

**Examen de Mathématiques 2 :**

1ère année Bachelier en Informatique de Gestion

**BINV1100 – Mathématiques 2**

Date : 17 août 2022

Durée de l’examen : 3 heures

Nombre de questions : 6

1. **Sauf avis contraire, toute réponse doit être justifiée.**
2. Si vous n’écrivez pas proprement et lisiblement, votre réponse recevra un zéro.
3. Écrire au crayon est autorisé si le point 2 ci-dessus est respecté.
4. Vous pouvez avoir à votre disposition 10 feuilles recto/verso respectant les conditions suivantes : vos nom et prénom doivent être indiqués, les feuilles doivent être manuscrites, reliées sur toute la longueur de manière à ne pas pouvoir en détacher sans l'arracher et le contenu ne fait pas l’objet de miniaturisation.
5. Pour les questions sur machine, vous devez travailler **sur le U :** . En effet, si vous travaillez ailleurs vos fichiers seront perdus.
6. Les points communiqués en regard des questions sont indicatifs. Des lacunes graves entraîneront l’échec au présent examen.
7. **Mettez vos noms et prénoms au début de chaque question !**

|  |  |
| --- | --- |
| Question 1 | **/10** |
| Question 2 | **/10** |
| Question 3 | **/15** |
| Question 4 | **/20** |
| Question 5 | **/10** |
| Question 6 | **/15** |
| **TOTAL** | **/80** |

**PARTIE I : SUR PAPIER**

**Question 1 (10 pts)**

Sur l’ensemble , on considère la relation définie par

Une image contenant montre, horloge

Description générée automatiquementEt la relation dont voici le digraphe :

1. Donnez le digraphe de .
2. Donnez le digraphe de
3. La relation est-elle réflexive, antiréflexive, symétrique, antisymétrique, transitive ? **Justifiez pour chaque propriété !!!**
4. Si la relation est maintenant définie sur (l’ensemble des naturels), quelle(s) propriété(s) est (sont) conservée(s) ? **Justifiez.**

**Question 2 (10 pts)**

Une image contenant ciel, jour

Description générée automatiquementSoit un ordre partiel sur l’ensemble, défini par le diagramme de Hasse ci-dessous :

1. Donnez l’ensemble des maximaux de .
2. L’ordre est-il un treillis ? **Justifiez.**
3. Donnez l’ensemble des minimaux de .
4. Donnez l’ensemble des majorants de l’ensemble
5. Donnez l’ensemble des minorants de l’ensemble
6. Donnez, s’il existe, le supremum de l’ensemble
7. Soit l’ensemble ordonné par . Combien y a-t-il de tri topologique de  ? **Justifiez.**
8. Proposez un sous-ensemble de , pas totalement ordonné, qui contienne minimum 4 éléments et qui soit un treillis pour l’ordre partiel ≤. **Justifiez**

**Question 3 (15 pts)**

Un étudiant étudie un cours pour la seconde session. Ce cours comporte 3 chapitres qui nécessitent 2h de lecture chacun. Dès qu’il a fini un chapitre, il passe au suivant. Mais

* Après avoir lu le chapitre 1, il passe au chapitre 2 et ne passe jamais du chapitre 1 au chapitre 2.
* Après avoir lu le chapitre 2, il a une chance sur 4 d’avoir besoin d’une notion du chapitre 1 qu’il n’a pas bien assimilée donc de devoir retourner lire le chapitre 1 en entier. Sinon il passe au chapitre 3.
* Après avoir lu le chapitre 3, il a une chance sur 4 d’aller relire le chapitre 1 et une chance sur 4 d’aller relire chapitre 2 pour revoir des notions mal comprises. Sinon il considère qu’il a terminé l’étude de ce cours.
* Si l’étudiant considère qu’il a terminé l’étude du cours alors il arrête définitivement sa lecture.

On vous demande de modéliser cette situation sous la forme d’un processus de Markov, l’unité de temps étant le temps nécessaire à la lecture d’un chapitre.

1. Dessinez le graphe du processus de Markov associé.
2. Donnez la matrice de transition du processus
3. Donnez les classes de communication, la période de chaque classe et la nature de chaque état.
4. Le processus admet-il une distribution stable ? **Justifiez**. Si oui, Est-elle unique ? **Justifiez sans la calculer !**
5. On désire déterminer le temps moyen qu’il faudra à l’étudiant pour étudier l’ensemble du cours. Donnez le système, la matrice à échelonner ainsi que la variable à déterminer pour trouver ce temps de parcours. **Ne résolvez pas le système et n’échelonnez pas sa matrice !**

**Question 4 (20 pts)**

1. a) Soit L le langage, sur l’alphabet , formé de tous les mots de minimum 2 lettres

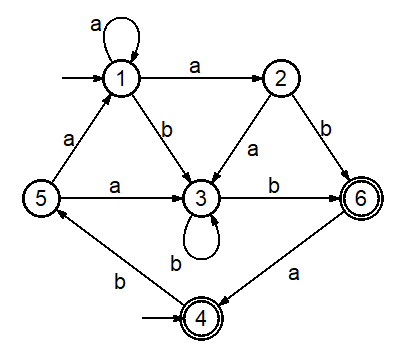
comprenant au maximum 1 b et dans lesquels un c est toujours suivi d’un a. Donnez une grammaire régulière engendrant ce langage. Précisez bien à quel « état » correspond chaque symbole non terminal !

|  |
| --- |
|  |

b) Soit L le langage, sur l’alphabet , formé de tous les mots où le nombre de a est divisible par 3 mais où il n’y a jamais deux a consécutifs. Donnez un automate de Moore reconnaissant ce langage. Précisez bien à quel mot correspond chaque état !

|  |
| --- |
|  |

1. Soit le langage défini sur reconnu par le NDFA .



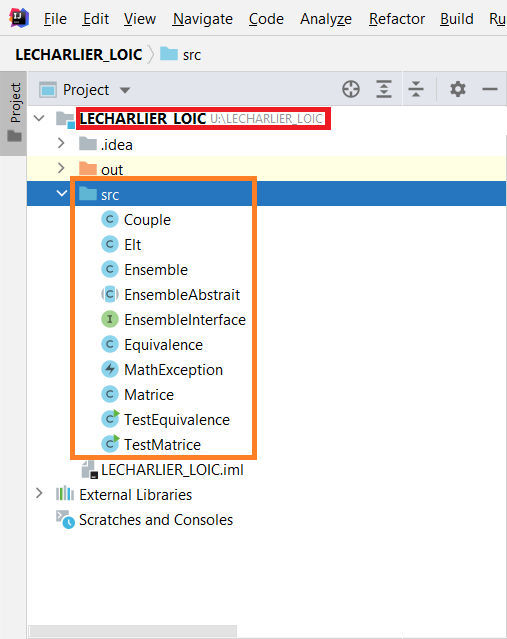
Utilisez la subset construction afin d’obtenir un automate de Moore reconnaissant le langage  :

**PARTIE II : IMPLÉMENTATION**

Dans cette partie nous allons vous demandez d’implémenter des méthodes dans plusieurs classes classes Java.

Pour ce faire :

1. Ouvrez IntelliJ
2. Créez, **sur le U :**, un projet NOM\_PRENOM (**avec vos nom et prénom** !)
3. Les classes données se trouvent dans le répertoire « Classes Java ». Faites un copier-coller de celles-ci dans le répertoire « src » de votre projet IntelliJ. Voici ce que vous devriez obtenir  :



Les questions 5 et 6 ci-après vous expliquerons ce que vous devez implémenter.

**Question 5 (10 pts)**

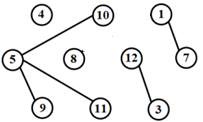
Les instances de la classe *Equivalence* sont des relations d’équivalence sur une partie de l’univers.

La partie de l’univers sur laquelle est définie la relation d’équivalence est stockée dans un *EnsembleAbstrait* (sousJac).

On a choisi de représenter une équivalence en choisissant pour chaque classe un représentant. Afin de réaliser cela, la classe Equivalence garde un tableau d’*Elt* (tabRep) dans lequel on stocke à l’indice correspondant à l’*Elt* le représentant de sa classe.

Exemple :

Soit ∼ une relation d’équivalence sur {1,3,4,5,7,8,9,10,11,12}.



Cette relation ∼ a 5 classes d’équivalence : {4}, {5,9,10,11}, {1,7}, {8} et {3,12}

On choisit un représentant par classe.

Par exemple : {**4**}, {5,**9**,10,11}, {1,**7**}, {**8**} et {**3**,12}

sousJac = {1,3,4,5,7,8,9,10,11,12}

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| \ | 7 | \ | 3 | 4 | 9 | \ | 7 | 8 | 9 | 9 | 9 | 3 | \ |  | \ |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | … | MAX |

tabRep =

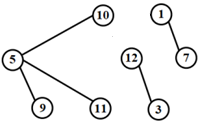
Définition : L’ensemble des isolés est l’ensemble des éléments du sous-jacent qui ne sont en relation avec aucun autre élément.

Dans l’exemple : il y a 2 éléments isolés : 4 et 8.

**Dans la classe *Equivalence*, complétez la méthode supprimerLesIsoles().**

Testez-la avec la classe *TestEquivalence*.

Après l’exécution de la méthode supprimerLesIsoles, la relation d’équivalence précédente devient



Avec

sousJac = {1,3,5,7,9,10,11,12}

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| \ | 7 | \ | 3 | \ | 9 | \ | 7 | \ | 9 | 9 | 9 | 3 | \ |  | \ |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | … | MAX |

tabRep =

Attention ! Cette classe ***Equivalence*** n’implémente pas *RelationAbstraite* ! Elle n’a donc ni de méthode contient() ni d’itérateur sur les couples de la relation.

Par contre, la classe *Ensemble* fournie implémente *Iterable*.

**Question 6 (15 pts)**

On vous demande d’implémenter la méthode

echelonner(int numLigne, int[] lignes, double[] coeff)

de la classe Matrice.

Cette méthode va remplacer la ligne numLigne, par la somme des lignes contenue dans le tableau lignes, chaque ligne étant pondérée par un coefficient se trouvant dans le tableau coeff.

*Exemple :*

Si on veut faire alors

* numLignes = **3** (on veut modifier la ligne 3)

|  |  |  |
| --- | --- | --- |
| **3** | **1** | **2** |

* lignes =

|  |  |  |
| --- | --- | --- |
| **1** | **-0.5** | **1.5** |

* coeff =

Mais on veut que la modification respecte les règles de l’échelonnage, il y a donc quelques conditions à vérifier :

* nSupumLigne est un numéro de ligne valide
* lignes et coeff ne sont pas null
* lignes et coeff doivent avoir la même taille
* lignes contient des numéros de lignes valide
* lignes ne contient pas deux fois le même numéro de ligne
* lignes doit contenir numLigne (on doit garder les infos de la ligne à modifier)
* Le coefficient contenu dans coeff correspondant à numLigne ne peut pas être égal à 0 (on doit garder les infos de la ligne à modifier)

Si une de ces conditions n’est pas vérifiées, il faut lancer une IllegalArgumentException.

*Exemple :* Supposons que this est une matrice

1. Si numLigne = 5 alors IllegalArgumentException car this n’a pas de ligne 5.

|  |  |  |
| --- | --- | --- |
| 3 | **1** | **1** |

1. Si numLigne = 3 et lignes = alors IllegalArgumentException car

lignes contient deux fois la ligne 1.

|  |  |  |
| --- | --- | --- |
| 3 | 1 | **-1** |

1. Si numLigne = 3 et lignes = alors IllegalArgumentException

car lignes contient -1 qui est un numéro de ligne qui n’existe pas.

|  |  |
| --- | --- |
| 1 | 2 |

1. Si numLigne = **3** et lignes = alors IllegalArgumentException

car lignes ne contient pas **3** qui est la ligne que l’on veut modifier.

|  |  |  |
| --- | --- | --- |
| **3** | 1 | 2 |

|  |  |  |
| --- | --- | --- |
| **0** | 0.5 | -1 |

1. Si numLigne = **3** , lignes = et coeff =

alors IllegalArgumentException car le coefficient, contenu dans coeff, correspondant à numLigne est égal à 0.

**BROUILLON**

**BROUILLON**

**BROUILLON**