TELECOM SudParis

Corrigé et Barème Exemple de questions

Notes :

- les seuls documents autorisés sont le polycopié distribué en cours et les notes personnelles sur ce polycopié ;
- le travail doit se faire individuellement. N'oubliez pas d'indiquer vos nom et prénom sur chaque feuille ;
- soyez concis et précis, et justifiez vos réponses par des commentaires appropriés si nécessaires ;
- soyez rigoureux dans la syntaxe UML ;
- soyez rigoureux dans la syntaxe JAVA ;
- **veillez à rendre une copie propre et lisible, avec une marge à gauche ;**
- le barème est donné à titre indicatif.

1 Cahier des charges

Tout au long des cours du module, nous avons illustré le développement logiciel avec les exemples de la « Médiathèque ». Dans cet énoncé, nous démarrons ce qui pourrait faire partie d'un deuxième sprint de développement du logiciel Médiathèque.

Nous rappelons ici certains éléments du cahier des charges initial et complétons pour indiquer ce qui est nouveau.

Le système de gestion de la médiathèque doit prévoir toute opération d'ajout et de suppression de clients et de documents. Les informations les concernant ne sont pas construites par le système (par exemple, la localisation des documents dans les locaux), mais supposées fournies lors de l'invocation des opérations des cas d'utilisation.

Comme pour l'inscription, les clients abonnés renouvellent leur abonnement auprès d'un employé de la médiathèque. Les clients abonnés règlent une cotisation annuelle et peuvent emprunter des documents sans payer.

Un client ne peut pas emprunter plus d'un certain nombre de documents fixé par sa catégorie : 10 pour la catégorie « abonné ». Dès que ce nombre maximal est atteint pour un client donné, tout nouveau prêt doit être impossible.

Tout client qui n'a pas restitué un document avant sa date limite de restitution ne peut plus faire de nouvel emprunt tant qu'il n'a pas régularisé sa situation, ceci même si le nombre maximal d'emprunts n'est pas atteint. Pour ce faire, à chaque demande d'emprunt, le système vérifie s'il est à jour dans ses restitutions. Si ce n'est pas le cas, l'emprunt n'est pas autorisé.

Nous ajoutons dans ce deuxième sprint qu'un client abonné ne peut pas emprunter un document lors de la dernière semaine avant la date de renouvellement de son abonnement.

L'ensemble des abonnements est parcouru chaque jour afin de repérer s'il existe des fiches clients pour lesquelles la date de renouvellement approche. Le quarante cinquième jour avant la date de renouvellement, le système notifie le client abonné. Averti, le client abonné peut venir à la médiathèque pour demander le renouvellement de son abonnement. Par ailleurs, le renouvellement de l'abonnement n'est possible qu'à partir du quarantième cinquième jour précédent la date anniversaire. Lors du renouvellement, la date de renouvellement est reculée d'un an.

Nous récapitulons ici les nouveautés du deuxième sprint :

— un client abonné ne peut pas emprunter un document la dernière semaine avant la date de renouvellement ;

— le renouvellement de l'abonnement d'un client abonné n'est possible qu'à partir du quarantième cinquième jour précédent la date de fin de l'abonnement ;

— quarante cinq jours avant la date de renouvellement, le client abonné est notifié de la nécessité de se réabonner.

NB : en guise de simplification pour l'exercice, on écrira « nbJoursEntre(date1, date2) » pour calculer le nombre de jours entre deux dates.

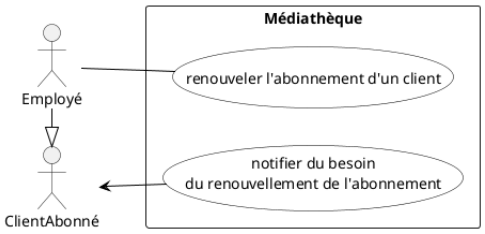
Dans la dernière page du sujet, nous rappelons le diagramme de classes de la Médiathèque, qui reste valable pour ce deuxième sprint.

1.1 Questions

Question 1 (2 points)

Construisez un diagramme de cas d'utilisation avec uniquement les nouveaux cas d'utilisation du deuxième sprint.

Réponse:



Barème de correction sur 2 points :

```
Répartition :
Max 2 points parmi
1 renouveler un abonnement avec bon lien de communication et bon acteur Employé
1 notifier des renouvellements avec bon lien de communication et bon acteur Client
```

Question 2 (2 points)

Établissez les précondition et postcondition du cas d'utilisation « renouveler un abonnement ».

Réponse:

```
Précondition :
   informations (nom, prénom) sur le client bien formées (non null et non vide)
ET client avec ces nom et prénom existant (inscrit)
ET client avec ces nom et prénom de catégorie « abonné »
ET date du jour >= date de renouvellement - 45

Postcondition :
   date de renouvellement = ancienne date de renouvellement + 1 an
```

Barème de correction sur 2 points :

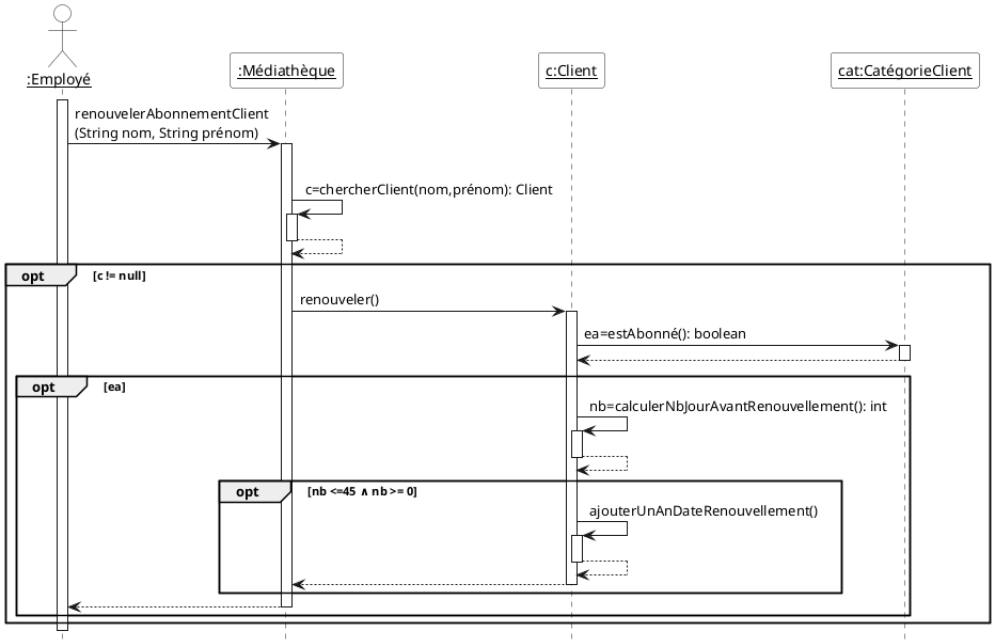
```
Répartition :
Max 2 points parmi :
0.5 deux termes sur les identifiants
0.5 terme sur existance client
0.5 terme sur client actuellement abonné
0.5 terme date pour renouvellement

malus 0.5 si non formulée en logique propositionnelle
```

Question 3 (3 points)

Construisez le diagramme de séquence du cas d'utilisation « renouveler un abonnement ».

Réponse:



Barème de correction sur 3 points :

```
Répartition :
Max 3 points parmi :
0.5 pas d'erreur UML (participants, messages, création participant, barres d'activation)
0.5 présence des objets nécessaires au fonctionnement de la séquence
1.0 précondition complète
1.0 effets complets (avec passage de références par arguments)
```

Question 4 (4 points)

Écrivez en JAVA toutes les méthodes mettant en œuvre le cas d'utilisation « renouveler un abonnement », c'est-à-dire toutes les méthodes identifiées avec le diagramme de séquence. Pour chaque méthode que vous écrivez, indiquez dans quelle classe elle se trouve. Par ailleurs, utilisez la programmation défensive ainsi que la programmation par assertion.

Réponse:

```
Classe cf2_2024_2025.Mediatheque
public void renouvelerAbonnementClient(final String nom, final String prenom) throws OperationImpossible
  Optional<Client> c = this.chercherClient(nom, prenom);
  if (c.isEmpty()) {
    throw new OperationImpossible("client " + nom + " " + prenom + " inexistant");
  }
  c.get().renouveler();
}

Classe cf2_2024_2025.client.Client
public void renouveler() throws OperationImpossible {
  if (!catClient.estAbonne()) {
    throw new OperationImpossible("pas la bonne catégorie");
  }
  // cf. énoncé : simplification opération sur dates
  long nbJours = Duration.between(Datutil.aujourdhui().atStartOfDay(),
                                  dateRenouvellement.atStartOfDay())
                         .toDays();
  if (nbJours > 45 || nbJours < 0) {
```

```
    throw new OperationImpossible("hors période de renouvellement");
  }
  // utilisant Datutil
  dateRenouvellement = Datutil.ajouterJoursADate(dateRenouvellement, 365);
}
```

Barème de correction sur 4 points :

```
Répartition :
Max 4 parmi :
1    pas d'erreur dans la programmation orientée objet et bonne méthode dans bonne classe
1    vérification arguments en entrée bien formés et client existant
1    vérification client abonné et période de renouvellement
0.5  mise à jour dateRenouvellement
1    exceptions et assertions
```

Question 5 (4 points)

Écrivez en JAVA et en utilisant JUnit les tests de validation du cas d'utilisation « renouveler un abonnement ». Vous écrivez toute la classe de test, hormis les déclarations `package` et `import`.

Réponse:

```
class TestRenouvelerAbonnement {
  private Mediatheque m1;
  @BeforeEach
  public void setUp() throws OperationImpossible {
    m1 = new Mediatheque("MediathequeTest");
    m1.ajouterCatClient("Abonné", 5, 25, 2.0, 0, false);
    m1.ajouterCatClient("Tarif Normal", 2, 25, 1.0, 1.0, false);
    m1.inscrire("nom1", "prenom1", "adresse1", "Abonné");
    m1.inscrire("nom2", "prenom2", "adresse2", "Tarif Normal");
  }
  @AfterEach
  public void tearDown() {
    m1 = null;
  }
  @ParameterizedTest
  @NullAndEmptySource
  void testRenouvelerAbonnementTest1(String input) {
    Assertions.assertThrows(OperationImpossible.class, () ->
                            m1.renouvelerAbonnementClient(input, "prenom1"));
  }
  @ParameterizedTest
  @NullAndEmptySource
  void testRenouvelerAbonnementTest2(String input) {
    Assertions.assertThrows(OperationImpossible.class, () ->
                            m1.renouvelerAbonnementClient("nom1", input));
  }
  @Test
  void testRenouvelerAbonnementTest3() {
    Assertions.assertThrows(OperationImpossible.class, () ->
                            m1.renouvelerAbonnementClient("nom1", "prenom1"));
  }
  @Test
  void testRenouvelerAbonnementTest4Jeu1() {
    Assertions.assertThrows(OperationImpossible.class, () ->
                            m1.renouvelerAbonnementClient("nom1", "prenom1"));
  }
  @Test
  void testRenouvelerAbonnementTest4Jeu2() throws OperationImpossible {
    Datutil.retirerJoursALaDateDuTest(300);
```

```
    m1.inscrire("nom3", "prenom3", "adresse3", "Abonné");
    Datutil.resetDateDuTest();
    Assertions.assertThrows(OperationImpossible.class, () ->
                            m1.renouvelerAbonnementClient("nom3", "prenom3"));
  }
  @Test
  void testRenouvelerAbonnementTest5() throws OperationImpossible {
    Datutil.retirerJoursALaDateDuTest(350);
    m1.inscrire("nom3", "prenom3", "adresse3", "Abonné");
    Datutil.resetDateDuTest();
    Assertions.assertEquals(1, m1.listerAbonnesRenouvellementPossible().size());
    m1.renouvelerAbonnementClient("nom3", "prenom3");
  }
}
```

Barème de correction sur 4 points :

```
Répartition :
Max 4 parmi :
1    scénario dans le @BeforeEach
1    différents tests correspondant à la précondition
0.5  @BeforeEach + @AfterEach + @Test
0.5  assertThrows en cas de problème
0.5  assertEqual, etc. pour le test avec succès
0.5  tests paramétrés
```

Question 6 (2 points)

À l'aide de *Stream* JAVA, écrivez la méthode du cas d'utilisation « lister les renouvellements en cours », qui liste tous les clients abonnés dont la date de renouvellement est proche, c'est-à-dire qui peuvent demander le renouvellement de leur abonnement.

Réponse:

```
public List<String> listerAbonnesRenouvellementPossible() {
  // cf. énoncé : simplification opération sur dates
  return lesClients.entrySet().stream()
         .filter(e -> e.getValue().getCategorie().estAbonne())
         .filter(e -> Duration.between(Datutil.aujourdhui().atStartOfDay(),
                                       e.getValue().getDateRenouvellement()
                                .atStartOfDay()).toDays() <= 45)
         .map(e -> e.getKey().toString()).toList();
}
```

Barème de correction sur 2 points :

```
Répartition :
Max 2 parmi :
0.5 Stream partant de la façade et de la collection des clients
0.5 méthode de la façade ne retournant pas d'objet de l'intérieur
0.5 filtre sur le nombre de jours avant la date de renouvellement
0.5 map pour obtenir les noms et prénoms dans une chaîne de caractères
0.5 toList ou autre pour la collection résultat
```

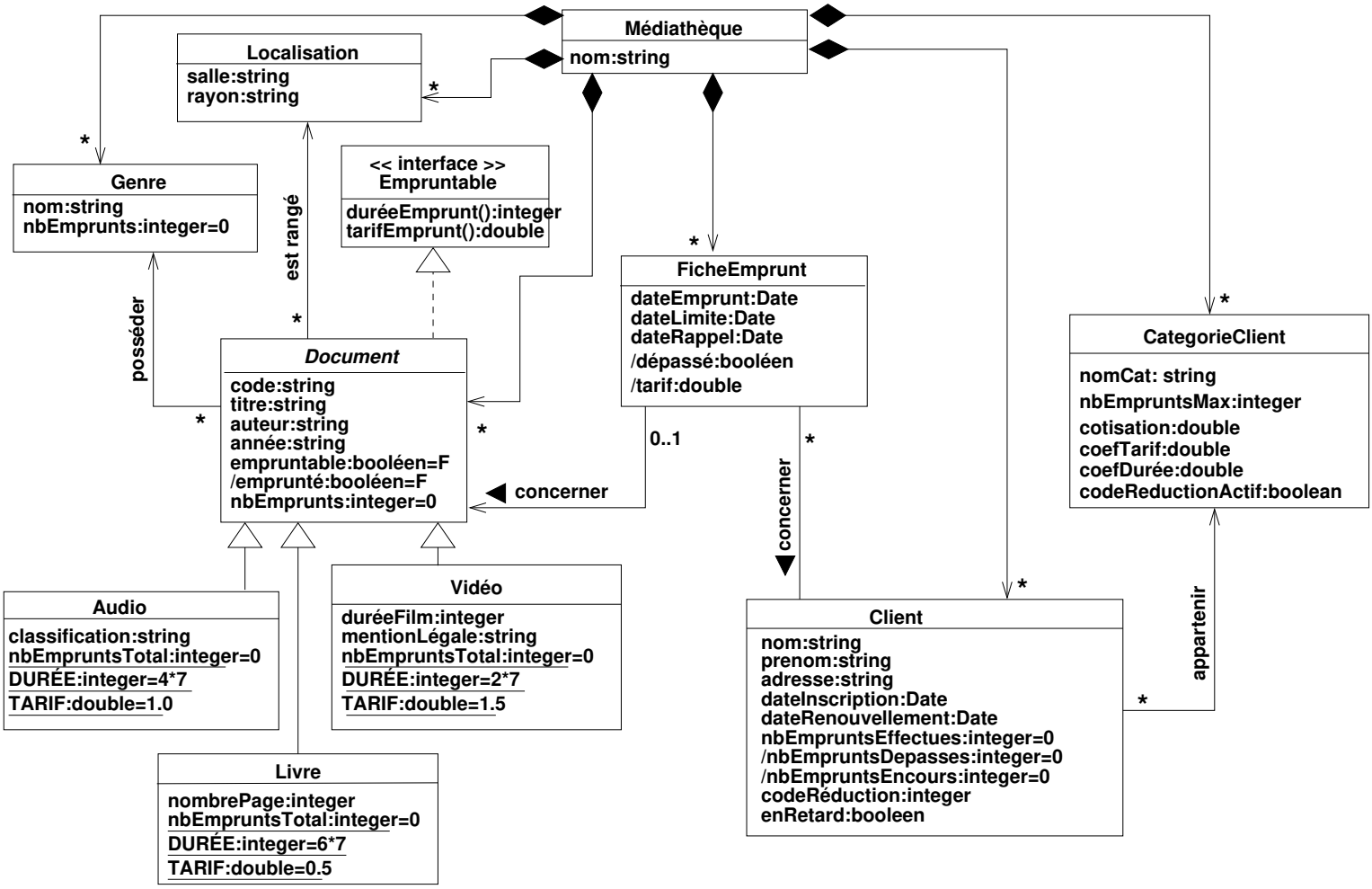
Question 7 (3 points)

Pour esquisser la mise en œuvre de la notification aux clients abonnés des renouvellements le quarante cinquième jour avant la date anniversaire, indiquez les éléments à ajouter. Pour chaque élément ajouté, précisez les informations sur l'élément avec le rôle de l'élément (à l'aide de quelques phrases et/ou de quelques lignes de code JAVA) : « telle classe avec tel contenu (attribut et méthode) pour tel rôle », « tel attribut de tel type dans telle classe pour tel rôle », « telle instruction dans telle méthode de telle classe pour tel rôle », etc.

Exemple de questions d'un sujet complet

Barème de correction sur 3 points :

```
Répartition :
Max 3 parmi :
1   le consommateur à créer
1   l'attribut de type SubmissionPublisher dans la classe Client
1   le subscribe à ajouter dans le cas d'utilisation « inscrire un client abonné»
1   le publish dans le cas d'utilisation « notifier des renouvellements »
```


Médiathèque
nom:string
Localisation
salle:string
rayon:string
Genre
nom:string
nbEmprunts:integer=0
<< interface >>
Empruntable
duréeEmprunt():integer
tarifEmprunt():double
est rangé
posséder
FicheEmprunt
dateEmprunt:Date
dateLimite:Date
dateRappel:Date
/dépassé:booléen
/tarif:double
CategorieClient
nomCat: string
nbEmpruntsMax:integer
cotisation:double
coefTarif:double
coefDurée:double
codeReductionActif:boolean
Document
code:string
titre:string
auteur:string
année:string
empruntable:booléen=F
/emprunté:booléen=F
nbEmprunts:integer=0
0..1
concerner
concerner
appartenir
Audio
classification:string
nbEmpruntsTotal:integer=0
DURÉE:integer=4*7
TARIF:double=1.0
Vidéo
duréeFilm:integer
mentionLégale:string
nbEmpruntsTotal:integer=0
DURÉE:integer=2*7
TARIF:double=1.5
Client
nom:string
prenom:string
adresse:string
dateInscription:Date
dateRenouvellement:Date
nbEmpruntsEffectues:integer=0
/nbEmpruntsDepasses:integer=0
/nbEmpruntsEncours:integer=0
codeRéduction:integer
enRetard:booleen
Livre
nombrePage:integer
nbEmpruntsTotal:integer=0
DURÉE:integer=6*7
TARIF:double=0.5