

Énoncé du Travail Pratique 1

- Partie 1 -

17 février 2017

Préparé par
Daniel Huot et Benjamin Lemelin

1 Résumé du travail

L'application à développer devra permettre de calculer le poids d'un composé chimique à partir de sa formule écrite. Votre application devra fonctionner sur deux plateformes : en mode console sur la plateforme Java SE et en mode graphique sur la plateforme Android.

2 Conditions de réalisation du travail

Valeur sur la note finale	Type	Durée	Nombre de remises
10 %	Individuel	2 semaines	2

3 Travail à effectuer – Remise 1

Cette section vise à vous présenter les différentes étapes du travail à effectuer pour cette première remise.

3.1 Importer et comprendre le code

Vous trouverez avec cet énoncé le projet de base à utiliser pour cette remise. C'est un petit projet Java dont le « Build Process » est guidé par Gradle, exactement comme sur la plateforme Android. Dans IntelliJ IDEA, ouvrez simplement le projet et à l'invite, conservez les paramètres par défaut avant d'appuyer sur « OK ».

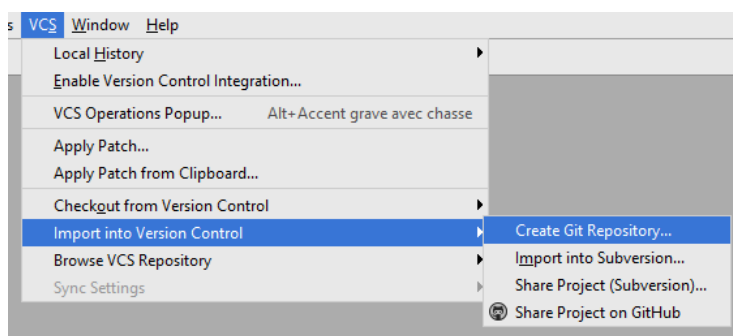
Ne soyez pas surpris que le projet ne compile pas. Passez à l'étape suivante, nous y reviendrons.

3.2 Importez le code sur GitHub, GitLab ou BitBucket

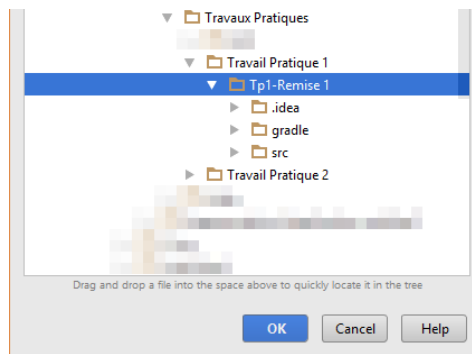
Le code de votre projet devra obligatoirement être hébergé sur un gestionnaire de code source. Pour ce travail, vous devrez utiliser [Git](#). Téléchargez [Git](#) et installez-le.

Créez-vous un compte [GitHub](#), [GitLab](#) ou [BitBucket](#). Ayez une préférence pour [GitLab](#), qui permet de créer des dépôts de code privés, et ce, gratuitement.

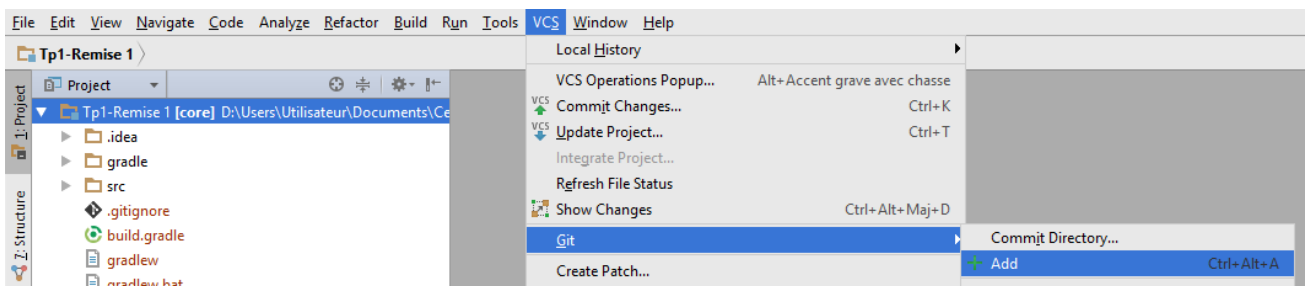
À partir de votre compte, créez un nouveau dépôt privé. Notez l'adresse de votre dépôt débutant par « https » et finissant par « .git ». Elle vous sera utile plus tard. Ensuite, importez votre code sous Git.



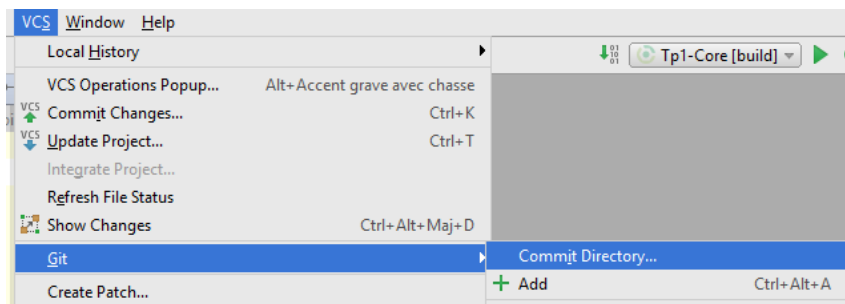
Sélectionnez la racine de votre projet (où se trouve le dossier « src ») et appuyez sur « OK ».



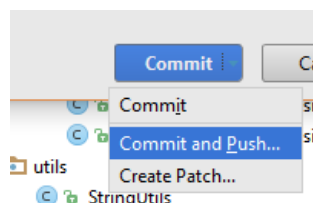
Dans l'explorateur de projet, sélectionnez la racine de votre projet. Ensuite, ajoutez les fichiers à « Git ».



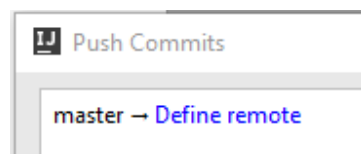
Les noms de vos fichiers deviendront verts. Ensuite, faites votre premier « Commit ».



Entrez une description (un message) et cliquez sur « Commit and Push ». Si l'on vous mentionne qu'il y a des « Warnings » ou des « Erreurs », faites quand même un « Commit ».

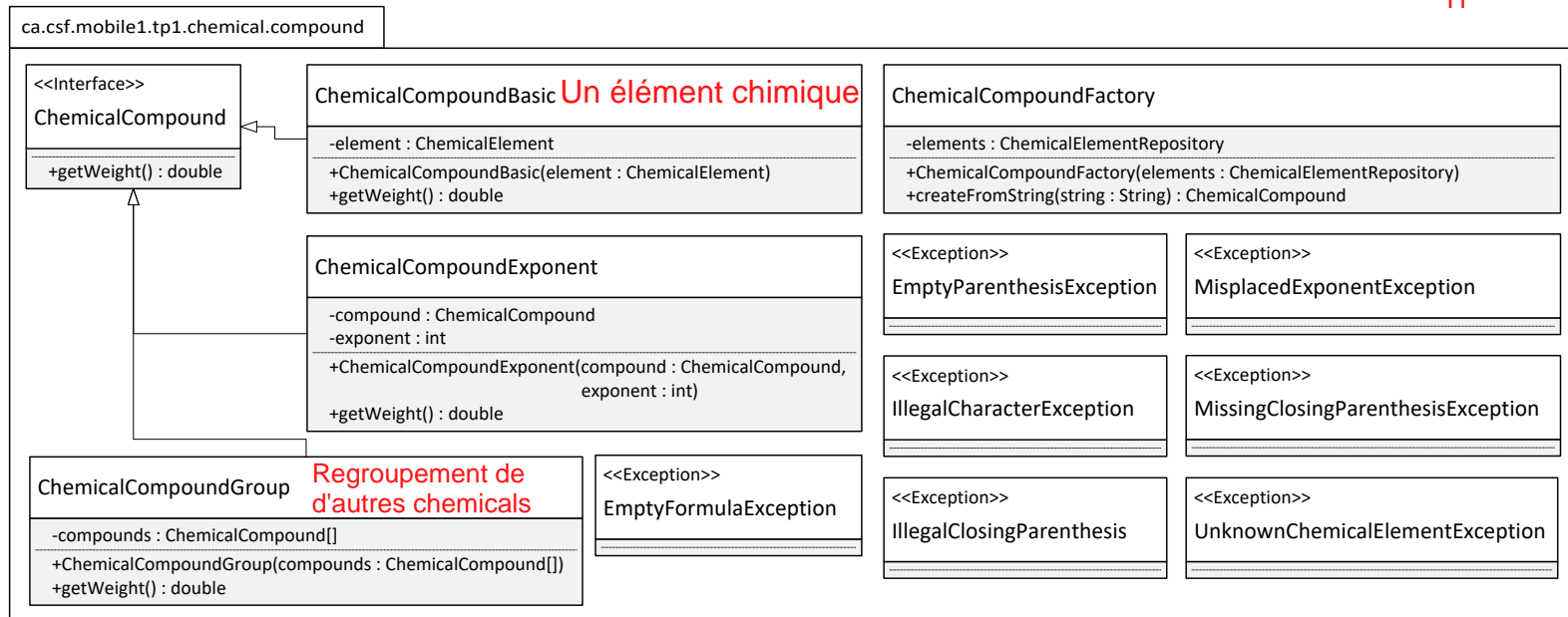
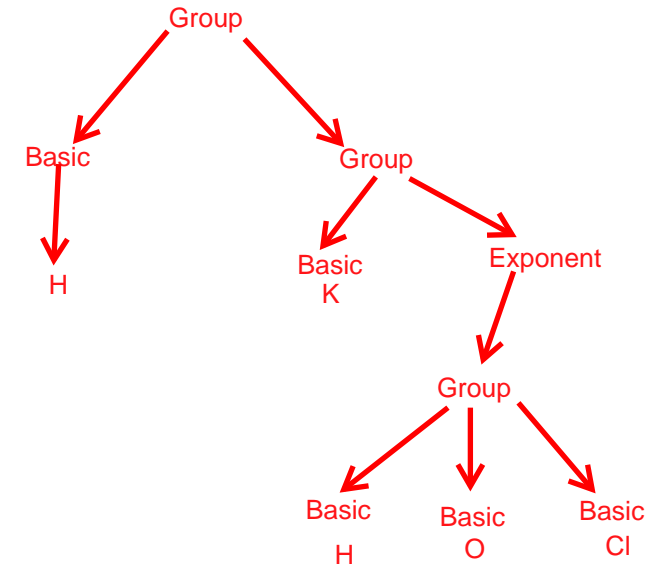
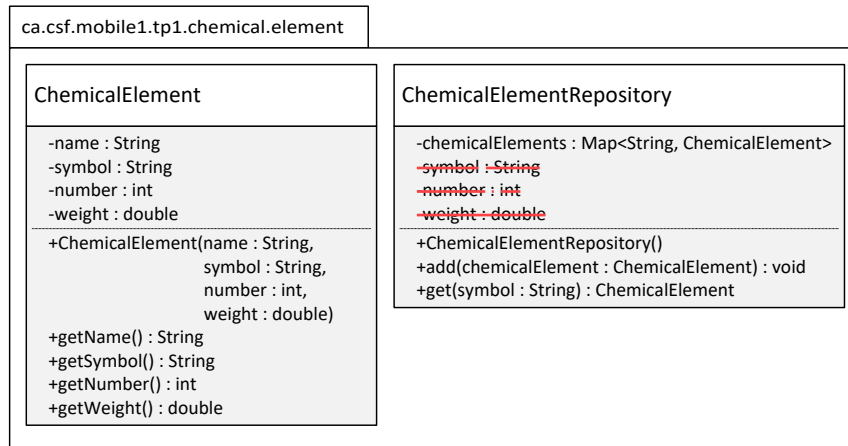


Ensuite, dans la nouvelle fenêtre qui apparaîtra, cliquez sur « Define remote » et entrez l'URL vers votre dépôt. On vous demandera alors votre nom d'utilisateur et votre mot de passe. Ensuite, cliquez sur « Push ».



Structures en arbre = structure récursive comme Algo

3.3 Implémenter la logique d'affaire de l'application



La page précédente présente le diagramme de classe de la couche « Modèle » de vos applications, qui serviront à calculer le poids de composés chimiques à partir de chaînes de caractères telles que :

- **H2O**
- **NaCl**
- **(NaCl) 4**
- **(H2SO4 (Be) 3 (H2O)) 2**

Ces chaînes de caractères pourront contenir :

- Des symboles d'éléments chimiques du tableau périodique, tel que « **H** », « **Na** » et « **Be** ».
- Des parenthèses contenant une sous-formule, tel que « **(NaCl)** » et « **(H(Be)(O))** ».
- Des exposants placés avant un symbole ou une parenthèse, tel que « **H2** » et « **(NaCl)4** ».

Le calcul du poids d'une formule chimique est assez simple. Chaque élément tel que « **H** », « **Na** » et « **Be** » possède un poids. Ces derniers sont tout simplement additionnés ensemble, l'un à la suite de l'autre. Par exemple :

- **NaCl = Na + Cl = 22,98976928 + 35,4527 = 58,44246928 g/mol**

Notez qu'un symbole débute toujours par une majuscule. Si le symbole fait plus d'une lettre, les lettres subséquentes seront systématiquement des minuscules.

Lorsqu'il y a un chiffre, il faut tout simplement multiplier le poids de l'élément précédent par ce chiffre. Par exemple :

- **H2O = H2 + O = H * 2 + O = 1,00794 * 2 + 15,9994 = 18,01528 g/mol**

Enfin, les parenthèses permettent de faire des regroupements, tout comme n'importe quel calcul. Les parenthèses peuvent être imbriquées et être affectées par un multiplicateur. Les parenthèses qui se suivent sont additionnées. Par exemple :

- **(NaCl) 4 = (22,98976928 + 35,4527) * 4 = 233,76987712 g/mol**

Voici le détail d'un composant plus complexe :

- **(H2SO4 (Be) 3 (H2O)) 2 = (H2 + S + O4 + (Be) 3 + (H2 + O)) 2**
= (H * 2 + S + O * 4 + (Be) * 3 + (H * 2 + O)) * 2
= (1,00794 * 2 + 32,066 + 15,9994 * 4 + (9,012182) * 3 + (1,00794 * 2 + 15,9994)) * 2
= 286,262612 g/mol

Si vous désirez vérifier vos calculs, vous pouvez utiliser le site web à [ce lien](#). Ce site prend en compte d'autres paramètres, et vos résultats pourraient différer. Cependant, la différence sera toujours minime ($\pm 0,1$ g/mol).

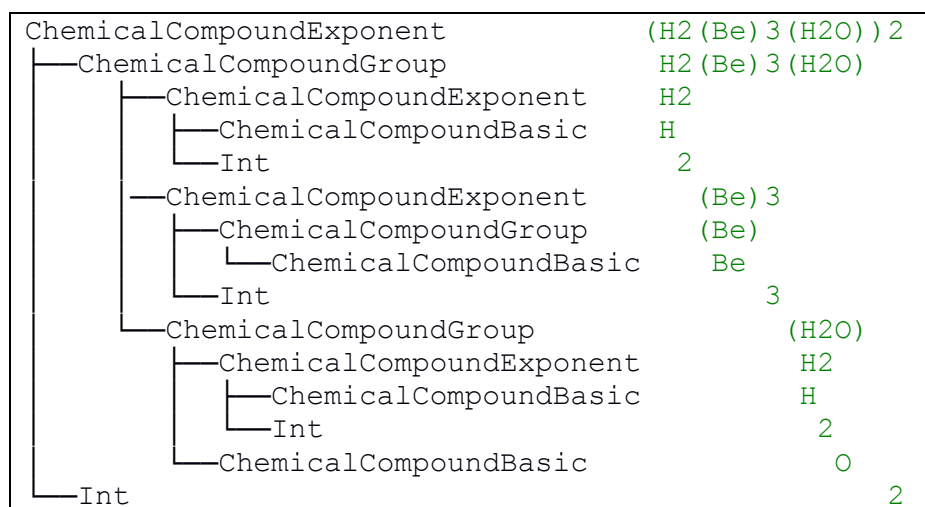
Consultez un tableau périodique (tel que [celui-ci](#)) pour voir le poids des éléments. Voici comment le lire :

Numéro d'identification	→	3	2 1
Symbole	→	Li	
Nom complet	→	Lithium	
Poids en g/mol	→	6,94	

La classe « ChemicalCompoundFactory » devra interpréter les chaînes de caractères qui lui sont envoyés et retourner un « ChemicalCompound » représentant la formule chimique. Vous devrez utiliser les différentes variantes de « ChemicalCompound » pour y arriver.

- Un « ChemicalCompoundBasic » ne contient qu'un seul « ChemicalElement » et retourne son poids.
- Un « ChemicalCompoundGroup » contient un ou plusieurs « ChemicalCompound » et retourne la somme de leurs poids combinés.
- Un « ChemicalCompoundExponent » contient un « ChemicalCompound » et un multiplicateur retourne son poids multiplié par ce multiplicateur.

Avec tout cela, vous devrez être en mesure de représenter n'importe quelle formule chimique. Par exemple, pour « $(H_2(Be)3(H_2O))_2$ » :



Il suffira alors d'appeler la méthode « `getWeight()` » sur la racine de cette structure pour calculer le poids total de ce composé chimique. En interne, le calcul se fera alors en traversant la structure au grand complet.

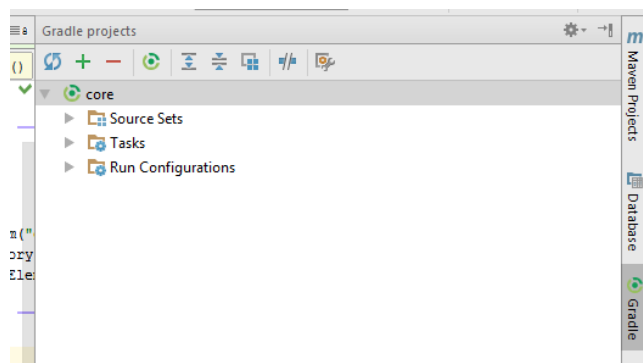
Notez que vous devrez effectuer des validations lors de l'interprétation de la chaîne de caractères, d'où les diverses exceptions dans le diagramme de classe. Parmi les erreurs possibles :


- Il est interdit d'avoir une formule vide.
- Il est interdit d'avoir une parenthèse vide.
- Il est interdit d'utiliser autre chose que des chiffres, des lettres et des parenthèses.
- Il est interdit d'avoir une parenthèse orpheline, sans sa parenthèse ouvrante ou fermante.
- Il est interdit d'appliquer un exposant sur autre chose qu'une parenthèse ou un symbole chimique.
- Il est interdit d'utiliser des symboles chimiques inconnus.

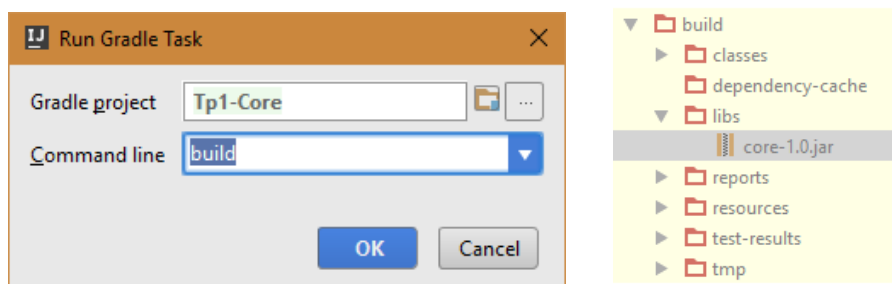
N'oubliez pas de documenter votre code selon les standards montrés en classe et de vous assurer que les tests fournis avec le projet de départ passent tous. À ce sujet, vous aurez à compléter les tests où il y a des « TODO » pour qu'ils fonctionnent.

3.4 Compiler la librairie

Pour créer un fichier « .jar » contenant vos classes, vous devrez faire un « Build » Gradle. Pour ce faire, ouvrez l'onglet Gradle.



Cliquez sur le bouton «  ». Un dialogue s'ouvrira, vous demandant une commande. Tapez « build » et faites « Ok ». Après la compilation, vous trouverez votre fichier « .jar » dans le dossier « build/libs ». Notez que Gradle refusera de faire un « Build » si vos tests ne passent pas. Cela vous oblige à faire une librairie de qualité.



4 Modalités de remise

Remettez sur LÉA à l'heure et à la date indiquée par votre professeur un fichier texte comportant l'adresse vers votre dépôt Git. Cette adresse sera utilisée pour effectuer un « git clone ». Ajoutez à ce fichier votre nom, votre matricule ainsi que tout commentaire pertinent à la correction. Enfin, le nom du fichier texte doit être votre matricule suivi de « .txt ».

Une seule journée de retard est tolérée entraînant alors une pénalité de 15 % de la note. Au-delà de ce délai, le travail est refusé et la note « 0 » est attribuée.

5 Évaluation

Ce travail sera évalué avec une correction normale et une correction négative. Les points seront retirés en fonction de la gravité de la faute, de la qualité globale du travail et de l'effort mis par l'étudiant. Le tableau suivant vise à résumer les éléments pouvant réduire la note, jusqu'à atteindre la note de 0 %.

Éléments	Pondération
Fonctionnalités : <ul style="list-style-type: none"> • Support des symboles d'éléments chimiques dans les formules. (0176-1, 0176-2) • Support des parenthèses dans les formules. (0176-1, 0176-2) • Support des exposants dans les formules. (0176-1, 0176-2) • Exception lancée lorsqu'une parenthèse vide est rencontrée. (0176-1, 0176-2) • Exception lancée lorsqu'un caractère illégal est rencontré. (0176-1, 0176-2) • Exception lancée lorsqu'une parenthèse illégale est rencontrée. (0176-1, 0176-2) • Exception lancée lorsqu'une parenthèse non fermée est rencontrée. (0176-1, 0176-2) • Exception lancée lorsqu'un exposant mal placé est rencontré. (0176-1, 0176-2) • Exception lancée lorsqu'un élément chimique inconnu est rencontré. (0176-1, 0176-2) • En mesure de saisir une formule chimique dans l'application. (0176-1, 0176-2) • En mesure de calculer le poids de la formule chimique saisie. (0176-1, 0176-2) • Affichage de messages d'erreur s'il y a erreur dans la formule chimique. (0176-1, 0176-2) 	<p>5 % Correction normale</p>
Interface utilisateur (incluant, mais sans s'y limiter) : <ul style="list-style-type: none"> • Interface console malpropre. (017D-4) • Non-respect de la grille de 8dp sous Android. (017D-4) • Éléments d'interface mal alignés sous Android. (017D-4) • Non-utilisation de « Snackbar » pour les messages sous Android. (017D-4) • Fichiers XML de « Layout » Android malpropres. (017D-4) • « Layouts » Android instables sous différentes tailles et densités d'écran. (017D-4) • Expérience utilisateur désagréable sous Android. (017D-4) • Perte de données lors d'un changement d'orientation. (017D-3, 017D-4) 	<p>15 % Correction négative</p>
Documentation du code (incluant, mais sans s'y limiter) : <ul style="list-style-type: none"> • Classe non documentée ou documentée de manière insuffisante. (017D-7) • Méthode non documentée ou documentée de manière insuffisante. (017D-7) • Paramètres non documenté ou documenté de manière insuffisante. (017D-7) • Valeur de retour non documenté ou documenté de manière insuffisante. (017D-7) • Lien non présent vers une classe référencé dans la documentation. (017D-7) • Documentation malpropre. (017D-7) • Indentation incorrecte de la documentation. (017D-7) • Documentation qui n'est pas en français. (017D-7) • Standard de documentation « JavaDoc » non respecté. (017D-7) 	<p>10 % Correction négative</p>

Tests unitaires (incluant, mais sans s'y limiter) : <ul style="list-style-type: none"> • Altération d'un test unitaire en dehors de ce qui est permis. (0176-4, 0177-3) • Suppression d'un test unitaire sans autorisation. (0176-4, 0177-3) • Vérification échouant dans un test unitaire. (0177-1, 0177-2, 0177-3) • Classe publique non testée. (0176-4, 0177-1) • Méthode publique non testée. (0176-4, 0177-1) • Classe de test qui n'est pas dans le dossier « src/test/java ». (0177-2, 0177-3) • Vérification manquante dans un test. (0176-4, 0177-1) • Test vérifiant plus qu'une fonctionnalité. (0177-1) • Utilisation de la mauvaise vérification « JUnit ». (0177-2) 	10 % Correction négative
Qualité générale de l'architecture (incluant, mais sans s'y limiter) : <ul style="list-style-type: none"> • Contrôleur effectuant de la logique réservée à la couche modèle. (017D-3, 017D-5) • Séparation incorrecte entre la vue et le modèle. (017D-3, 017D-5) • Abstractions non utilisées où cela aurait été nécessaire. (016T-1) • Référence concrète où une référence abstraite aurait été suffisante. (016T-3) • Pas d'usage de l'héritage où cela aurait été nécessaire. (016T-1, 016T-3) • Non-respect du diagramme de classe. (016T-2) • Méthode ayant trop de responsabilités pas découpées convenablement. (016T-3) • Classe ayant trop de responsabilités pas découpées convenablement. (016T-3) • Découpage en « package » insuffisant. (016T-2) • Nom d'un « package » imprécis. (016T-3) • « dot.case » non utilisé pour un nom de « package ». (016T-3) 	30 % Correction négative
Qualité générale du code (incluant, mais sans s'y limiter) : <ul style="list-style-type: none"> • Logique applicative « patchée ». (016T-3) • Nom d'une variable imprécis. (016T-3) • Nom d'une classe imprécis. (016T-3) • Nom d'une interface imprécis. (016T-3) • Nom d'une constante imprécis. (016T-3) • Nom d'une énumération imprécis. (016T-3) • Nom d'une méthode imprécis. (016T-3) • « camelCase » non utilisé pour un nom de variable ou de méthode. (016T-3) • « PascalCase » non utilisé pour un nom d'interface, de classe ou d'énumération. (016T-3) • « UPPER_CASE » non utilisé pour un nom de constante. (016T-3) • Manque d'une constante ou cela aurait été nécessaire. (016T-3) • Méthode sans attribut de visibilité. (016T-3) • Méthode publique qui devrait être privée. (016T-3) • Méthode statique qui ne devrait pas l'être. (016T-3) • Méthode protégée qui ne devrait pas l'être. (016T-3) • Attribut de classe qui n'est utilisé que dans une seule méthode. (016T-3) • Commentaire inutile ne compensant pas le manque d'expressivité du code. (016T-3) • Erreur à la compilation. (016T-3, 016T-5) • Avertissement à la compilation. (016T-3, 016T-5) 	20 % Correction négative
Qualité générale du travail (incluant, mais sans s'y limiter) : <ul style="list-style-type: none"> • Corruption de la configuration Gradle des projets. (016T-5) • Non-respect des consignes de remise. (016T-5) • Fichiers inutiles remis avec les projets. (016T-5) • Travail non remis via l'outil gestion de code source Git. (0176-3) 	10 % Correction négative

Notez que la qualité écrite de la langue française fait partie intégrante de l'évaluation de ce travail. Étant donné que l'accès aux ressources de la langue française est permis, chaque faute de français retirera donc 0,5 % à la note finale jusqu'à concurrence de 20 % retirés. Sont corrigées les fautes d'orthographe, les fautes de grammaire, les fautes syntaxiques et les fautes de ponctuation.

6 Compétences reliées au travail pratique

Voici la liste des compétences évaluées, évaluées partiellement ou non-évaluées par ce travail. Vous trouverez les explications sur ces choix en dessous de cette liste.

- | | |
|--|---|
| • 016T-1 Créer un modèle objet : Évalué. | • 0177-1 Planifier les tests : Évalué. |
| • 016T-2 Perfectionner le modèle objet : Évalué. | • 0177-2 Procéder à l'exécution des différents tests : Évalué. |
| • 016T-3 Procéder à la codification d'une classe : Évalué. | • 0177-3 Valider la qualité de l'application : Évalué. |
| • 016T-5 Générer la version exécutable du programme : Évalué. | • 017D-2 Établir le cadre technologique : Non évalué. |
| • 0176-1 Analyser les caractéristiques de l'application : Évalué. | • 017D-3 Préparer le travail de développement : Évalué. |
| • 0176-2 Analyser la nature des améliorations à apporter à l'application : Évalué. | • 017D-4 Produire le prototype de présentation : Évalué. |
| • 0176-3 Ajouter et modifier des fonctionnalités à une application : Évalué. | • 017D-5 Produire le prototype de communication : Évalué. |
| • 0176-4 Valider le fonctionnement de l'application : Évalué. | • 017D-6 Développer l'application : Non évalué. |
| | • 017D-7 Produire la documentation relative à l'application : Évalué. |

La compétence 017D-2 sera évaluée au travail pratique 2 lorsque l'étudiant sera amené à effectuer des requêtes à un service web à partir de son application mobile. La compétence 017D-6 sera évaluée au travail pratique 3 lorsque l'étudiant devra utiliser une base de données relationnelle pour son application mobile.