1. In a Spring xml configuration file, how would you define a bean definition for the class EmployeeServiceImpl where it has multiple alias's i.e. names

1 / 1 point

- ○ <bean id="myID" class="EmployeeServiceImpl"/>
- ○ <bean id="myID another, andAnother" class="EmployeeServiceImpl"/>
- ○ <bean class="EmployeeServiceImpl"/>
- ◉ <bean id="myID" name="another, andAnother" class="EmployeeServiceImpl"/>

Correct
Correct; alias's include a single id and a space and or comma delineated list of names

2. You have an EmployeeService bean that has a three argument a constructor of; public EmployeeService(String s, Integer i, Boolean b). Using XML how would you configure this bean?

1 / 1 point

- ☑ <bean id="emplService class="EmployeeServiceImpl" >
  <constructor-arg name="b' value="true"/>
  <constructor-arg value="hello"/>
  <constructor-arg type="int" value="3" />
  </bean>

Correct
Correct; resolves by name where true is injected into b, "3" into I by matching type and what's left into s

- ☑ <bean id="emplService class="EmployeeServiceImpl" >
  <constructor-arg index="2" value="true"/>
  <constructor-arg index="0" value="hello"/>
  <constructor-arg value="3" />
  </bean>

Correct
Correct; resolves by index true is injected into b, hello into s and what's left i.e. "3" into i

- ☑ <bean id="emplService class="EmployeeServiceImpl" >
  <constructor-arg name="b' value="true"/>
  <constructor-arg index="0" value="hello"/>
  <constructor-arg value="3" />
  </bean>

Correct
Correct; resolves by name where true is injected into b, hello into s and what's left i.e. "3" into i

- ☑ All the above

Correct
Correct; All of the above

3. What are key advantages of using the Spring framework? **1 / 1 point**

- (●) Promotes decoupling and reusability
- ( ) Code is coupled to Spring
- ( ) All of the above
- ( ) Object dependencies are clearly define din the application code itself

Correct
Correct; using Inversion of Control, dependencies are declared in configurations not code

4. What are we defining here? **1 / 1 point**

A principal where the control of a program, i.e. delegations from one object to another, are inverted: instead of the programmer controlling the flow in code, external sources (framework, services, configuration resources) take control of it.

- ( ) Code to the interface
- (●) Inversion of control
- ( ) Inheritance
- ( ) Dependency injection

Correct
Correct, configurations for example in Spring identify the rules of engagement when it comes to delegation patterns i.e. which object delegates to which object

5. What are we defining here? **1 / 1 point**

A design pattern where class X is dependent on Y. So rather than creating object of Y within the class "X", we can inject the dependencies via a constructor or setter injection.

- (●) Dependency injection
- ( ) Inheritance
- ( ) Inversion of control
- ( ) Code to the interface

Correct
Correct; this is a design pattern where the actual injection of a dependency into an object is controlled by Inversion of Control

6. What types of dependency injection are there in XML configurations **1 / 1 point**

- [ ] Method injection
- [x] Setter

Correct
Correct; Setter injection uses the <Property> tag

- [x] Constructor

Correct
Correct; Constructor injection involves complex XML resolution techniques on <constructor-arg> tags

- [ ] Field

7. ApplicationContext implementations have what type of built in class type to read metadata — 1 / 1 point

- ○ InputStreamReader
- ○ FileReader
- ◉ Resource
- ○ JSON Parser

Correct
Correct, built in Resource implementations can read and parse XML for the ApplicationContext to create BeanDefinition objects

8. In an XML configuration file, what bean tag attribute do you use to you ensure that the bean definition creates different managed bean instances per request — 1 / 1 point

- ○ scope=singleton
- ○ scope=request
- ◉ scope=prototype
- ○ it's the default there is no need to add an attribute

Correct
Correct; prototype bean use the Prototype pattern to lazy instantiate bean instances on each request

9. Constructor argument Resolution in an XML file can be achieved by, pick all that apply — 1 / 1 point

- ☑ Order within type

Correct
Correct in XML everything is a string or a bean ref type, Order by default aligns with constructor args within type in the XML file

- ☑ Constructor name

Correct
Correct, e.g. type="int"

- ☑ Type

Correct
Correct, e.g. type="int"

- ☑ Index

Correct
Correct e.g. index="0

10. XML configuration file tags and grammar is controlled by; pick all that apply — 1 / 1 point

- ☐ Java jars
- ☑ Namespaces

Correct
Correct; tags belong to XML namespaces

- ☑ Schemas

Correct
Correct; the grammar or what attributes belong in which tag, or if a tag can be nested inside another tag are controlled by schemas for each namespace

- ☐ Java code