Rappel 1 (Liste 1).

Une *liste* est une suite d'éléments. Cela peut être une liste d'entiers, par exemple [5,-7,12,99], ou bien une liste de chaînes de caractères, par exemple ["Mars", "Avril ", "Mai"] ou bien les objets peuvent être de différents types [3.14," pi",10e-3,"x",True].

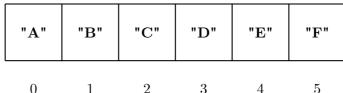
- Construction d'une liste. Une liste se définit par des éléments entre crochets :
 - liste1 = [5,4,3,2,1] une liste de 5 entiers,
 - liste 2 = ["Vendredi", "Samedi", "Dimanche"] une liste de 3 chaînes de caractères,
 - liste3 = [] la liste vide (très utile pour la compléter plus tard!).
- Accéder à un élément. Pour obtenir un élément de la liste, il suffit d'écrire liste [i] où i est le rang de l'élément souhaité.



Le piège c'est que l'on commence à compter à partir du rang 0!

Par exemple après l'instruction liste = ["A","B","C","D","E","F"] alors :

- liste [0] renvoie "A"
- liste [1] renvoie "B"
- liste [2] renvoie "C"
- liste [3] renvoie "D"
- liste [4] renvoie "E"
- liste [5] renvoie "F"



rang:

5

- Ajouter un élément. Pour ajouter un élément à la fin de la liste, il suffit d'utiliser la commande maliste.append(element) (to append signifie « ajouter »). Par exemple si premiers = [2,3,5,7] alors premiers.append(11) rajoute 11 à la liste, si ensuite on exécute la commande premiers.append(13) alors maintenant la liste premiers vaut [2,3,5,7,11,13].
- Exemple de construction. Voici comment construire la liste qui contient les premiers carrés :

```
liste carres = []
                           # On part d'une liste vide
for i in range (10):
    liste_carres.append(i**2)
                                # On ajoute un carré
```

À la fin liste carres vaut :

Rappel 2 (Liste 2).

- Longueur d'une liste. La longueur d'une liste est le nombre d'éléments qu'elle contient. La commande len (liste) renvoie la longueur (length en anglais). La liste [5,4,3,2,1] est de longueur 5, la liste ["Vendredi", "Samedi", "Dimanche"] de longueur 3, la liste vide [] de longueur 0.
- Parcourir une liste. Voici la façon la plus simple de parcourir une liste (et ici d'afficher chaque élément):

```
for element in liste:
    print (element)
```

• Parcourir une liste (bis). Parfois on a besoin de connaître le rang des éléments. Voici une autre façon de faire (qui affiche ici le rang et l'élément).

```
n = len(liste)
for i in range(n):
    print(i, liste[i])
```

• Pour obtenir une liste à partir de range() il faut écrire :

list (range(n))

Rappel 3 (Liste 3).

• Concaténer deux listes. Si on a deux listes, on peut les fusionner par l'opérateur + ». Par exemple avec liste = [4,5,6] et liste = [7,8,9]

liste1 + liste2 vaut
$$[4,5,6,7,8,9]$$
.

• Ajouter un élément à la fin. L'opérateur « + » fournit une autre méthode permettant d'ajouter un élément à une liste :

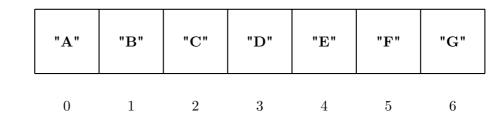
$$liste = liste + [element]$$

Par exemple [1,2,3,4] + [5] vaut [1,2,3,4,5]. Attention! Il faut entourer l'élément à ajouter de crochets. C'est une méthode alternative à liste .append(element).

• Ajouter un élément au début. Avec :

$${\rm liste} = [{\rm element}] + {\rm liste}$$
 on ajoute l'élément en début de liste. Par exemple $[5] + [1,2,3,4]$ vaut $[5,1,2,3,4]$.

• Trancher des listes. On peut extraire d'un seul coup toute une partie de la liste : liste [a:b] renvoie la sous-liste des éléments de rang a à b-1.



Par exemple si liste = ["A","B","C","D","E","F","G"] alors :

- liste [1:4] renvoie ["B", "C", "D"]
- liste [0:2] renvoie ["A", "B"]

rang:

— liste [4:7] renvoie ["E", "F", "G"]

Il faut encore une fois faire attention à ce que le rang d'une liste commence à 0 et que le tranchage liste [a:b] s'arrête au rang b-1.