Exercices Python Boucles

Python Cheat Sheet

1 Où suis-je ?

```
for i in range(34,190,2):
        for j in range(10):
                a=i*j
print(a)
```

1. Que vaut *a* à la première itération de boucle ?
2. Que vaut *a* à la fin du programme ?

Correction

À la première itération *a* vaut 0, à la dernière itération *a* vaut 188 x 9 = 1692

2 Tant qu'il y a de la vie, il y a de l'espoir

```
vie = 10
espoir = 0
while vie>0:
        while espoir < 5:
                espoir+=1
                print("tant qu'il y a la vie")
        print("on dit toujours y a espoir")
        vie-=1
print("Magic System")
```

1. Que vaut *espoir* à la fin du programme ?
2. Que vaut *vie* à la fin du programme
3. Combien de fois la phrase "Tant qu'il y a la vie" apparait elle avant que *vie == 9*
4. Combien de fois la phrase "on dit toujours y a espoir" apparait elle avant que *vie == 5*
5. Créer une autre boucle pour récrire les paroles des 6 premiers couplets de Magic in the air des Magic System (https://www.azlyrics.com/lyrics/magicsystem/magicintheair.html)

Correction

1. espoir vaut 5 à la fin du programme
2. vie vaut 0 à la fin du programme
3. La phrase "tant qu'il y a la vie" apparait 5 fois
4. La phrase "on dit toujours y a espoir" apparait 5 fois avant que vie == 5
5. En python :

```
for i in range(6):
        for j in range(2):
                if j==0:
                        print("Feel the magic in the air")
                else
```

```
                        print("Levez les mains en l'air")
                print("Allez, allez, allez")
```

3 Double boucle à indice variable

Je veux réaliser un programme qui me permet d'imbriquer deux boucles ensemble. Je veux que la seconde boucle démarre au même nombre que l'indice de la première boucle.

```
for i in range(10):
        for j in ##écrire ici la plage
                print(j)
```

La sortie de ce programme doit être :
0123456789123456789234567893456789456789567896789789899

Correction

```
for i in range(10):
        for j in range(i,10):
                print(j)
```

4 Boucler sur du texte

En python il est possible de boucler sur toute structure à plusieur éléments, un *range*, une *liste* ou du *texte*.

Compter le nombre de fois que la lettre **e** apparait dans cette phrase :

Sans honneur que précaire, sans liberté que provisoire, jusqu'à la découverte du crime ; sans situation qu'instable, comme pour le poète la veille fêté dans tous les salons, applaudi dans tous les théâtres de Londres, chassé le lendemain de tous les garnis sans pouvoir trouver un oreiller où reposer sa tête, tournant la meule comme Samson et disant comme lui : "Les deux sexes mourront chacun de son côté" ; exclus même, hors les jours de grande infortune où le plus grand nombre se rallie autour de la victime, comme les juifs autour de Dreyfus, de la sympathie – parfois de la société – de leurs semblables, auxquels ils donnent le dégoût de voir ce qu'ils sont, dépeint dans un miroir, qui ne les flattant plus, accuse toutes les tares qu'ils n'avaient pas voulu remarquer chez eux-mêmes et qui leur fait comprendre que ce qu'ils appelaient leur amour (et à quoi, en jouant sur le mot, ils avaient, par sens social, annexé tout ce que la poésie, la peinture, la musique, la chevalerie, l'ascétisme, ont pu ajouter à l'amour) découle non d'un idéal de beauté qu'ils ont élu, mais d'une maladie inguérissable ; comme les juifs encore (sauf quelques-uns qui ne veulent fréquenter que ceux de leur race, ont toujours à la bouche les mots rituels et les plaisanteries consacrées) se fuyant les uns les autres, recherchant ceux qui leur sont le plus opposés, qui ne veulent pas d'eux, pardonnant leurs rebuffades, s'enivrant de leurs complaisances ; mais aussi rassemblés à leurs pareils par l'ostracisme qui les frappe, l'opprobre où ils sont tombés, ayant fini par prendre, par une persécution semblable à celle d'Israël, les caractères physiques et moraux d'une race, parfois beaux, souvent affreux, trouvant (malgré toutes les moqueries dont celui qui, plus mêlé, mieux assimilé à la race adverse, est relativement, en apparence, le moins inverti, accable celui qui l'est demeuré davantage), une détente dans la fréquentation de leurs semblables, et même un appui dans leur existence, si bien que, tout en niant qu'ils soient une race (dont le nom est la plus grande injure), ceux qui parviennent à cacher qu'ils en sont, ils les démasquent volontiers, moins pour leur nuire, ce qu'ils ne détestent pas, que pour s'excuser, et allant chercher comme un médecin l'appendicite l'inversion jusque dans l'histoire, ayant plaisir à rappeler que Socrate était l'un d'eux, comme les Israélites disent de Jésus, sans songer qu'il n'y avait pas d'anormaux quand l'homosexualité était la norme, pas d'anti-chrétiens avant le Christ, que l'opprobre seul fait le crime, parce qu'il n'a laissé subsister que ceux qui étaient réfractaires à toute prédication, à tout exemple, à tout châtiment, en vertu d'une disposition innée tellement spéciale qu'elle répugne plus aux autres hommes (encore qu'elle puisse s'accompagner de hautes qualités morales) que de certains vices qui y contredisent comme le vol, la

cruauté, la mauvaise foi, mieux compris, donc plus excusés du commun des hommes ; formant une franc-maçonnerie bien plus étendue, plus efficace et moins soupçonnée que celle des loges, car elle repose sur une identité de goûts, de besoins, d'habitudes, de dangers, d'apprentissage, de savoir, de trafic, de glossaire, et dans laquelle les membres mêmes, qui souhaitent de ne pas se connaître, aussitôt se reconnaissent à des signes naturels ou de convention, involontaires ou voulus, qui signalent un de ses semblables au mendiant dans le grand seigneur à qui il ferme la portière de sa voiture, au père dans le fiancé de sa fille, à celui qui avait voulu se guérir, se confesser, qui avait à se défendre, dans le médecin, dans le prêtre, dans l'avocat qu'il est allé trouver; tous obligés à protéger leur secret, mais ayant leur part d'un secret des autres que le reste de l'humanité ne soupçonne pas et qui fait qu'à eux les romans d'aventure les plus invraisemblables semblent vrais, car dans cette vie romanesque, anachronique, l'ambassadeur est ami du forçat : le prince, avec une certaine liberté d'allures que donne l'éducation aristocratique et qu'un petit bourgeois tremblant n'aurait pas en sortant de chez la duchesse, s'en va conférer avec l'apache ; partie réprouvée de la collectivité humaine, mais partie importante, soupçonnée là où elle n'est pas, étalée, insolente, impunie là où elle n'est pas devinée; comptant des adhérents partout, dans le peuple, dans l'armée, dans le temple, au bagne, sur le trône; vivant enfin, du moins un grand nombre, dans l'intimité caressante et dangereuse avec les hommes de l'autre race, les provoquant, jouant avec eux à parler de son vice comme s'il n'était pas sien, jeu qui est rendu facile par l'aveuglement ou la fausseté des autres, jeu qui peut se prolonger des années jusqu'au jour du scandale où ces dompteurs sont dévorés ; jusque-là obligés de cacher leur vie, de détourner leurs regards d'où ils voudraient se fixer, de les fixer sur ce dont ils voudraient se détourner, de changer le genre de bien des adjectifs dans leur vocabulaire, contrainte sociale, légère auprès de la contrainte intérieure que leur vice, ou ce qu'on nomme improprement ainsi, leur impose non plus à l'égard des autres mais d'eux-mêmes, et de façon qu'à eux-mêmes il ne leur paraisse pas un vice.

--Proust

Correction

```
cpt=0
for i in phrase:
        if i==e:
                cpt+=1
print(cpt)
```

5 Les multiples

Écrire une fonction qui permet de trouver les nombres qui sont **à la fois** multiple de 5, multiple de 3 et divisible par 7. La fonction prendra en paramètre deux bornes (inférieur et supérieur) pour limiter la recherche entre ces bornes inclues.

1. Afficher ces valeurs
2. Faire la somme de ces valeurs et l'afficher
3. Garder en mémoire ces valeurs dans une liste

Correction

```
def multiple(borne_inf, borne_sup):
        liste_multiple=[]
        for i in range(borne_inf, borne_sup):
                if i%5 ==0:
                        if i%3==0:
                                if i%7==0:
                                        liste_multiple.append(i)
        return liste_multiple
```

Autre possibilité

```
def multiple(borne_inf, borne_sup):
        liste_multiple=[]
        for i in range(borne_inf, borne_sup):
                if i%5==0 and i%3==0 and i%7==0:
                        liste_multiple.append(i)
        return liste_multiple
```

6 La feuille de papier

Nous disposons d'une feuille de papier de 0.1mm en combien de fois doit on la plier pour que son épaisseur atteigne 400m ?

Correction

```
ep = 0.1 #mm
dist = 400000 #mm
cpt=0
while ep<dist:
        ep+=ep
        cpt+=1
print(f"il faut la plier {cpt} fois")
```

7 Heureux qui comme Ulysse

Statistiquement nous parcourons environ 12 200km par an, sachant que la circonférence de la terre est de 40 075 km, en combien d'année avons nous parcouru le tour de la terre ?

Compliquons légèrement les choses. Une voiture réalise 1000km par jour mais coute 100 euros par jour à faire avancer, un vélo réalise 100km par jour mais coute 10 euros à faire avancer, un avion réalise 6000km par jour mais coute 500 euros à faire avancer, à pied nous réalisons 20km par jour mais nous avons besoin de 10 euros pour avancer. Sur le modèle de la fonction ci-dessous, réaliser une fonction par modalité de voyage qui retourne la durée et le coût d'un tour du monde.

```
def voyagerPied(nombre_km):
        km_parcouru = 0 #les km parcourus
        temps_passe = 0 #le temps passé en jours
        argent_depense = 0 #argent dépensé en euros
        ## Modélisation sous forme de boucle
        ##
        ##
        return temps_passe,argent_depense
```

Quelle est la modalité la plus rapide ?
Quelle est la modalité la moins couteuse ?

Correction

```
def voyagePied(nombre_km):
        km_parcouru = 0 #les km parcourus
        temps_passe = 0 #le temps passé en jours
        argent_depense = 0 #argent dépensé en euros
        while km_parcouru < nombre_km:
                temps_passe+=1
```

```
                km_parcouru+=20
                argent_depense+=10
        return temps_passe,argent_depense

def voyagerVoiture(nombre_km):
        km_parcouru = 0 #les km parcourus
        temps_passe = 0 #le temps passé en jours
        argent_depense = 0 #argent dépensé en euros
        while km_parcouru < nombre_km:
                temps_passe+=1
                km_parcouru+=1000
                argent_depense+=100
        return temps_passe,argent_depense

def voyagerVelo(nombre_km):
        km_parcouru = 0 #les km parcourus
        temps_passe = 0 #le temps passé en jours
        argent_depense = 0 #argent dépensé en euros
        while km_parcouru < nombre_km:
                temps_passe+=1
                km_parcouru+=100
                argent_depense+=10
        return temps_passe,argent_depense

def voyagerAvion(nombre_km):
        km_parcouru = 0 #les km parcourus
        temps_passe = 0 #le temps passé en jours
        argent_depense = 0 #argent dépensé en euros
        while km_parcouru < nombre_km:
                km_parcouru+=6000
                temps_passe+=1
                argent_depense+=500
        return temps_passe,argent_depense


temps_pied,argent_pied = voyagerPied(40075)
temps_voiture,argent_voiture = voyagerVoiture(40075)
temps_avion,argent_avion = voyagerAvion(40075)
temps_velo,argent_velo = voyageVelo(40075)

print(f"Le temps le plus court est {min(temps_pied,temps_voiture,temps_avion,temps_velo)}")
print(f"Le moins couteux est {min(argent_pied,argent_voiture,argent_avion,argent_velo)}")
```

8 Break dance

L'instruction **break** permet d'arreter une boucle, alors que l'instruction **continue** permet de continuer le programme, comme si rien ne se passait.

```
texte = "Ceci est un texte, tout ce qu'il y a de plus simple"

for lettre in texte :
        print(lettre)
        if lettre == "x" :
                print("Un x a été trouvé !")
                break
```

Cette instruction permet d'arreter sur un cas particulier

```
liste = [ -4, 2, 6, 0, 1, 3, 0, 10]
for n in liste:
        if n==0:
                continue
        print(1/n)
```

Et dans ce cas, cette instruction permet de faire une exception sans s'arrêter d'executer.

En se basant sur la prmière structure écrire un programme pour trouver l'indice du premier 0 dans cette liste :

```
liste1=[1,4,6,7,4,0,1,2,3,5,0,6,7]
```

En se basant sur la seconde structure définir une autre liste de façon à ce que qu'un élément de la liste soit défini comme suivant lorsque c'est possible :

```
elt = (elt_suivant+1)/elt_suivant
```

Correction

```
for i in range(len(liste1)):
        if liste1[i]==0:
                print(i)
                break
```

```
liste2 =[]
for i in range(len(liste1)):
        if i+1>=len(liste1):
                continue
        elif liste1[i+1] == 0:
                continue
        else:
                liste2.append(liste1[i+2]/liste1[i+1])
```

On pouvait aussi écrire :

```
liste2 =[]
for i in range(len(liste1)):
        if i+1>=len(liste1) or liste1[i+1]==0:
                continue
        else:
                liste2.append(liste1[i+2]/liste1[i+1])
```