Université d'Ottawa · University of Ottawa

Faculté de Génie – EECS

CSI2520 : PARADIGMES DE PROGRAMMATION

Hiver 2014 – Tutorat 9

Exercice I

1. Soit la liste suivante :

list = ['a', 'b', 'c', 'd']

Quel est le résultat des instructions suivantes:

```
print list[1:-1]
list[0:2] = 'z'
print list
```

['b','c']
['z','c','d']

2. Soit la liste suivante:

list = ['larry', 'curly', 'moe']

Quel est le résultat des instructions suivantes:

```
list.append('shemp')
list.insert(0, 'xxx')
list.extend(['yyy', 'zzz'])
print list
print list.index('curly')

list.remove('curly')
list.pop(1)
print list
```

['xxx','larry','curly','moe','yyy','zzz']
2

['xxx', 'moe', 'shemp', 'yyy', 'zzz']

3. Soit la liste suivante:
nums = [2, 8, 1, 6]

list = [i for i in nums if i<=2]

créer une liste small qui contient les éléments de la liste nums qui sont <=2

4. Sélectionner de la liste suivante, les fruits qui contiennent la lettre a, et mettre le nom du fruit en majuscule.

fruits = ['apple', 'cherry', 'bannana', 'lemon']

map(lambda x: x.capitalize() if 'a' in x else x, fruits)

5. Créer une liste avec les éléments de la liste suivante en majuscule, et ajouter !!! à la fin de chaque élément.

strs = ['hello', 'and', 'goodbye'] map(lambda x: x+"!!!", strs)

Créer une liste square qui continent le carré de chaque élément de la liste suivante :

nums = [1, 2, 3, 4] squares = map(lambda x: x**2, nums)

Exercice II: Listes à plusieurs dimensions

Ecrire un programme qui prend deux chiffres, X,Y comme entrée et génère un tableau bidimensionnelle dont chaque entrée est le produit de l'index de la colonne et l'index de la ligne.

Note: i=0,1.., X-1; j=0,1,.., Y-1.

Exemple

Si l'entrée du programme est:
3,5

Alors la sortie doit être:
[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]

Note: On suppose que l'entrée est fournie par la console sous format de deux chiffres séparés par une virgule.

see lab.py

Exercice III : Dictionnaires

Soit le dictionnaire suivant :

```
dict = {}
dict['a'] = 'alpha'
dict['g'] = 'gamma'
dict['o'] = 'omega'
```

Quel est le résultat des instructions suivantes :

```
print dict
```
{'a': 'alpha', 'o': 'omega', 'g': 'gamma'}

```
print dict['a']
```
alpha

```
dict['a'] = 6
```

```
'a' in dict
```
True

```
print dict['z']
```
raises KeyError

```
if 'z' in dict: print dict['z']
print dict.get('z')
```
None

Par défaut, lorsqu'on itère sur un dictionnaire, on itère sur ses clés. Notez que les clés sont dans un ordre aléatoire.

```
for key in dict: print key
```
a \n o \n g \n

```
for key in dict.keys(): print key
```
a \n o \n g \n

Liste des clés du dictionnaire

```
print dict.keys()
```
['a', 'o', 'g']

Similairement, il y a la liste des valeurs

```
print dict.values()
```
[6, 'omega', 'gamma']

Cas commun – itérer sur la liste ordonnée des clés pour accéder à chaque pair clé/valeur

.items() est le dictionnaire comme les pairs (clé, valeue)

print dict.items() [('a', 6), ('o', 'omega'), ('g', 'gamma')]

La syntaxe ci-dessous itère sur tous les elements du dictionnaire pour accéder aux
pairs (clé, valeur)

for k, v in dict.items(): print k, '>', v

a > 6
o > omega
g > gamma

Exercice IV: Fichiers

1. Imprimer "Python is a great language.\nYeah its great!!\n" dans un fichier foo.txt

2. Lire du fichier 10 caractères et les imprimer sur l'écran

Ceci produira le résultat suivant:
Read String is: Python is

3. Vérifier la position courante dans le fichier

4. Repositionner le pointeur au début

Ce programme produira le résultat suivant:
Read String is : Python is
Current file position : 10
Again read String is : Python is

5. Produire une liste de mots lus dans un fichier.