Nom : _____

Prénom : _____

Identifiant : _____ Groupe : _____ Enseignant : _____

/20

HE2B ESI

Haute École Bruxelles-Brabant
École Supérieure d'Informatique
Bachelor en Informatique

janvier 2019
DEV1
NRI

DEV1 – Développement I

Laboratoire de langage Java

Partie pratique JAVL

Consignes

1. L'examen dure 3 heures.
2. Cette partie compte pour $\frac{5}{10}$ de la cote de l'unité.
3. Vous pouvez utiliser vos notes et internet *en lecture seule*.
4. Vous n'êtes pas obligé d'écrire le code dans l'ordre des questions.
5. Vous pouvez utiliser du papier brouillon.
6. Vous devez écrire et utiliser des méthodes lorsque c'est pertinent.

Important en vue de la remise

Vous utiliserez Git pour la remise.

1. Créez un projet Netbeans intitulé `dev1-javl-janvier-g51243` où `g51243` est votre identifiant.
2. Effectuez immédiatement un premier commit avec le message "Commit initial, projet vide".
3. Pour remettre votre code, faites un dernier commit et poussez-le (push) sur le dépôt Gitlab créé par votre professeur.

Si vous ne parvenez pas à remettre avec Git, vous pouvez compresser le dossier de votre projet et déposer le fichier compressé dans le eCasier de votre professeur. Ceci vous fera perdre maximum 2 points sur 20.

Le présent examen vise à construire un logiciel très rudimentaire de gestions des notes pour un·e prof qui donne un cours de math et un cours de français dans une classe donnée. Chaque étudiant·e aura son nom (une chaîne de caractères) et ses notes (des entiers entre 0 et 20) encodées par l'enseignant.

1 Accueil (2 points)

1. Dans le projet créé (voir ci-dessus), créez une classe `Janvier`.
2. Dans la méthode principale de cette classe, affichez sur la sortie standard "Système d'encodage des notes."
3. Demandez à l'utilisateur ou l'utilisatrice son prénom, et souhaitez-lui la bienvenue : "Bienvenue <nom donné>".

Effectuez un commit avec un message de commit pertinent avant de passer à une autre question.

2 Lecture robuste (2 points)

Écrivez une méthode

```
int lireEntier(String message)
```

qui permet de demander (en affichant le message en paramètre) un nombre entier à l'utilisateur ou l'utilisatrice de façon robuste : lorsque l'utilisateur ou l'utilisatrice se trompe, la méthode affiche "Ceci n'est pas un entier" et re-demande jusqu'à obtenir une réponse entière.

Écrivez la javadoc de cette méthode.

3 Encodage des noms (3 points)

Dans votre méthode principale, demandez à l'enseignant le nombre d'étudiant·e·s de sa classe. Écrivez et utilisez ensuite une méthode

```
String[] encoderNoms(int nbEtudiants)
```

pour permettre à l'utilisateur ou l'utilisatrice d'encoder les noms de ces étudiant·e·s.

4 Encodage des notes (3 points)

Dans la méthode principale, déclarez et créez deux tableaux (`notesMath` et `notesFrançais`) de taille adaptée.

Écrivez et utilisez une méthode

```
void encoderNotes(String[] noms, int[] notesMath, int[] notesFrançais)
```

qui permet de demander à l'utilisateur ou l'utilisatrice les notes (nombres entiers) de chaque étudiant·e, et les inscrit dans ces deux tableaux.

Voici un exemple d'utilisation de votre programme jusqu'ici :

```
Système d'encodage des notes
Encodez votre nom : Nicolas
Bienvenue Nicolas
Combien d'étudiant·e·s avez-vous : 2
Nom de l'étudiant·e numéro 0
Atchoum
Nom de l'étudiant·e numéro 1
Dormeur
Encoder les notes pour Atchoum
Math : 14
```

```
Français : 19
Encoder les notes pour Dormeur
Math : 12
Français : 10
```

5 Tests unitaires (2 points)

Pour la prochaine question, on vous demande d'écrire les tests unitaires.

6 Moyenne de chaque élève (3 points)

Écrivez une méthode qui produit un tableau d'entiers dont chaque élément est la moyenne d'un élève de la classe. Lorsque la moyenne théorique n'est pas entière, il faut effectuer l'arrondi vers le bas si elle est strictement inférieure à 10, et vers le haut sinon. Par exemple un 9.2, un 9.5 ou un 9.9 produira 9, mais 10.2 ou 10.5 ou 10.9 produira 11.

Cette méthode recevra comme arguments les deux tableaux `notesMath` et `notesFrançais`.

```
int[] moyennesÉlèves(int[] notesMaths, int[] notesFrançais)
```

La méthode doit lancer une `IllegalArgumentException` si les tableaux reçus en paramètre n'ont pas la même taille.

N'oubliez pas les tests unitaires pour cette méthode.

7 Recherche (2 points)

Écrivez une méthode qui retourne la note d'un·e étudiant·e dont le nom est donné. Cette méthode aura pour signature :

```
int note(String étudiant, String[] noms, int[] notes)
```

Lancez une `IllegalArgumentException` si les tableaux n'ont pas la même dimension. Lancez une `NoSuchElementException` si l'étudiant·e n'est pas trouvé. (Vous avez besoin d'ajouter un `import java.util.NoSuchElementException`.)

8 Conclusion (3 points)

Vous allez maintenant complétez votre méthode principale afin d'offrir à l'utilisateur ou l'utilisatrice la possibilité d'afficher la moyenne de chacun des étudiant·e·s de son choix grâce à leur nom. Si le nom demandé n'existe pas, le programme le signale par un message mais continue son exécution.

Voyez à la dernière page de l'examen pour un exemple d'exécution du programme, illustrant ce qui est attendu.

N'oubliez pas d'effectuer un dernier commit avant de faire votre push final.

Voici à quoi pourrait ressembler une exécution complète du programme :

```
Système d'encodage des notes
Encodez votre nom : Prof
Bienvenue Prof
Combien d'étudiant·e·s avez-vous : 3
Nom de l'étudiant·e numéro 0
Atchoum
Nom de l'étudiant·e numéro 1
Dormeur
Nom de l'étudiant·e numéro 2
Blanche
Encodez les notes pour Atchoum
Math : 20
Français : 12
Encodez les notes pour Dormeur
Math : 18
Français : 19
Encodez les notes pour Blanche
Math : 8
Français : 4
Donnez le nom d'un·e étudiant·e (ou 'quit' pour quitter): Blanche
La note de Blanche est 6
Donnez le nom d'un·e étudiant·e (ou 'quit' pour quitter): Dormeur
La note de Dormeur est 18
Donnez le nom d'un·e étudiant·e (ou 'quit' pour quitter): Personne
Étudiant·e non-trouvé·e: Personne
Donnez le nom d'un·e étudiant·e (ou 'quit' pour quitter): quit
Au revoir !
```