Cours SI6

Python – Les bases par l'exemple

python

Introduction

- Langage de programmation
  - Objet
  - Interprété
- Créé en 1994
- Dispo sur la majorité des plateformes

Caractéristiques

- Langage interprété
- Pas de déclaration de types
- Pas de déclaration de variables
- Gestion auto de la mémoire
- Programmation orienté objet, procédural et fonctionnel
- possibilité de générer du byte-code
- interactions standards (appels systèmes, protocoles, etc.)
- Blocs instructions suivant indentation

Utilisation

- Mode interactif dans un shell

```
$ python
Python 2.7.2 (v2.7.2:8527427914a2, Jun 11 2011, 15:22:34)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print 'hello world!'
hello world!
>>> ^D
$
```

- Mode programme

```
$ cat hello.py
#! /usr/bin/env python
print 'hello world!'
$ python hello.py
hello world!
```

- Commentaires précédés de #
- Affectations =
- Affectations parallèles
- Concaténation avec +
- Convertion:
  - int(var)
  - str(var)
  - float(var)

Bases

Affichage console

- Affichage console avec print:
  - En concaténant avec +
    - **>>> print** “Bonjour ”+nom+” “ +age+” ans"
  - En séparant les variables par des virgules
    - **>>> print** “Bonjour ”,nom,” “,age,” ans"
  - En substituant %s, %d, %f…:
    - **>>> print** “Bonjour: *%s %d* ans" % (nom, age)
- Attention: pour avoir l’accentuation de la langue française sous windows:
  - # -*- coding: iso8859_1 -*-
  - En tout début de fichier

Saisie depuis console

- Avec la fonction input:
  - varTxt = input (‘Message :\n’)
- Récupération sous forme de string
- Pour récupérer un mdp:
  - import getpass
  - mdp = getpass.getpass(‘Msg’)
  - Saisie non cachée env Windows!

- Demander à l'utilisateur:
  - 2 nombres
  - Lui donner la somme
  - Lui donner le produit

Exo01 : Addition

Listes

- Ensembles ordonnés et dynamiques d'éléments
  - **>>>** mylist = []
  - **>>>** mylist2 = list()
  - **>>>** x = True
  - **>>>** foo = ['bar', 12345, x]
  - **>>>** foo
    - ['bar', 12345, True]
- Ensemble indexé
  - **>>>** foo[2]
    - True
  - **>>>** foo[1:]
    - [12345, True]

- **>>>** foo[2] = 1
- **>>>** foo
  - ['bar', 12345, 1]
- **>>>** foo.append('new')
- **>>>** foo
  - ['bar', 12345, 1, 'new']
- **>>>** foo.insert(2, 'new')
- **>>>** foo
  - ['bar', 12345, 'new', 1, 'new']
- **>>>** foo.extend([67, 89])
- **>>>** foo
  - ['bar', 12345, 'new', 1, 'new', 67, 89]

Modifications éléments de liste

- **>>>** foo.index('new')
  - 2
- **>>>** foo.index(34)
  - Traceback (most recent call last): File "<stdin>", line 1, in ? ValueError: list.index(x): x not in list
- **>>>** 34 **in** foo
  - False

Recherche d'élément

- **>>>** bar = [0, 1] + [1, 0]
- **>>>** bar
  - [0, 1, 1, 0]
- **>>>** bar += [2, 3]
- **>>>** bar
  - [0, 1, 1, 0, 2, 3]
- **>>>** [0, 1] * 3
  - [0, 1, 0, 1, 0, 1]

Fusion de listes

- **>>>** matrix = [[1, 2, 3]] * 3
- **>>>** matrix
  - [[1, 2, 3], [1, 2, 3], [1, 2, 3]]
- **>>>** matrix[1][1] = 4
- **>>>** matrix
  - [[1, 4, 3], [1, 4, 3], [1, 4, 3]]

Attention duplication référence

- **>>>** foo.remove('new')
- **>>>** foo
  - ['bar', 12345, 1, 'new', 67, 89]
- **>>>** foo.remove(34)
  - Traceback (most recent call last): File "<stdin>", line 1, in ? ValueError: list.remove(x): x not in list
- **>>> del** foo[1:3]
- **>>>** foo
  - ['bar', 'new', 67, 89]

Suppression dans une liste

Exo02 : Liste et random

- Valeurs aléatoires comprises entre 0 et 10
- Mettre 3 entiers aléatoires dans une liste.
- Tirer une 4ème valeur
- Cette valeur est-elle dans la liste:
  - True
  - False
- import random
- random.random() renvoie un réel compris entre 0 et 1

- **>>>** x = 'hello world!‘
- **>>>** x[4]
  - 'o‘
- **>>>** x[2:4]
  - 'll‘
- **>>>** x[3:]
  - 'lo world!'
- **>>>** x[:]
  - 'hello world!‘
- **>>>** x[-3:]
  - 'ld!'
- **>>>** x[1:-1]
  - 'ello world'

Chaînes de caractère : séquence

- **>>>** ' ; '.join(['a', 'b', 'c'])
  - 'a ; b ; c`
- **>>>** 'hello crazy world!'.split(" ")
  - ['hello', 'crazy', 'world!']
- **>>>** 'hello crazy world!'.split(" ", 1)
  - ['hello', 'crazy world!']
- **>>>** 'hello crazy world!'.replace("o", "u")
  - 'hellu crazy wurld!'

Chaînes join/split/replace

Exo03:Csv

- Demander à l'utilisateur d'entrer une chaine contenant les informations des étudiants:
  - nom1 prenom1 classe1 nom2 prenom2…
- Indications:
  - Chaque champ est séparé par un espace
  - Chaque occurrence est séparée 2 espaces
- Manipuler la chaîne afin d'avoir:
  - Un tableau contenant 1 occurrence par case
  - Une occurrence est une chaine ou les champs sont séparés par un « ; »
- Afficher le tableau

Dictionnaires

- **>>>** dict1 = {}
- **>>>** dict2 = {'foo': 456, 123: 'bar'}
- **>>>** dict3 = dict() #clefs -> chaines
- **>>>** dict4 = dict(foo=456, bar=123)
- **>>>** dict1['foo'] = 456
- **>>>** dict1[123] = 'bar`
- **>>>** dict1[123]
  - 'bar`
- **>>>** dict1
  - {123: 'bar', 'foo': 456}
- **>>>** dict4
  - {'foo': 456, 'bar': 123}

- **>>>** dict1.keys()
  - [123, 'foo']
- **>>>** 123 **in** dict1
  - True
- **>>>** dict1.values()
  - ['bar', 456]
- **>>>** dict1.items()
  - [(123, 'bar'), ('foo', 456)]

Méthodes des dictionnaires

- **>>>** dict1[123] = 789
- **>>>** dict1
  - {123: 789, 'foo': 456}
- **>>> del** dict1['foo']
- **>>>** dict1
  - {123: 789}

Dictionnaires aff/supp

- **>>>** flames = {'windows': 'bof', 'unix': 'cool'}
- **>>> for** key **in** flames.keys():
- **... print** key, 'is', flames[key]
- **...**
  - windows is bof unix is cool
- **>>> for** key, value **in** flames.items():
- **... print** key, 'is', value
- **...**
  - windows is bof unix is cool

Parcourir dictionnaire

- Créer un dico pour stocker des numéros de téléphone:
  - Clef : le nom du contact
  - Valeur : son numéro
- Entrer 3 ou 4 contacts
- Demander à l'utilisateur d'entrer le nom du contact pour lequel il veut le numéro
- Afficher le numéro du contact

Exo04 : répertoire téléphone

Structuration/indentation

- Structuration définie par l'indentation
  - Une 1ère ligne pour l'entête suivie de :
  - Puis le bloc indenté
- Possibilité de tout mettre sur la même ligne si bloc réduit. (Déconseillé)

Les tests

- Opérateurs booléens:
  - <, <=, >, >=, !=, ==, is, and, or, not
  - == pour l'équivalence de valeurs
  - is pour comparaison d'instance
- **>>>** l1 = [1, 2, 3]
- **>>>** l2 = [1, 2, 3]
- **>>>** l1 == l2
  - True
- **>>>** l1 **is** l2
  - False
- **>>>** l3 = l1
- **>>>** l3 **is** l1
  - True

- **if** x == 'hello':
  - **print** 'hello too!‘
- **elif** x == 'bonjour':
  - **print** 'bonjour aussi!‘
- **else**:
  - **print** 'moi pas comprendre'

If…elif…else

- oeufs = 4 **if** souhaite_une_tarte() **else** 0

Expression conditionnelle

Pas de switch

- Python objet donc utilisation du polymorphisme
- Sinon utilisation d'un dictionnaire:
  - traitements = {
    - 'en': traiter_anglais,
    - 'de': traiter_allemand,
    - 'fr': traiter_francais,
    - *# etc.* }
  - **def** traiter_commande(langue, commande):
    - traiter = traitements[langue]
    - traiter(commande)

```
for <var> in <sequence>:
    <instructions>
else:
    <instructions, sequence epuisee sans break>
```

- **>>> for** i **in** range(0,3,1):
- **… print** (i)
- **>>>for** e **in** lesEtudiants**:**
- **…print (** e **)**
- **>>>for** lettre **in** ‘Anthony’
- **…print (**lettre**)**

Boucle for

- Pour le parcours d'une liste
- Lorsque l'on veut l'indice de l'élément dans la liste

```
listeObjet = ['Anthony','est','fort','en','Python']
for i,e in enumerate(listeObjet):
    print ("Objet "+e+" en position:"+str(i))
```

```
Objet Anthony en position:0
Objet est en position:1
Objet fort en position:2
Objet en en position:3
Objet Python en position:4
```

Enumerate

- En entrée : 1 tableau + une chaine

```
data = ["medassi;anthony;sio1","naudier;nans;sio1"]
entete ="nom;prenom;classe"
```

- En sortie : 1 dictionnaire

```
{'prenom': 'anthony;nans', 'classe': 'sio1;sio1', 'nom': 'medassi;naudier'}
```

Exo05 : Transposition csv

```
while <condition>:
    <instructions>
else:
    <instructions, condition fausse>
```

- i=0
- while i<4:
- print (i)
- i+=1

Boucle while

- Faire deviner un nombre aléatoire à l'utilisateur
- Nombre compris entre 0 et 100
- Pénalité de 5 coups si l'utilisateur propose un nombre qu'il a déjà proposé
- Utilisateur guidé par l'affichage d'indice:
  - Trop petit
  - Trop grand
- Une fois le nombre trouvé, on affiche le nombre d'essais

Exo06:Nb mystère avec pénalité

Fin