☐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

Langages et Environements Évolués
Examen 2017

Nom, prénom :

.....

JEE

Question 1 ♣ La journalisation est

- ■ le log
- ■ un outil de débug
- ■ Un aspect
- ☐ toujours affiché sur le terminal
- ☐ un cron quotidien

Question 2 ♣ Lequels sont des serveurs d'application:

- ■ Glassfish
- ■ JBoss
- ☐ JEE
- ☐ Tomcat
- ☐ Derbby

Question 3 ♣ Les sigles suivant sont des design paterns:

- ☐ JPQL
- ■ Factory
- ☐ ORM
- ■ MVC
- ☐ Log4J

Question 4 ♣ En Spring, ces annotation font de l'injection de dépendance:

- ☐ @Table
- ■ @PersistenceContext
- ☐ @Scope
- ■ @Autowired
- ■ @RequestMapping

Question 5 ♣ Que veut dire JSP

- ☐ JavaScript Page
- ☐ Java Service Presentation
- ☐ Java Servlet POJO
- ☐ Java Server Persistance
- ■ Java Server Pages

Question 6 ♣ *Remarque: La question avait une erreur, la répons "Aucun" était donc considérée comme correcte.*
La requète JPQL : "SELECT a FROM Pizza a WHERE a.prix < :prix" rend un objet de type

- ■ Collection<Pizza>
- ☐ PizzaDAO
- ☐ String
- ☐ Collection<String>
- ☐ Aucun (message d'erreur)

Question 7 ♣ *Remarque : La réponse "je croise les doigts" était une blague, elle n'enlevait ne ne donnait aucun point.*
Sur IntelliJ, avant de de monter une archive JEE :

- ■ Je lance les tests unitaires
- ■ j'étein ma derby embeded
- ■ Je croise les doigts
- ☐ Je kill mon browser
- ☐ Je redémare Glassfish

Question 8 ♣ Il faut utiliser des *setters* et *getters* plutôt que des paramètres pubilcs dans une classe car ceux-ci sont utilisés par:

- ☐ JDBC
- ■ Spring IoD
- ■ JavaEL
- ☐ EJB
- ■ JPQL

Question 9 ♣ Pour faire du front-end, j'utilise

- ☐ JPA
- ■ JSP
- ■ Une Servlet
- ■ Angular
- ☐ Une entité

Question 10 ♣ Lesqels sont des schemas d'architecture

- ■ MVC
- ☐ EJB
- ■ AOP
- ■ MVVM
- ■ 5-tier

Question 11 ♣ Une immage Docker

- [ ] est chargé à chaque utilisation
- [ ] ne voit pas l'exterieur
- [x] contien le binaire de votre programme
- [x] peut être chargé sur plusieurs machines
- [x] contien le binaire d'un système d'exploitation

Question 12 ♣ Lesquels sont des niveaux de log:

- [x] Info
- [ ] Exception
- [x] Fatal
- [ ] Oups
- [ ] Note

Question 13 ♣ Pour interagir avec une base de donnée relationelle, j'utilise

- [ ] Git
- [ ] Derbby
- [x] JDBC
- [x] un ORM
- [ ] JEE

Question 14 ♣ Un serveur git

- [ ] est un SVN
- [x] dispose d'une copie local
- [x] est un gestionaire de version
- [ ] contien vos fichiers de codes
- [ ] stoque vos binaires

Question 15 ♣ Docker permet de

- [ ] sécuriser facilement
- [x] faire de la virtualisation légère
- [x] déployer facilement
- [ ] faire de la virtualisation
- [x] paraméter finement votre serveur

Question 16 ♣ Lesquels de ces beans sont bien des technos java:

- [x] Java Bean
- [x] Spring Bean
- [ ] Hibernate Bean
- [x] Enterprise Java Bean
- [ ] Persistance Bean

Question 17 ♣ Un IDE permet

- [ ] de monter une base de donnée
- [ ] de compiler du code
- [x] de générer du code
- [x] d'indiquer vos fautes
- [x] d'utiliser d'autres programes

Question 18 ♣ Un contener Docker

- [ ] contien le binaire de votre programme
- [x] peut être chargé sur plusieurs machines
- [x] est chargé à chaque utilisation
- [x] ne voit pas l'exterieur
- [ ] contien le binaire d'un système d'exploitation

Question 19 ♣ Que veut dire JPA

- [ ] Java Persistance Application
- [ ] Java Package Application
- [ ] Java Presentation App
- [x] Java Persistance API
- [ ] Java POJO AOP

Question 20 ♣ Une page JSP est compilé en

- [x] Une Servlet
- [ ] Aucun
- [ ] Un code javascript
- [ ] Un EJB stateless
- [ ] Un EJB statefull

Question 21 ♣ Votre projet java contien des fichiers

- [ ] .class
- [x] .jsp
- [ ] .form
- [x] .java
- [x] .xml

Question 22 ♣ Les tests unitaire:

- [ ] Se lancent sur le serveur
- [x] Testent chaque méthode
- [ ] Sont infaillbles
- [ ] Utilisent JSP
- [x] Se font avec JUnit

Question 23 ♣ Un ORM

- ☐ fait une copie local de votre BdD
- ☐ utilise une BdD noSQL
- ■ dispose d'un langage de requète
- ■ gère la vie des entités
- ■ cré une BdD virtuelle objet

Question 24 ♣ Parmit les modules de Spring, on trouve:

- ☐ Maven
- ■ Boot
- ■ AOP
- ■ MVC
- ☐ Hibernate

Question 25 ♣ Dans IntelliJ, la génération de mapping de persistance à l'aide d'un schéma de BdD permet

- ■ de faire des liens many-to-many
- ■ de générer des entités
- ☐ de créer des classes embeded
- ☐ de faire de l'héritage
- ☐ de faire des liens many-to-two

Question 26 ♣ Pour configurer un mapping ORM, je peux utiliser les annotations:

- ☐ @Override
- ☐ @PersistenceUnit
- ■ @Table
- ■ @ManyToMany
- ■ @Embeded

Exercice 2 *Décrivez le fonctionnement d'un ORM*

Exercice 3 *On veut écrire une appli qui distribue des pubs. Il doit y avoir deux page web, l'une de pub et l'autre de gestion client. On suppose que chaque nouvelle session sur la page de client est un nouveau client. Un client peut charger une page html, une immage, une phrase ou un lien; mais un seul (s'il recommence, le dernier est supprimé). La page client affiche (correctement) une des publicités chargés par les clients au hasard, ce à chaque rafraîchissement. Le client doit voire combien de fois sa publicité a été affichée.*

Décrire les technos que vous utiliseriez pour implémenter ça.

Décrire comment vous l'implémentez (vous pouvez écrire du code ou dire quels options de IntelliJ vous utilisez).

Exercice 4. Donnez une trace d'exécution du programme suivant :

Dans `src/Visiteur.java` :

```
@Component
@Configuration
@Scope("prototype")
@Primary
public class Visiteur {
  static private int compteur = 0;
  public int num = compteur++;

  public String annonce() {
    return String.format("vous s le visiteur n.%d", num);
}  }
```

Dans `src/subpkg/Client.java` :

```
@ComponentScan
@Component
@Configuration
public class Client extends Visiteur {
  @Autowired
  public Id id;

  public String annonce() {
    return String.format("Bienvenu %s, %s", id.toString(), super.annonce());
}  }
```

Dans `src/subpkg/Id.java` :

```
@Component
@Configuration
public class Id {
  public String nom;
  public String prenom;

  public Id () {
    Scanner scan = new Scanner(System.in);
    System.out.println("Nom:");
    nom = scan.nextLine();
    System.out.println("Prenom:");
    prenom = scan.nextLine();
}  }
```

Dans `src/Main.java` :

```
@ComponentScan
public class Main {
  public static void auxi (ApplicationContext context) {
    Visiteur v = context.getBean(Visiteur.class);
    System.out.println(v.annonce());
  }

  public static void main (String[] args) {
    ApplicationContext context = new AnnotationConfigApplicationContext(Main.class);
    ApplicationContext context2 = new FileSystemXmlApplicationContext(subpkg/Client.java);
    auxi(context);
    auxi(context2);
    auxi(context);
    auxi(context2);
} }
```