

# Bases de la programmation impérative

Ensimag 1A

Examen intermédiaire 2017

On s'intéresse pour cet examen à la lecture et au traitement d'un fichier de notes au format *CSV*.

Avant de démarrer prenez 1 minute pour remplir les variables `NOM`, `PRENOM`, `GROUPE` du fichier *stats.py*.

L'accent est mis sur la justesse du code réalisé ainsi que sur sa clarté et son style.

N'oubliez pas *pylint*...

## 1 Fichier d'entrée

Le format *CSV* est un format de fichier très simple pour le stockage de tableaux de données. Un fichier contient un unique tableau 2D. Chaque ligne du fichier est une ligne du tableau et sur chaque ligne les données de chaque colonne sont séparées par un séparateur ; ici, des virgules.

Dans notre cas, chaque ligne correspond aux données d'un étudiant et les colonnes sont les suivantes :

1. prénom,
2. nom,
3. note à l'examen numéro 0 (entier),
4. note à l'examen numéro 1 (entier),
5. note à l'examen numéro 2 (entier).

Si l'on examine par exemple les trois premières lignes du fichier *notes.csv* on voit que Lilia Markowitz a une note de 15 à l'examen 0, 16 à l'examen 1 et 6 à l'examen 2. À la ligne suivante, on passe à un nouvel étudiant : Aura Rebello...

```
Lilia,Markowitz,15,16,6
```

```
Aura,Rebello,15,0,8
```

```
Torrie,Thackston,12,16,9
```

On se propose de stocker les données de chaque étudiant dans une classe `Etudiant`, définie dans le fichier *stats.py*, qui est le fichier que vous devez compléter. On a trois attributs : `prenom` et `nom` qui sont des chaînes de caractères et `notes` qui est un triplet d'entiers entre 0 et 20.

La première fonction à programmer est la lecture du fichier. On vous demande de programmer à l'aide de `yield` un générateur `parcours_notes` permettant, à partir d'un nom de fichier CSV, d'itérer sur tous les étudiants stockés dans ce fichier. Votre générateur devra créer chaque instance de la classe `Etudiant` à la volée.

On rappelle que toute chaîne de caractères dispose d'une méthode `split` permettant de la décomposer en sous-chaînes selon un séparateur donné. Par exemple, `"foo,bar,baz".split(",")` vaut `["foo", "bar", "baz"]`.

## 2 Fonctions d'analyse

Il s'agit ensuite d'écrire un certain nombre de fonctions d'analyse et de tri des étudiants.

Toutes ces fonctions **devront être testées au fur et à mesure** en lançant le programme *analyse.py* et en comparant la sortie obtenue avec celle attendue. Cette dernière est présentée à la fin de ce document.

Les fonctions à compléter prennent toutes en entrée un itérable sur les étudiants (ce peut être un vecteur, mais aussi un itérateur.) On cherche lorsque c'est possible à réaliser les traitements en une seule passe sur les données.

Les fonctions à implémenter sont les suivantes :

1. `calcul_moyennes` : calcule la moyenne de la promo à chaque examen (renvoie un triplet de nombres flottants) ;
2. `etudiants_brillants` : un itérateur sur les étudiants dont au moins une note vaut 20 (si possible à l'aide de `filter`) ;
3. `min_et_max` : renvoie un triplet de couples de vecteurs : pour chaque épreuve, le couple constitué du vecteur de tous les étudiants ayant la note minimale à cette épreuve et du vecteur de ceux ayant la note maximale (ces vecteurs contiennent des objets de la classe `Etudiant`) ;
4. `classement` : on calcule la moyenne pour le module de chaque étudiant, les coefficients étant 1, 1 et 2 (pour les épreuves 0, 1 et 2 respectivement). On renvoie un vecteur de couples (nom, moyenne) contenant les couples correspondants pour tous les étudiants, classés de la moins bonne à la meilleure moyenne.

Dans tous les cas, on pose en précondition que l'ensemble des étudiants est non vide...

## 3 Histogramme

Dernière question, on cherche à visualiser graphiquement les notes obtenues à chaque épreuve. On vous demande de compléter la fonction `histogramme` qui prend un itérable sur les étudiants, ainsi qu'un numéro d'épreuve (entre 0 et 2), et génère un histogramme en bâton en svg (et renvoie le nom du fichier créé).

On se propose de dessiner pour chaque note possible (de 0 à 20) une barre verticale sous forme d'un rectangle rouge. Pour chaque note  $n$ , la largeur du rectangle est fixée à 20 et sa hauteur à 20 fois le nombre d'étudiants ayant une note de  $n$ .

Vous pouvez trouver différents exemples de fichiers svg dans votre répertoire, ou ci-dessous.

```

<svg width='420' height='320'>
<rect width='420' height='320' fill='white' />
<text x='0' y='320'>0</text>
<text x='20' y='320'>1</text>
<text x='40' y='320'>2</text>
<text x='60' y='320'>3</text>
<text x='80' y='320'>4</text>
<text x='100' y='320'>5</text>
<text x='120' y='320'>6</text>
<text x='140' y='320'>7</text>
<rect width='20' height='20' x='160' y='280' fill='red' />
<text x='160' y='320'>8</text>
<text x='180' y='320'>9</text>
<rect width='20' height='20' x='200' y='280' fill='red' />
<text x='200' y='320'>10</text>
<rect width='20' height='20' x='220' y='280' fill='red' />
<text x='220' y='320'>11</text>
<rect width='20' height='120' x='240' y='180' fill='red' />
<text x='240' y='320'>12</text>
<rect width='20' height='140' x='260' y='160' fill='red' />
<text x='260' y='320'>13</text>
<rect width='20' height='300' x='280' y='0' fill='red' />
<text x='280' y='320'>14</text>
<rect width='20' height='180' x='300' y='120' fill='red' />
<text x='300' y='320'>15</text>
<rect width='20' height='60' x='320' y='240' fill='red' />
<text x='320' y='320'>16</text>
<rect width='20' height='60' x='340' y='240' fill='red' />
<text x='340' y='320'>17</text>
<rect width='20' height='20' x='360' y='280' fill='red' />
<text x='360' y='320'>18</text>
<rect width='20' height='40' x='380' y='260' fill='red' />
<text x='380' y='320'>19</text>
<rect width='20' height='20' x='400' y='280' fill='red' />
<text x='400' y='320'>20</text>
</svg>

```

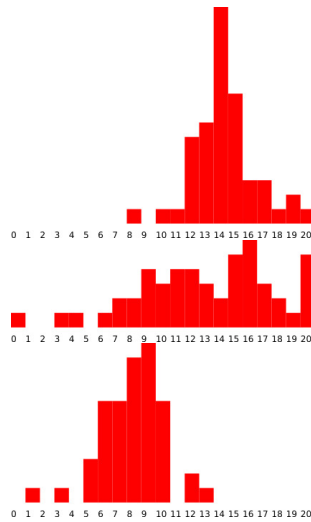


FIGURE 1 – les trois histogrammes à obtenir

## 4 Sortie attendue

Normalement, votre programme devrait générer le texte suivant :

```
*****
pour l'épreuve numero 0 la moyenne est 14.24
pour l'épreuve numero 1 la moyenne est 12.9
pour l'épreuve numero 2 la moyenne est 7.92
*****
les etudiants ayant au moins un 20 sont : ['Hogue', 'Bukowski', 'Seth', 'Tiedemann', 'Ma
*****
pour l'épreuve numero 0
les moins bons sont : ['Sanden']
les meilleurs sont : ['Hogue']
pour l'épreuve numero 1
les moins bons sont : ['Rebello']
les meilleurs sont : ['Hogue', 'Bukowski', 'Seth', 'Tiedemann', 'Mackie']
pour l'épreuve numero 2
les moins bons sont : ['Tiedemann']
les meilleurs sont : ['Proulx']
*****
le classement est : ('Dey', 6.75)
('Cundiff', 7.5)
('Rebello', 7.75)
('Portillo', 8.25)
('Fitzpatrick', 9.0)
```

('Bottomley', 9.0)  
('Cokley', 9.0)  
('Tiedemann', 9.0)  
('Winnie', 9.25)  
('Sanden', 9.25)  
('Morris', 9.25)  
('Fielder', 9.25)  
('Hairston', 9.5)  
('Orta', 9.5)  
('Bevis', 9.5)  
('Zwick', 10.0)  
('Ketelsen', 10.25)  
('McMillin', 10.25)  
('Henninger', 10.25)  
('Barham', 10.5)  
('Martyn', 10.5)  
('Grothe', 10.5)  
('Callery', 10.5)  
('Markowitz', 10.75)  
('Desrochers', 10.75)  
('Rosner', 10.75)  
('Frutos', 11.0)  
('Lawry', 11.0)  
('Pellegrino', 11.25)  
('Goo', 11.25)  
('Thackston', 11.5)  
('President', 11.5)  
('Heffner', 11.5)  
('Toney', 11.5)  
('Thorne', 11.75)  
('Ricotta', 11.75)  
('Culbert', 11.75)  
('Bukowski', 12.0)  
('Cuthbert', 12.0)  
('Mackie', 12.0)  
('Schmidtke', 12.25)  
('Narron', 12.25)  
('Breedon', 12.5)  
('Vanderford', 12.5)  
('Mestayer', 12.5)  
('Hartzell', 12.5)  
('Proulx', 13.0)  
('Seth', 13.25)

('Bordeau', 13.5)

('Hogue', 14.5)

\*\*\*\*\*