# Programmation Impérative en Python

**Examen L1-I/MI**

*Université Nice Sophia-Antipolis, Décembre 2016*

*LE FORMULAIRE EN DERNIERE PAGE EST LE SEUL DOCUMENT AUTORISE. Tâchez de soigner l'**indentation**, la limpidité et l'exactitude du code. Répondez aux questions sur la copie d'examen qui vous est fournie, ne joignez aucune autre feuille ni intercalaire. N'hésitez pas à utiliser le résultat d'une question précédente même si vous ne l'avez pas faite. Et ne perdez pas de temps...*

## Q1. Compréhension de calcul [3 pts]

**a)** Quels sont les résultats de `foo(2,1)` et `foo(3,4)` avec la définition suivante :

```
1 def foo(n,p) :                                    # on suppose n et p entiers ≥ 0
2     if n == 0 : return 0
3     return n + p + foo(n-1,p+1)
```

**b)** Avec `n` et `p` entiers $\geq 0$ *quelconques*, et en observant par exemple les résultats de la question précédente, pouvez-vous **deviner** une formule générale pour le résultat de `foo(n,p)` en fonction de `n` et `p` ? On ne demande aucune justification, il s'agit seulement de deviner la formule.

**c)** Procédez en Python à une vingtaine de `tirages aléatoires` de `n` et `p` dans [0,100] pour vérifier si votre formule en réponse à la question **b)** est raisonnable. Coupez le calcul dès qu'un contre-exemple est trouvé ! Si vous n'avez pas fait **b)**, notez `FORMULE(n,p)` la formule censée être calculée par `foo(n,p)`.

## Q2. Nombres premiers [3 pts]

Un nombre entier est **premier** s'il est $\geq 2$ et s'il n'est multiple d'aucun entier mis à part 1 et lui-même. On considère la définition ci-dessous d'une fonction programmée pour tester si un entier `n` est premier :

```
4 def est_premier(n) :                                  # on suppose n entier
5     return (n >= 2) and (sum(d for d in range(?,?) if ? % ? == 0) == ?)
```

**a)** Ecrivez cette définition sur votre copie après avoir remplacé les `?` pour qu'elle soit correcte.

**b)** En tant que programmeur, que pensez-vous de cette définition de `est_premier`, supposée correcte ? On ne vous demande pas de produire une autre version. Donnez simplement votre opinion, en la justifiant.

**c)** En utilisant la fonction `est_premier` ci-dessus, supposée correcte, écrivez quelques lignes de Python permettant d'*afficher* combien il y a de nombres premiers inférieurs à 10000 dont tous les chiffres sont distincts. *Indication : il est parfois intéressant de convertir un entier en chaîne de caractères...*

## Q3. Chaînes de caractères [4 pts]

*Les deux questions sont indépendantes.*

**a)** Ecrivez une fonction `babel(s)` prenant une chaîne de caractères `s` et retournant une chaîne identique à `s` mais dans laquelle on aura supprimé tous les chiffres. Exemple : `(babel 'THX-11-38x')` $\rightsquigarrow$ `'THX--x'`

**b)** Programmez une fonction `mybin(n)` prenant un entier $n > 0$ et retournant une chaîne de caractères représentant l'écriture **binaire** de `n`. Il va de soi qu'il est interdit d'utiliser la primitive `bin` de Python. Exemple : `mybin(13)` $\rightsquigarrow$ `'1101'`.

## Q4. Calculs en direct au toplevel [5 pts]

Un concessionnaire dispose d'une liste `STOCK` contenant plusieurs centaines de voitures. Une *voiture* sera représentée par un *tuple* à deux éléments `('modèle',année)`. Une telle liste pourrait être (bien entendu ceci n'est qu'un exemple, sur lequel vous ne vous appuierez pas dans vos réponses) :

```
STOCK = [('208',2016),('Cayenne',2012),('DS',1972),('307',2012),('Clio',2013),...] # etc
```

Le patron interroge au téléphone son informaticienne Eva Luhatrice. Eva connaît bien Python et tape en général UNE SEULE ligne au toplevel pour lui répondre. Montrez ce qu'elle a demandé au toplevel (en 1 ligne pour les questions 1 à 4) afin d'obtenir la réponse à chacune des questions suivantes :

1. **Eva, pouvez-vous définir la liste `MODELES` de tous les modèles et uniquement des modèles s'il vous plaît.** *On espère que* `MODELES` *ait pour valeur* `['208', 'Cayenne', 'DS',...]`.
2. **J'ai un client qui veut une 307. Nous en avons une ?**
3. **Ah bien, et la première disponible est de quelle année ?**
4. **Combien avons-nous de 307 disponibles ?**
5. **Donnez-moi pour un collectionneur une voiture la plus ancienne possible. Tapez plusieurs lignes si besoin, je vais faire une sieste.**

## Q5. Une liste au hasard [2 pts]

**a)** Définissez une liste `L` de 30 entiers **distincts** et dans le **désordre** tirés au hasard dans [0,100].

**b)** Faites afficher la liste croissante des 10 plus grands éléments de `L`, *sans mutation sur* `L`.

## Q6. Insertion dans une liste [2 pts]

Programmez la fonction `ins_apres(x,y,L)` retournant une **copie** de la liste `L` obtenue en insérant `x` **après la première** apparition de `y` dans `L`. Cette fonction ne fera rien si `y` $\notin$ `L`.
*Une stratégie possible consiste à se positionner directement sur* `y` *s'il existe*. Exemple :

`ins_apres(10,8,[5,8,7,2,6,8])` $\rightsquigarrow$ `[5,8,10,7,2,6,8]`

## Q7. Travail sur un fichier [4 pts]

On dispose d'un fichier `jeux.txt` contenant plusieurs lignes. Sur chaque ligne se trouvent des nombres entiers, en quantité variable sur chaque ligne. Ecrivez quelques lignes de Python permettant de savoir quel est le **plus grand nombre** contenu dans le fichier, et à quel **numéro de ligne** il se trouve. On numérotera les lignes à partir de 1. Exemple d'exécution :

```
Le plus grand nombre est 4092 en ligne 8
```