HAUTE ÉCOLE DE
NAMUR-LIÈGE-LUXEMBOURG
Catégorie technique
Année académique : 2021-2022

TI208 Programmation orientée objet 2
Examen : Enoncé général et directives

Sauf exception, l'énoncé du travail est imposé.

Chaque groupe d'étudiants devra implémenter des fonctionnalités différentes ; chaque groupe recevra donc un énoncé spécifique différent.

La base de données sur laquelle vous travaillerez vous sera fournie.
Vous disposez sur Moodle d'un document contenant la description détaillée de la base de données ainsi que le script de création de la BD.

Précisions sur la cotation et le déroulement de l'examen

- La cotation attribuée pour l'examen final du cours de Programmation orientée objet 2 (théorie et laboratoire) est basée d'une part sur un *travail de groupe* et d'autre part sur un *examen oral*.

- Le travail consiste en la réalisation d'un programme écrit en java. Ce travail est à effectuer en groupe de 2 étudiants. La constitution des groupes doit être validée par le professeur de laboratoire.

- Une première note sera attribuée par le professeur après évaluation du travail. Cette note sera considérée comme point de départ pour l'évaluation orale ; elle sera augmentée ou diminuée en fonction des réponses données par l'étudiant aux questions posées par le professeur lors de l'examen oral.

- Lors de l'examen oral :

    - L'étudiant recevra un document contenant quelques questions théoriques et/ou pratiques concernant la programmation java. Il aura +/- 20' pour préparer les réponses qu'il défendra au moment de son examen oral, avant la défense de son application java. Des exemples de questions type seront disponibles sur Moodle.

    - L'étudiant devra être capable d'expliquer ***n'importe quelle ligne de code de l'application java réalisée en groupe***. Une attention particulière sera apportée à la compréhension des principes de la programmation orientée objet, des mécanismes liés à **l'héritage**, à la gestion des **exceptions**, à la gestion des **évènements** et aux **accès aux bases de données** relationnelles.

Fonctionnalités génériques de l'application

Vous trouverez une description détaillée de la base de données sur Moodle.

A titre d'information, voici une description **générique** des fonctionnalités qui vous seront imposées. Les fonctionnalités réelles à implémenter par chaque groupe seront communiquées en temps utile à chaque groupe.

Au lancement de l'application, la première fenêtre qui apparaît doit contenir un *message d'accueil* et proposer les **différentes fonctionnalités de l'application** sous forme de **menus**.

Les différentes fonctionnalités que l'application doit proposer sont :

- Permettre **l'encodage** via le formulaire le plus convivial possible **d'une *nouvelle ligne*** dans une table bien précise de la base de données.
    - Faites preuve d'imagination dans l'organisation de ce formulaire. Utilisez les différents composants swing vus au cours de façon la plus adéquate. N'oubliez pas que vous pouvez organiser ces composants swing en panneaux (JPanel).
    - Le caractère facultatif de certains des attributs doit impérativement être maintenu (valeur ***null*** et pas 0 pour les numériques ou une suite de blancs pour les chaînes de caractères). En effet, l'utilisateur peut ne pas introduire de valeur pour certains champs du formulaire. Ceci impose une gestion particulière lors de l'élaboration de l'instruction SQL d'insertion de la nouvelle ligne dans la table correspondante.

- **Lister le contenu complet de la table** (y compris les lignes correspondant aux nouvelles insertions éventuellement effectuées lors de la session en cours).

- Permettre **la suppression d'une ligne de la table** répondant à un ou plusieurs critères de sélection.

- Permettre plusieurs **listes (recherches).** Pour chacune, les critères de sélection seront d'abord demandés à l'utilisateur via l'interface Swing adéquate. Après le choix des critères de sélection par

l'utilisateur, chaque ligne résultant de la requête devra apparaître dans une JTable. Attention, cette JTable devra être constituée des colonnes appropriées.

Directives et contraintes pour la réalisation

1. Vous devez au moins une fois :
   - Modifier le contenu du container de la fenêtre principale ;
   - Ouvrir une nouvelle fenêtre de type JFrame (autre que la fenêtre principale).

2. **Tout nouveau (contenu de) container de fenêtre** (ensemble de composants graphiques de type Swing : bouton, listes, …) doit être **décrit dans une classe propre**, **sous-classe de JPanel**. Il est par conséquent interdit de ne prévoir qu'une seule classe *gigantesque* qui implémenterait la fenêtre principale et dans laquelle les composants swing des nouveaux containers à afficher seraient créés par les classes internes gérant les événements.
   Ainsi, par exemple, il est exclu que la méthode *actionPerformed* soit chargée de créer des composants swing. Ce que peut faire la méthode actionPerformed, c'est créer et afficher un objet d'une autre classe (sous-classe de JPanel) qui, elle, se chargera de créer les composants swing et de les disposer au bon endroit à l'aide d'un gestionnaire de tracé.

3. Tout accès à une base de données doit être géré par une **classe générique** *(cf. cours).* Cette classe mise à votre disposition sur Moodle propose les **méthodes génériques <u>statiques</u>** gérant :

   - L'établissement de la connexion ;
   - L'insertion, suppression ou modification de lignes dans des tables ;
   - L'exécution de n'importe quelle requête et la récupération des lignes résultant de cette requête quelle que soit **la requête**, quel que soit **le nombre de lignes** retournées par SQL, quel que soit **le type des colonnes** apparaissant dans la requête.

4. Lors de toute requête dans la base de données, si l'utilisateur a le choix parmi plusieurs critères dont les valeurs sont stockées dans la base de données, vous devez lui proposer la **liste de ces critères** (via liste, comboBox, …). Vous pouvez cependant éventuellement lui permettre d'en proposer un nouveau (autre que les critères proposés).
   Par conséquent, il est probable que le contenu de cette liste ou comboBox doive être récupéré dans une ou l'autre table de la base de

données, ce qui implique un accès à la base de données en vue de remplir la liste des critères proposée à l'utilisateur.

5. Pour rappel, lors de l'insertion d'une nouvelle ligne dans une table, vous devez permettre et **gérer les attributs facultatifs**. L'utilisateur peut donc ne pas introduire de valeur dans les champs facultatifs du formulaire. Vous devez alors gérer les valeurs nulles éventuelles lors de l'insertion dans la base de données (null en SQL). *Il en va de même dans les requêtes* : **vous devez pouvoir récupérer et afficher des lignes comprenant des valeurs inconnues (null).**

6. Votre travail doit IMPERATIVEMENT tourner dans un environnement Eclipse. L'examen (défense individuelle du travail) se déroulera sur machine dans un environnement Eclipse.

7. Votre programme sera testé par le professeur en présence des deux étudiants responsables du projet **lors du dernier cours/labo**. Vous devez également fournir à ce moment au professeur **une copie du code de votre programme.**

Votre travail étant testé au dernier cours, c'est donc cette version qui sera évaluée et cotée, et en aucun cas, une version ultérieure du programme (que vous apporteriez éventuellement le jour de l'examen).

Bon Travail !