Nom et Prénom…… classe ……

Partie 1 : QCM (8 points)

Question	1	2	3	4	5	6	7	8
Réponse	D	D	A	B	C	D	D	C

Partie 2 : (12 points)

1. Compléter le code suivant de l'entité Parc sachant que l'attribut id est la clé primaire, entier et s'incrémente automatiquement, l'attribut nom est une chaine de caractère **[0.5 pt].**

[1]……GeneratedValue……[0.25 pt]
[2]…… Column…… [0.25 pt]

2. a/ Quelles sont les étapes à faire pour faire le mapping des entités vers les tables de la base de données ? **[1pt]**

……php bin/console make :migration…… ……[0.5 pt]

……php bin/consoledoctrine :migrations :migrate……[0.5 pt]

b/ Quel est le fichier qui sera généré après cette étape (de la question 2-a) ainsi que son emplacement ? **[0.5pt].**

……Le fichier de migration……[0.25 pt]

……/migrations ……[0.25 pt]

3. Ajouter le code nécessaire pour avoir une relation entre les deux entités**. [1pt]**

[1]……ManyToOne……[0.25 pt]

[2]…… inversedBy……[0.25 pt]

[3]…… OneToMany…… [0.25 pt]

[4]…… mappedBy ……[0.25 pt]

4. Compléter le code nécessaire pour avoir le formulaire correspondant à l'entité Arbe comme indique la figure suivante **[1.5pts]**

[1]…… EntityType……[0.25 pt]

[2]…… class……[0.25 pt]

[3]…… Parc……[0.25 pt]

[4]…… choice_label……..[0.25 pt]

[5]…… nom……[0.25 pt]

[6]…… SubmitType……[0.25 pt]

5. Compléter la fonction de mise à jour d'un arbre (updateArbre) dans le contrôleur. **[2pts]**

[1]…… Request $req……[0.25 pt]

[2]…… $numArbre……[0.25 pt]

[3]……Arbre::class……[0.25 pt]

[4]……find($numArbre)…… [0.25 pt]

[5]…… $req(la variable de type Request) ……[0.25 pt]

[6]…… …… $this->getDoctrine()->getManager()……[0.25 pt]

[7]…… …… flush() ……[0.25 pt]

[8]…… …… $form->createView() ……[0.25 pt]

6. Compléter le code nécessaire afin d'afficher le numéro, la date d'implantation et le parc des arbres. **[2pts]**

[1]…… $this->getDoctrine()->getRepository(Arbre::class)…… [0.25 pt]

[2]…… findAll()…….. [0.25 pt]

[3]…… $arbres…… [0.25 pt]

[4]…… {% for arbre in arbres %}……[0.25 pt]

[5]…… date …… [0.25 pt]

[6]…… c.parc.nom……[0.5 pt]

[7]…… {% endfor %}……[0.25 pt]

7. Ecrire la fonction « deleteTree() » qui permet de supprimer un arbre qui est à risque de mourir et de rediriger vers la lister des arbres sinon retourner le message 'Erreur' **[2 pts]**

```
/**
 * @Route("/arbre/delete/{numArbre}", name="deleteArbre")
 */
public function delete($numArbre,EntityManagerInterface $em,ArbreRepository $rep)
{
   $arbre = $rep->find($numArbre); [0.25 pt]
   if($arbre->getARisque())  [0.5 pt]
   {
```

```
            $em->remove($arbre); [0.25 pt]
            $em->flush();[0.25 pt]
            return $this->redirectToRoute('arbre'); [0.25 pt]
        }
        else
            return new Response('Erreur'); [0.5 pt]
    }
```

8. Réécrire en corrigeant les erreurs de la fonction DQL « ParcStartByA() » suivante qui permet d'afficher l'ensemble des Parc dont le nom commence par la lettre « «**A** » **[1.5pts]**

……1) `createQueryBuilder ->` createQuery…… ……[0.5 pt]

……2) FROM p -> FROM App\Entity\Parc p…… ……[0.5 pt]

……3) 'A'-> 'A%'…… ……[0.25 pt]

……4) `getScalarResult` () -> getResult()…… ……[0.25 pt]