Devoir en temps libre n° 1

Ce premier devoir en temps libre est à préparer pour le jeudi 11 octobre 2018 et sera à rendre pendant le TP, à 13h15 (vous aurez 15 minutes pour prendre connaissance de vos éventuelles erreurs et pour les corriger).

⚠

Vous devez apporter votre programme Python sur une clé USB *et* vous l'être envoyé par mail depuis chez vous.

Le programme doit se présenter sous la forme d'un unique fichier `.py` dont le nom sera obligatoirement et exactement `dm1-login.py` où `login` est votre nom d'utilisateur du laboratoire d'informatique (tout en minuscules avec les tirets).

Le modèle à compléter se trouve dans le répertoire du cours en suivant le chemin `~/mpsi3/informatique/dm/dm1/dm1-login.py` et sur le site du cours `https://cpge.info`. Il faut évidemment remplacer `login` par votre nom d'utilisateur. Par exemple, mon fichier sera `dm1-nicolas-pecheux.py`. L'encodage du fichier devra être UTF-8 (*Fichier* > *Encodage* > utf-8 sous Pyzo).

Le code devra impérativement compiler : *aucune* erreur ne doit apparaître lorsque l'on exécute le script avec F5. Il est impératif de veiller au strict respect de toutes ces consignes, car le programme sera évalué automatiquement.

Seul le code Python est à rendre, il n'y a rien à rédiger sur feuille.

✔

On prendra bien soin de tester toutes les fonctions sur des cas triviaux, puis un peu plus complexes.

Lorsque l'on pose des hypothèses sur les arguments (par exemple qu'un argument est un entier naturel), il n'est pas nécessaire dans vos fonctions de le vérifier ni de prévoir le comportement si l'hypothèse est violée (par exemple si l'argument est négatif).

Exercice 1

Définir en Python une fonction `fonction_conditionnelle(x)` qui implémente la fonction f définie de la manière suivante :

$$f : \begin{array}{ccc} \mathbb{R} & \longrightarrow & \mathbb{R} \\ x & \longmapsto & \begin{cases} 1 & \text{si } x < 0 \\ x+1 & \text{si } 0 \leq x \leq 2 \\ -3x & \text{si } x > 2 \end{cases} \end{array}$$

Exercice 2

L'auberge dans laquelle vous avez prévu de passer la nuit ce soir propose des tarifs très intéressants, pour peu que l'on n'arrive pas trop tard. En effet, plus on arrive tôt moins on devra payer. Vous devez construire une fonction `prix_auberge` vous donnant directement le prix à payer en fonction de votre heure d'arrivée (un entier entre 7 et 24 car l'auberge ferme passé minuit). Le prix de base est de 10 euros plus 5 euros pour toute heure après midi. Il ne peut cependant pas dépasser 53 euros. Votre fonction doit renvoyer un entier qui est le prix à payer (en euros) correspondant à l'heure d'arrivée donnée.

Exercice 3

1. Écrire une fonction `rad_vers_deg` qui convertit en degrés, minutes et secondes un angle passé en argument en radians. Par exemple,

Python

```
In [1]: rad_vers_deg(4.7)
Out[1]: (269, 17, 24.589361352882406)
```

2. Écrire la réciproque `deg_vers_rad` qui prend en arguments un nombre entier de degrés et de minutes, ainsi qu'un nombre flottant de secondes et qui renvoie cet angle exprimé en radians.

On vérifie que :

Python
```
In [2]: rad_vers_deg(
            deg_vers_rad(269, 17, 24.589361352882406))
Out[2]: (269, 17, 24.589361352882406)
```

En revanche, `deg_vers_rad(rad_vers_deg(4.7))` pose problème.

3. Après avoir compris pourquoi, modifier la fonction en une fonction de nom `deg_vers_rad_bis` de telle manière à ce que l'appel

   `deg_vers_rad_bis(rad_vers_deg(4.7))`

   s'évalue bien à `4.7`.

Exercice 4

Écrire une fonction `tri(a, b, c)` qui prend en arguments trois nombres `a`, `b` et `c` et qui renvoie un tuple formé de ces trois valeurs dans un ordre croissant.

Exercice 5

Écrire une fonction `racine(a, b, c)` renvoyant le nombre de solutions réelles distinctes de l'équation $ax^2 + bx + c = 0$, avec la convention -1 pour une infinité de racines. On ignore les éventuels problèmes posés par la représentation des réels par des flottants.

Exercice 6

Le RMS Titanic était un paquebot transatlantique britannique, tragiquement célèbre pour son naufrage. Bien qu'il soit qualifié d'insubmersible, le paquebot fait naufrage lors de son voyage inaugural reliant Southampton (GB) à New York (EU) et alors qu'il transportait environ 2200 personnes (les chiffres exacts sont aujourd'hui encore inconnus, notamment à cause des embarcations clandestines). Le navire sombre totalement dans l'Atlantique le 15 avril 1912 à 2h20. La mauvaise gestion de l'évacuation (comme les canots qui sont affalés alors qu'ils ne sont pas remplis) et les eaux très froides de l'Atlantique Nord causeront la mort de 1490 personnes pour 771 survivants.

Le but de cet exercice consiste à créer un premier [1] modèle simpliste de prédiction de la chance de survie d'un passager du Titanic suite à son naufrage. On va écrire une fonction `chance_survie` qui reçoit cinq arguments :

1. le genre du passager : 0 pour homme et 1 pour femme ;
2. l'âge du passager : un entier positif ;
3. la classe du voyage du passager : 1, 2 ou 3 ;
4. le prix du ticket du voyage en £ : un flottant ;
5. le numéro de cabine : un entier entre 0 et 762.

On considère qu'un passager a plus de chances de survivre si :

- c'est une femme ;
- il est mineur (< 18 ans) ou sénior (≥ 60 ans) ;
- il voyage en 1re ou 2e classe ;
- le prix du ticket est au moins de $10,35$ £ (inclus) ;
- son numéro de cabine est pair et multiple de sept.

On veut déterminer à quel point un passage correspond aux cinq critères. La fonction doit tester tous les critères. S'ils sont tous vérifiés, elle doit renvoyer la chaîne de caractères "Très probable". Si seulement trois ou quatre critères sont vérifiés, elle doit renvoyer "Probable". Si aucun critère n'est vérifié elle doit renvoyer "Quasiment impossible" et dans les autres cas, elle doit renvoyer "Peu probable".

1. Si vous avez terminé le TP, vous pouvez essayer de construire un meilleur modèle et participer à la compétition *Titanic : Machine Learning from Disaster* (https://www.kaggle.com/c/titanic).