# Examen de Langages de script n° 2 : 2011/2012

- Durée de l'examen : 2h
- Vous devez éteindre et ranger vos téléphones.
- Les programmes sont à faire en PYTHON 3.
- L'annexe du sujet contient
  - des rappels de PYTHON ;
  - la documentation du module `re`.

**Exercice 1 :**

1. Écrire un fonction `lterner(l,s)` qui reçoit deux listes de même longueur en argument, et renvoie une liste contenant les éléments des deux arguments en alternance, c'est-à-dire `lterner([1,2,3],[4,5,6])` renvoie `[1,4,2,5,3,6]`, par exemple.
2. Écrire une fonction `flatten(l)` qui reçoit une liste de listes `l` en argument, et renvoie la concaténation des listes dans l. Par exemple, `flatten([[2,3,4],[3,5],[1]])` renvoie `[2,3,4,3,5,1]`.
3. Écrire une fonction `diffsym(a,b)` qui reçoit deux ensembles `a,b` en argument, et renvoie leur *différence symétrique*, c'est-à-dire leur union sans leur intersection. Autrement dit, `x` est dans l'union symétrique de `a` et `b`, si $x$ est dans l'un des deux, mais pas dans les deux en même temps. Par exemple, `diffsym({1,2,4,7},{1,3,4})` renvoie `{2,3,7}`.
4. Qu'imprime le programme suivant ?

```
a=sorted(list({x//2 for x in range(10)}))
b=sorted(list({x//2 for x in range(10,0,-1)}))
c=sorted(list({x//2 for x in range(10,0,-2)}))
print(a)
print(b)
print(c)
```

5. Qu'imprime le programme suivant ?

```
print(len({(a,b,c) for a in range(3) for b in range(4) for c in range(5)}))
```

6. Qu'imprime le programme suivant ?

```
def p(l):
    if len(l) == 0:
        return [[]]
    r = p(l[1:])
    x = l[0]
    s = []
    for m in r:
        for j in range(len(l)):
            s.append(m[:j]+[x]+m[j:])
    return s
print(p([0,1,2]))
```

   Expliquez votre réponse.
7. Écrire une fonction `csum(n)` qui étant donné un entier positif retourne la somme de ses chiffres. Par exemple pour `1234`, la fonction renvoie `10`.

### Exercice 2 : Sécurité

Afin d'aid r l s utilisat urs à choisir d s mots d pass plus sécurisés, on s propos d programm r un script t stant la vulnérabilité du mot d pass .

Afin d contourn r l s attaqu s par forc brut , un mot d pass doit vérifi r c rtain s propriétés qu nous allons t st r.

Pour répondr à c t x rcic , vous d v z rédig r d'un part un modul python nommé `security.py` t d'autr part donn r l s xplications néc ssair s à la compréh nsion d votr cod .

Dans c modul vous pouv z ajout r tout l cod qu vous jug r z néc ssair (import d modul , fonctions auxiliair s,...).

C modul d vra cont nir l s fonctions suivant s qui pr nn nt n argum nt un mot d pass sous la form d'un chaîn d caractèr t r nvoi un boolé n `true` ou `f lse` si l mot d pass vérifi l t st décrit :

**mdp1 :** L mot d pass st ass z long (plus d 8 caractèr s)

**mdp2 :** L mot d pass conti nt au moins d ux caractèr s non-alphanumériqu s.

**mdp3 :** L mot d pass n conti nt pas d mot issu d'un d s l xiqu s suivants `c pit les.txt`, `fr nc is.txt`, `ngl is.txt`, `nnu ire.txt`. On suppos ra qu c s l xiqu s sont d s fichi rs t xt s cont nus dans l rép rtoir courant t dont chaqu lign conti nt un s ul mot. On justifi ra l s structur s d donné s choisi s n t rm d' fficacité.

**mdp4 :** L mot d pass n conti nt pas un variant d'un d s mots issu d s l xiqu s ci-d ssus. La variant étant obt nu n r mplaçant l caractèr « a » par « @ » ; t la chaîn d caractèr s « t » par « & ». On pourra fair app l à la fonction précéd nt .

**mdp5 :** L mot d pass n conti nt pas l miroir d'un mot issu d'un d s l xiqu s précéd mm nt cité (l miroir d « pass » st « ssap »).

L modul `security.py` s t rmin ra par la parti principal du cod qui n s ra xécuté qu lorsqu l modul st lancé n lign d command à partir d'un t rminal unix. C morc au d cod d mand ra à l'utilisat ur d' ntr r son mot d pass t imprim ra un not ntr 1 t 5 s lon l nombr d t sts passés.

### Exercice 3 : Mot de passe

C t x rcic propos d'écrir d s scripts pour gér r l s mots d pass .

Dans l systèm Unix, l s id ntifiants d s utilisat urs t l ur mot d pass sont sauv gardés dans l s fichi rs `/etc/p sswd` sous la form suivant :

```
...
utilis teur1:x:1000:1000:Nom Utilis teur1,,,:/home/utilis teur1:/bin/b sh
utilis teur2:x:1001:1001:Nom Utilis teur2,,,:/home/utilis teur2:/bin/b sh
...
```

t `/etc/sh dow` sous la form suivant :

```
...
utilis teur1:$1$XopOFYH9$IfxyQwBe9b8tiyIkt2P4F/:13262:0:99999:7:::
utilis teur2:$1$vXGZLVbS$ElyErNf/ gUDsm1DehJMS/:13261:0:99999:7:::
...
```

L s ntré s du fichi r `p sswd` sont séparé s par d s « : », ll s ont la signification suivant :

- id ntifiant d l'utilisat ur
- ntré d spécification du mot d pass
- id ntifiant numériqu d l'utilisat ur
- id ntifiant numériqu du group
- nom d l'utilisat ur ou champ d comm ntair
- rép rtoir p rsonn l d l'utilisat ur
- int rprét ur d command s, optionn l, d l'utilisat ur.

L s ntré s du fichi r `sh dow` ont la signification suivant :

- id ntifiant d l'utilisat ur
- mot d pass chiffré (l « 1 » du début indiqu l'utilisation d'un chiffr m nt MD5. L sign « * » indiqu qu l compt n p ut pas s conn ct r)
- l s autr s donné s n s ront pas utilisé s, ll s corr spond nt à un nombr d jours, à partir du 1 r janvi r 1970.

Dans la suit , on suppos ra l' xist nc d'un modul python `crypto.py` cont nant un fonction d cryptag `chiffre` t un fonction d décryptag `dechiffre` qu l'on pourra utilis r à condition d l s avoir importés.

1. Écrir un fonction `lecture` qui pr nd n argum nt l nom d'un fichi r t r nvoi la list d s s lign s. Écrir un fonction `ecriture` qui pr nd n ntré un list d lign s t l nom d'un fichi r t qui écrit l s lign s à la suit d s autr s dans l fichi r donné.
2. Afin d manipul r l s fichi rs `/etc/p sswd` t `/etc/sh dow/`, on propos d'utilis r un dictionnair ayant comm clé l'id ntifiant d l'utilisat ur t comm val ur un autr dictionnair associant aux cl fs suivant s :
   - `id_num` : id ntifiant numériqu
   - `nom` : nom d l'utilisat ur
   - `mdp` : mot d pass chiffré d l'utilisat ur

   Écrir un fonction p rm ttant d'initialis r c dictionnair . Écrir un fonction p rm ttant d sauv gard r c dictionnair dans l s fichi rs `/etc/p sswd` t `/etc/sh dow/`. C rtain s d s donné s n'étant pas stocké s dans l dictionnair , vous pouv z l s r mplac r par un val ur arbitrair .
3. Écrir un script `dduser` qui p rm t d cré r un nouv l utilisat ur n d mandant succ ssiv m nt
   - l'id ntifiant d l'utilisat ur,
   - l nom d l'utilisat ur,
   - l mot d pass d l'utilisat ur, t un confirmation d c mot d pass .

   L script d vra au fur t à m sur
   - vérifi r qu l'id ntifiant n' st pas déjà attribué,
   - vérifi r qu l s d ux mots d pass sont id ntiqu s,
   - vérifi r qu l mot d pass st un bon mot d pass ,
   - dét rmin r un id ntifiant numériqu qui n' st pas déjà attribué

   avant d' ntr r l s donné s du nouv l utilisat ur dans l dictionnair .
4. Écrir un script `p sswd` qui p rm t à un utilisat ur xistant d modifi r son mot d pass .
5. Décrir l s opérations à ff ctu r afin qu l s command s suivant s `./ dduser` t `./p sswd` lancé s à partir d'un t rminal lanc nt l s scripts corr spondants.

# A Annexe

## a) Rappel de quelques éléments de Python

– `r nge(i,j,l)` p rm t d parcourir l s nti rs d `i` à `j` xclu av c un pas d `l`.

```
>>> for i in r nge(3,-4,-1):
...     print(i)
...
3
2
1
0
-1
-2
-3
```

– L'opérat ur `//` n python calcul la parti ntièr d la division. Par x mpl , `5//2` donn `2`, t `3//4` a val ur `0`.

## b) Module `re` - descriptif basé sur le livre de Harold Erbin

**Syntaxe** L s r g x répond nt à un syntax très codifié t possèd nt d nombr ux symbol s ayant un s ns particuli r. Pour début r, tout caractèr alphanumériqu n'a pas d signification spécial : A corr spond simpl m nt à la l ttr A, 1 au chiffr 1, tc. Quant aux principaux symbol s spéciaux, il sont :

`.` : désign n'import qu l caractèr ;

`^` : indiqu qu l début d la chain doit corr spondr ;

`$` : indiqu qu la fin d la chain doit corr spondr ;

`{n}` : indiqu qu l caractèr précéd nt doit êtr répété n fois.

`{n,m}` : indiqu qu l caractèr précéd nt doit êtr répété ntr n t m fois.

`*` : l caractèr précéd nt p ut êtr répété aucun ou plusi urs fois. Par x mpl , à ab* p uv nt corr spondr : a, ab, ou a suivi d'un nombr qu lconqu d b.

`+` : l caractèr précéd nt p ut êtr répété un ou plusi urs fois. Par x mpl , à ab+ corr spond un a suivi d'un nombr qu lconqu d b.

`?` : l caractèr précéd nt p ut êtr répété zéro ou un fois. Par x mpl , à ab? corr spond nt ab t a.

L'antislash p rm t d'échapp r tous c s caractèr s spéciaux. L s croch ts [] p rm tt nt d'indiqu r un plag d caractèr , par x mpl [ -h] corr spondra à , f, g ou h. Final m nt, il r st qu lqu s caractèr s spéciaux ass z util s :

`\w` : il corr spond à tout caractèr alphanumériqu , c' st à dir qu'il st similair à [a-zA-Z0-9_] ;

`\W` : il corr spond à tout c qui n' st pas un caractèr alphanumériqu ;

`\b` : il corr spond à la frontièr (début ou fin) d'un mot ;

`\d` : il corr spond à tout caractèr numériqu , c' st à dir qu'il st similair à [0-9] ;

`\D` : il corr spond à tout c qui n' st pas un caractèr numériqu .

### Utilisation

`re.search(pattern, string)` Cherche le motif dans la chaîne passée en argument et retourne un `MatchObject` si des correspondances sont trouvées, sinon retourne `None`.

`re.split(pattern, string)` Découpe la chaine string selon les occurences du motif.

```
>>> re.split(r'\W', 'Truth is beautiful, without doubt.')
['Truth', 'is', 'beautiful', '', 'without', 'doubt', '']
```

`re.findall(pattern, string)` Retourne tous les sous-chaines de `string` correspondant au motif.

`re.sub(pattern, repl, string)` Retourne la chaîne `string` où le motif a été remplacé par `repl`.