)<figure>

c)<aside>

d)<details>

ANS:b)

35)Angular cli command?

a)cli

b)ng

c)…

d)….

ANS:

Angular CLI commands start with ng.

A quick reference of Angular CLI command is here

<https://www.javatpoint.com/angular-7-all-cli-commands>

36)Code formatting scenario questions

37)Define bean for class of spring IOC Container

In XML use <bean id=”” class=””> Tag

In Annotation

@Configuration

public class BeanCreator{

@Bean

public MyBean getBean(){

return new MyBean();

}

}

38)Lambda expression ->True or false

39)a)<bean id=”” scope=”singleton”>

b) )<bean id=”” scope=”Request”>

c) )<bean id=”” scope=”prototype”>

d) )<bean id=”” scope=”Prototype” request=”true”> This is incorrect

40)Java code question

a)camelcasing

b)Hungarian

c)…

d)….

ANS:CAmelcasing