Algorithmique et Programmation 1

TD2 - Corrigé

1 Les n-uplets (tuples)

2 Les listes

2.1 Lecture simplet et modification en place

2.1.1 Affichage du double des éléments d'une liste

```
for x in lst:
    print(2*x)
```

2.1.2 Doubler les éléments d'une liste

```
for i in range(len(lst)):
    lst[i] *= 2
```

2.1.3 Transformer les éléments d'une liste

```
for (i,x) in enumerate(lst):
    lst[i] = str(x)
```

En Python, il existe une syntaxe abbrégée permettant de transformer tous les éléments d'une liste. On peut la penser comme l'équivalent de la fonction `List.map` en OCaml. Cette syntaxe ne vous sera pas demandée pour les examens, mais elle est typique du langage Python.

```
l = [str(x) for x in l]
```

2.2 Algorithmes nécessitant une simple lecture

2.2.1 Recherche d'une valeur

```
res = False
for x in lst:
    if x == 3:
        res = True
if res:
    print("La liste lst contient la valeur 3.")
else:
    print("La liste lst ne contient pas la valeur 3.")
```

2.2.2 Détection du minimum

```
if lst == []:
      print("La liste est vide ; elle n'a donc pas d'element minimal.")
else:
      x_min = lst[0]
      for x in lst:
            if x < x_min:
                  x_min = x
      print("La valeur minimal de la liste est "+str(x_min))
```

2.2.3 Indice d'un élément

```
ind = -1
for (i,x) in enumerate(lst):
      if x == 3:
            ind = i
if indice < 0:
      print("La liste ne contient pas la valeur 3")
else:
      print("La derniere occurrence de 3 dans la liste est en position "+str(ind))
```

2.2.4 Test de croissance

```
res = True
for i in range(len(lst)-1):
      res = res and lst[i+1] >= lst[i]
if res:
      print("La liste est croissante")
else:
      print("La liste n'est pas croissante")
```

Autre solution qui s'arrête dès qu'elle rencontre une contradiction à la croissance.

```
i = 0
while i < len(l)-1 and lst[i+1] >= lst[i]:
      i += 1
if i >= len(l)-1:
      print("La liste est croissante.")
else:
      print("La liste n'est pas croissante.")
```

2.3 Algorithmes nécessitant de construire une nouvelle liste

2.3.1 Remplir une liste avec des copies d'un même élément

```
lst = []
for i in range(l):
      lst.append(5)
```

2.3.2 Remplir une liste avec des éléments fonctions de l'indice

```
lst = []
for i in range(l):
      lst.append(str(i+1))
```

2.3.3 Remplir une liste avec des valeur aléatoires

```
import random
lst = []
for i in range(l):
      lst.append(int(256*random.random()))
```

2.4 Algorithmes portant sur les listes de listes

2.4.1 Modification de l'image

```
ht = len(img)
lg = len(img[0])

for i in range(ht):
      for j in range(lg):
            img[i][j] = i+1
```

2.4.2 Création de l'image

```
img = []
for i in range(h):
      ligne = []
      for _ in range(l): # _ represente une variable anonyme
            ligne.append(i+1)
      img.append(ligne)
```