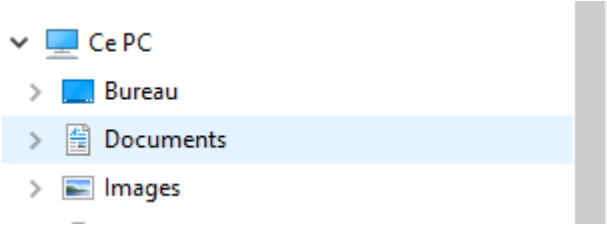
Fonctionnement de la séquence

- Le fichier que vous êtes en train de lire va vous permettre d’apprendre les bases de python à votre rythme.
- Cette séquence donnera lieu à plusieurs notes dont une évaluant votre capacité à respecter les consignes écrites et orales.
- Pour faciliter votre travail comme votre évaluation, vous allez commencer par ouvrir l’explorateur de fichier, vous placer dans votre répertoire document et créer un répertoire python. (si ça ne vous semble pas évident, n’ayez crainte dans la diapo suivante il y aura une explication.
- Durant la séquence vous stockerez vos programmes et vos fichiers Word contenant vos penses bêtes et vos proposition pour les évaluations.
- Il est vivement conseillé de faire une copie de ce répertoire dans votre clé USB au cas où. Vous êtes l’unique responsable de vos fichier et de l’organisation de votre compte sur les ordinateurs du lycée.

Tuto : Créer un répertoire/dossier

- Vous trouverez l’icone de l’explorateur de fichier sur la barre en bas de votre écran. Cliquer dessus.
- Vous trouverez le répertoire document à gauche

de votre écran. Cliquer dessus.



- Vous allez voir apparaitre le contenu actuel de votre répertoire document. Pour créer un répertoire, vous faites un click droit (bouton droit de la souris) et vous choisissez l’option « Nouveau » puis « Dossier »
- Vous verrez apparaitre un nouveau répertoire, avez un texte surligné en bleu disant « nouveau dossier » remplacez ce texte par « Python ».
- Si vous voulez changer le nom d’un répertoire , vous clickez une ou deux fois sur le nom actuel jusqu’à ce que celui-ci soit surligné en bleu, et là vous pouvez effectuer votre modification.

Tuto : faire une copie d’écran

- Vous allez devoir vous faire un pense bête où vous écrirez tout ce que vous avez appris durant la séance. Ce pense bête vous permet de réviser, mais aussi de retrouver rapidement l’information quand vous l’avez oubliée sans avoir à parcourir ce diaporama (qui est plutôt long)
- Vous pouvez noter l’information en toute lettre, mais vous pouvez aussi copier des morceaux d’écran et les coller dans votre fiche.
- Etape 1
- Chercher sur votre clavier la touche « imprim écran » et appuyer dessus. (le contenu complet de ce qui est présenté sur l’écran est maintenant en mémoire)
- Lancer le programme paint (on clique sur l’icone en bas à gauche de la page) et on sélectionne le programme paint (son icone est une palette de peinture)
- Copier le contenu de la mémoire dans paint (on appuie simultanément sur les touches « Ctrl » et « V »)
- On sélectionne la zone de l’image nous intéressant en traçant autour de celle-ci un cadre avec notre souris. Puis on copie cette zone encadrée en mémoire (on appui simultanément sur les touches « Ctrl » et « C »
- Dans le traitement de texte on choisira où on veut mettre notre image puis on la collera avec Ctrl+V

PRISE EN MAIN DE PYTHON

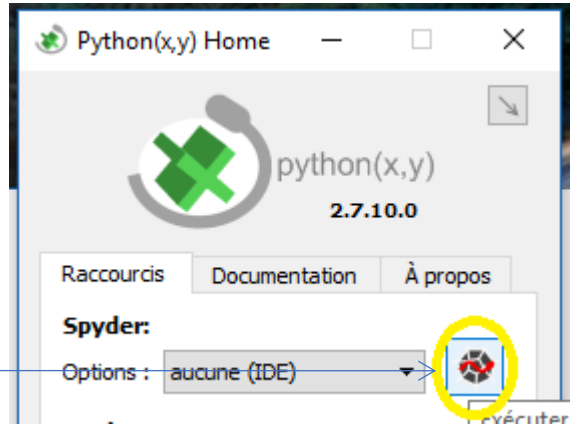
Python est un langage de programmation , pour l'utiliser il nous faut utiliser une des nombreux environnement de travail le contenant.

python (x,y)

Après avoir lancé l'application

Cliquer sur le carré entouré de

Jaune dans l'image à droite :

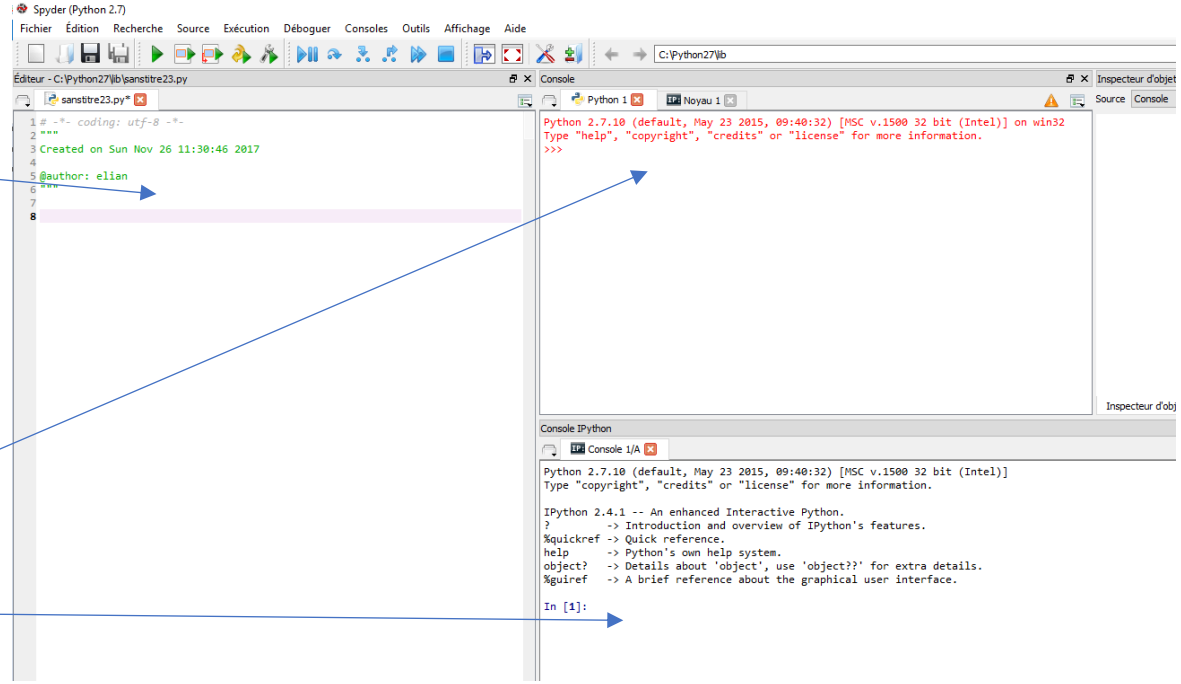


Sur LorDi

Lancer la suite de logiciel MCNL

Dans le menu choisir : Maths , puis à droite cliquer sur + sous Python et choisir Spyder (très lent) ou éduPython (un peu plus rapide)

Il apparaît alors un fenêtre séparée en plusieurs parties dont :



Une partie édition de programme

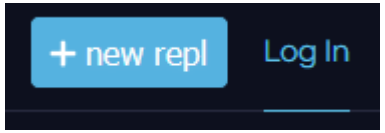
Cette partie permet d'éditer des programmes (à utiliser quand ils sont longs

Une partie console

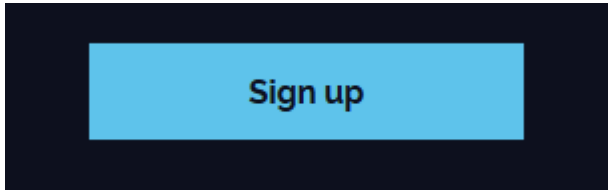
(il y en a deux ici) à utiliser pour effectuer des calculs, c'est ici que sera écrit les traitements de vos algorithmes (réponses de vos programmes)

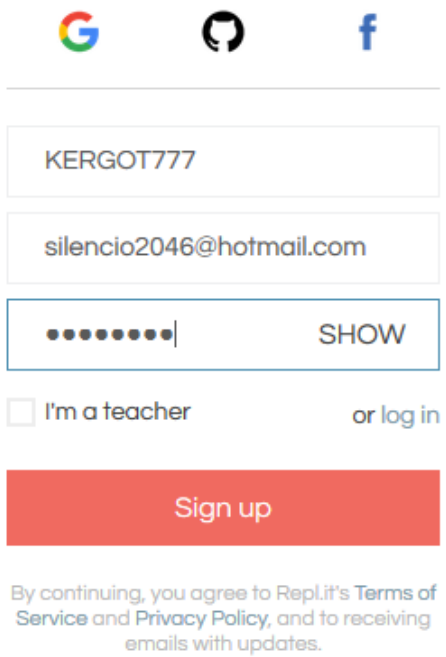
Utiliser Repl.it (1/3)

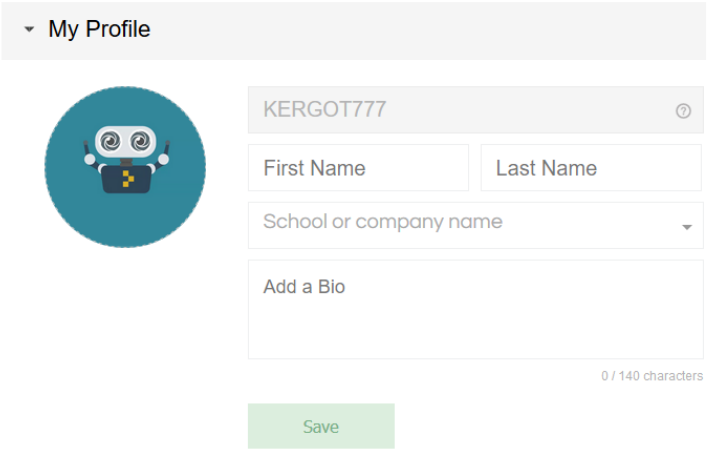
- À l'aide d'un navigateur internet, on se rend à l'adresse : repl.it
- Dans la partie supérieure de l'écran on a deux boutons :



- + new repl veut dire lancer une nouvelle session
- Log in : permet de se connecter (ce qui n'est possible que si on a un compte)
- Au milieu de l'écran il y a un bouton permettant de créer un compte :







Utiliser Repl.it (2/3)

Processus d'inscription :

Première page

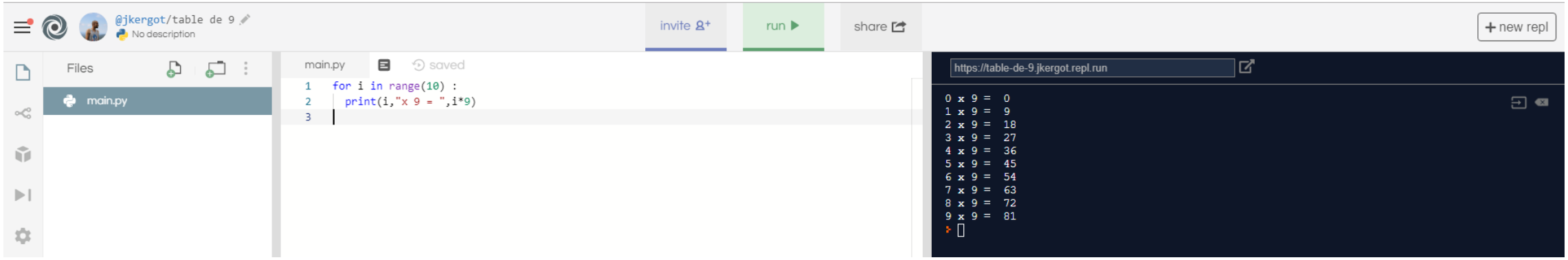
- Après avoir appuyé sur « Sign up »
- Pour username j'ai choisi KERGOT777 qui sera mon alias
- Sur le site, chaque utilisateur a un alias unique
- Email : mon adresse email
- Password : votre mot de passe (vous le noterez dans votre téléphone ou un autre endroit sûr, histoire de le retrouver en cas d'oubli.
- En tant qu'élève il ne faut pas cocher la case « I'm a teacher »

Seconde page (ou en cliquant sur votre username, puis sur (edit your profile)

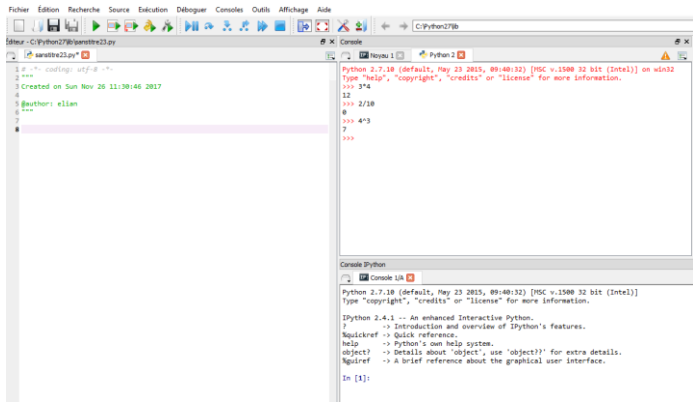
- Après vous devez donner votre Prénom (First Name) et votyrre Nom de famille (Last Name)
- Pour la rubrique School or company name, commencez par taper Lycee Albert Einstein et rapidement on vous proposera une liste d'établissement et vous pourrez choisir votre lycée dans la liste.

Utiliser Repl.it (3/3)

- Après avoir choisi l'option : « + new repl », choisi comme langage Python, et choisi le nom du projet, on obtient un écran comme celui de la capture suivante :



- On peut y voir 3 zones :
  - Colonne de droite : la console. C'est un environnement où l'on peut tester nos commandes python, et on peut aussi étudier les conséquences du lancement d'un programme.
  - Colonne centrale : l'éditeur de programme, on y écrit des lignes de codes qui seront compilées et lancées quand on appuiera sur le bouton vert « Run »
  - Colonne de gauche, la gestion de l'environnement, on y trouve main.py le programme principal, mais on peut ajouter des fichiers supplémentaires, c'est de là qu'on pourra commander l'intégration de librairies supplémentaires dans le noyau python (ce qui nous donnera le droit de les appeler dans notre programme)
- Notre travail s'enregistre en temps réel, et si l'on travaille à partir de notre compte on pourra toujours retrouvé ce que l'on a programmé en cherchant « My Repls » dans le menu (on accède à celui-ci en cliquant sur l'icone en haut à gauche présentant trois lignes horizontales superposées).



I : LES NOMBRES

Dans la console on peut effectuer toute sorte de calculs.

Effectuez les calculs suivants : 3x4 , 2/10, 2.7/10, 4^3

Vérifiez les résultats affichés par la console.

Rappel : sur la calculatrice pour calculer 4^3 , on Tape 4^3

Que pouvez vous observer ?

$$\frac{2}{10}$$

Pour la division :

Python n’est pas un logiciel de mathématiques,

Dans **sa version 2**, lorsqu’on manipule des nombres entiers, il retournera des nombres entiers.

Pour que Python prenne un entier pour un réel, il suffit de rajouter un point à la suite du nombre :

2 est un entier alors que 2.0 ou 2. est considéré comme un réel.

Pour éviter cela il faut taper 2/10. (mettre un point à la fin de 2/10)

ou taper float(2)/float(10) (plus lourd)

Remarque : dans la version 3 la question ne se pose même pas

Pour la puissance:

On remarque qu’il a ajouté 4 et 3, pour éviter cela il suffit de taper

4**3

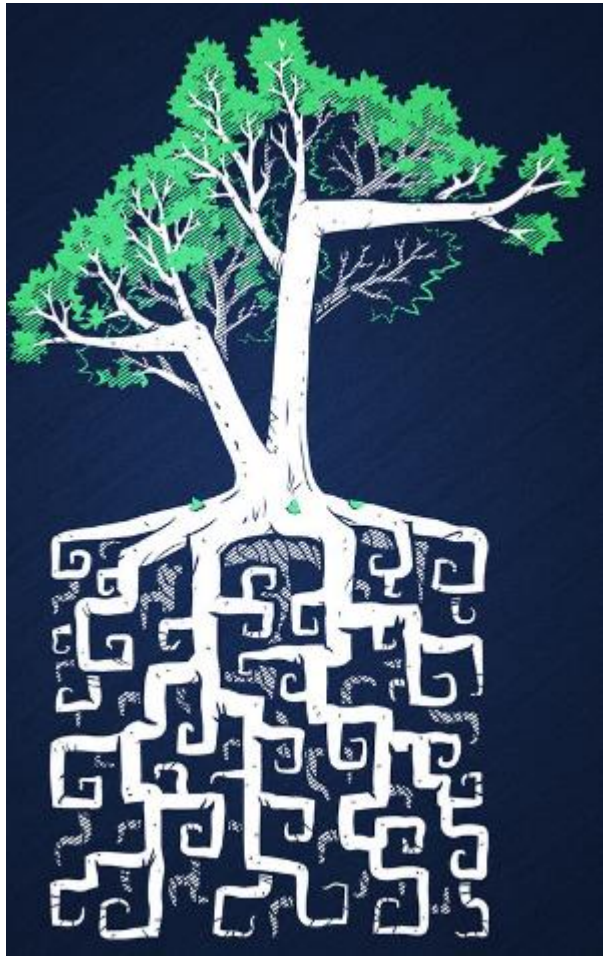
Pour faire un calcul utilisant une racine on peut se rappeler qu'en anglais racine carrée se dit « square root », deux mots qui seront abrégés sur Géogébra comme dans la plus part des langages de programmation par sqrt.

En python la racine carrée sera aussi effectuée avec :

sqrt(nombre dont on veut la racine)

Tentez d'effectuer $\sqrt{3}$

Que se passe-t-il ?



La console nous a affiché le message suivant :

```
>>> sqrt(3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'sqrt' is not defined
>>> |
```

Autrement dit, il ne reconnait pas la fonction sqrt().

Contradiction ?

Non, en fait la fonction sqrt() n'étant utilisée que dans des contexte assez rares, elle existe mais n'est pas accessible de manière directe, elle est rangée dans une bibliothéque « math » qu'il faudra activer.

```
>>> from math import sqrt
>>> sqrt(3)
1.7320508075688772
```

L’utilisation de librairies permet d’enrichir le langages Python d’une foule de fonctions plus ou moins utiles (gestions de la carte son, des différents cœurs du processeur, dessin 3D, animations, …)

Charger toutes ces fonctions alourdirait les programmes :ça les rend lents et gourmands en ressources (mémoire vive et espace disque) c’est pour cela qu’on préfère les ranger dans des groupes de fonctions appelées librairies et quand on en a besoin on peut charger en mémoire (rendre utilisable) toutes les fonctions ou on peut comme dans l’exemple précédent charger une seule fonction de la librairie.

Astuce : pour charger toutes les fonctions on peut écrire :

from math import * (* veut dire tout)

Les librairies les plus utilisées sont

Math : pour ce qui est du calcul

Numpy : pour ce qui est des graphiques, listes…

Matplotlib et pylab

Random : pour tout ce qui concerne les probabilités

PIL: traitement des images

Joblib : calcul parallèle

Sympy : pour ce qui est du calcul formel (équations,…)

Variables

En programmation on est amené à enregistrer le résultat de nos calculs ou des valeurs importante dans des variables.

Pour stocker la valeur 5 dans la variable « nombre » il suffit d’écrire :

nombre = 5

Si on tape nombre*3 l’ordinateur nous répondra : 15

Pour stocker le résultat de l’opération 7-9x3 dans la variable « number » :

number = 7-9x3

Variables

- Prévoir ce que python nous écrira à la fin de chaque ligne.

```
A = 17
B=2*A
A = A-5
print(A)
print(B)
```

Conclusion : l’assignation est statique (tant qu’on ne demande pas de manière explicite le changement de la valeur d’une variable celle-ci restera égale à elle même

Exercices

1/Expliquer en détails ce qu'il se passe ci-contre

```
In [47]: p=10

In [48]: q=10.

In [49]: p==q
Out[49]: True

In [50]: p is q
Out[50]: False

In [51]: type(p)
Out[51]: int

In [52]: type(q)
Out[52]: float
```

2/Effectuer sur la console les calculs suivants de telle sorte que les résultats soient donnés sous forme décimale

calcul 1 : $\frac{\sqrt{3}}{4}$ calcul 2:$\frac{135}{124}$ calcul 3:$\left(\frac{5}{3}\right)^3$

II Les TEXTES

Si on veut tape bonjour dans la console, celle-ci croit qu'on fait référence à une variable s'appelant bonjour et donc affichera un message d'erreur si une telle variable n'a pas été définie au préalable.

```
Python 2.7.10 (default, May 23 2015, 09:40:32) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> bonjour
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'bonjour' is not defined
```

Pour éviter le message d'erreur, il faut savoir qu'un texte, autrement dit une chaine de caractère se met toujours entre guillemets.

Donc il faut taper ''bonjour'',

L'opérateur + appliqué à des chaines de caractères les concatène (les colle à le suite l'une de l'autre)

Questions :

1) avec cette logique devinez ce que Python vous affichera si vous écrivez ''Bonjour''*2

2) Prévoir le retour à la fin de la séquence suivante :

mot = '' les gens''

''bonjour ''*2+mot

3) en considérant la capture d'écran suivante déduire une règle concernant la multiplication d'une chaine de caractère par un nombre.

```
In [15]: b*1.5
Traceback (most recent call last):

  File "<ipython-input-15-ca2237451620>", line 1, in <module>
    b*1.5

TypeError: can't multiply sequence by non-int of type 'float'
```

4) A quoi sert la fonction len?

```
In [20]: machaine="moi tarzan"

In [21]: len(machaine)
Out[21]: 10

In [22]: machaine1="jane"

In [23]: len(machaine1)
Out[23]: 4
```

- Pour accéder à un caractère d'une chaine, il faudra faire suivre le nom de la chaine par un crochet contenant l'index du caractère.

Voir l'exemple à droite :

```
In [43]: texte="Je suis heureux d'apprendre Python et de progresser"

In [44]: texte[3]
Out[44]: 's'

In [45]: texte[0]
Out[45]: 'J'

In [46]: texte[2]
Out[46]: ' '
```

On remarquera que l'index est

légèrement décalé par rapport

à ce qu'on pouvait penser

En fait pour les chaine de caractère et les listes le premier élément a pour index 0, le suivant a pour index 1 ainsi de suite.

On peut prendre plusieurs caractère d'une chaine à la fois

5) Après avoir testé les commandes suivantes :

texte[3:7] texte[8:] texte[:13]

Vous indiquerez à quoi servent les commandes suivantes :

chaine[i:j] chaine[i:] chaine[:j]

```
In [24]: texte="Je suis heureux d'apprendre Python et de progresser"

In [25]: texte.count("eu")
Out[25]: 2

In [26]: texte.count("u")
Out[26]: 3

In [27]: texte.count("p")
Out[27]: 3
```

6) On donne la capture d'écran,

En déduire l'utilité de la méthode count

Une méthode est une sorte de fonction un peu étrange.

Elle agit sur un objet et son action est généralement en relation avec des paramètres.

Dans ce qui précède : texte.count(''u '') l'objet est la chaine de caractère s'appelant **texte**

Le paramètre c'est le caractère ''u''

7) Copier le texte suivant et stockez le dans une variable nommée **Brassens** les premières strophes de « la mauvaise réputation »:

Au village, sans prétention, J'ai mauvaise réputation ; Que je me démène ou je reste coi, Je pass' pour un je-ne-sais-quoi. Je ne fais pourtant de tort à personne, en suivant mon ch'min de petit bonhomme ; Mais les brav's gens n'aiment pas que l'on suive une autre route qu'eux… Non, les brav's gens n'aiment pas que l'on suive une autre route qu'eux… Tout le monde médit de moi, sauf les muets, ça va de soi. Le jour du quatorze-Juillet, je reste dans mon lit douillet ; la musique qui marche au pas, cela ne me regarde pas. Je ne fais pourtant de tort à personne, en n'écoutant pas le clairon qui sonne ; mais les braves gens n'aiment pas que l'on suive une autre route qu'eux… non les braves gens n'aiment pas que l'on suive une autre route qu'eux… Tout le monde me montre du doigt, sauf les manchots, ça va de soi.

Comptez le nombre d'occurrence du mot ''gens''

8) Petite synthèse

En reprenant la variable brassens indiquez le nombre de fois où apparaît le 44ème caractère de ce texte.

III Les listes

C’est une structure très importante en Python

Elle est délimitée par des crochets

C’est en fait une collection ordonnée de différents objets, qui peuvent avoir un type différent (texte, nombre). Chaque objet est séparé du suivant par une virgule.

Exercices

1) Ci-contre on a la séquence suivante

Que faut il taper si on veut renvoyer :

- Le premier objet de la liste ?
- L'avant dernier objet de la liste?

2) Quelles sont les différences

avec les chaînes de caractères?

```
In [64]: maliste=[1,2,'lol',3.14,"e"]

In [65]: maliste[1]
Out[65]: 2

In [68]: maliste[-1]
Out[68]: 'e'

In [69]: len(maliste)
Out[69]: 5

In [70]: liste2=["soupe",1,2.41,"pas bon"]

In [71]: maliste+liste2
Out[71]: [1, 2, 'lol', 3.14, 'e', 'soupe', 1, 2.41, 'pas bon']

In [72]: maliste.count(1)
Out[72]: 1
```

3) A quoi sert la fonction append?

Est-ce que cette fonction peut permettre d'insérer entre le 2 et le 3 le mot « coucou »? Pourquoi?

```
In [74]: liste=[1,2,3]

In [75]: liste.append(6)

In [76]: print(liste)
[1, 2, 3, 6]
```

4) compléter la commande [77], pour insérer « coucou » entre le 2 et le 3

Même chose pour insérer le mot « coucou » avant le 1

```
In [76]: print(liste)
[1, 2, 3, 6]

In [77]: liste.insert(
```

```
Arguments
insert(index, object)
```

5) On est arrivé en commande 82, on veut enlever le mot « coucou » qu'il y a entre le 2 et 3, compléter la commande 83, est ce que ça marche ?

```
In [82]: print(liste)
['coucou', 1, 2, 'coucou', 3, 6]

In [83]: liste.remove(
```

```
Arguments
remove(value)
```

Il existe une autre fonction pour enlever des éléments de la liste. Explorer le fonctionnement de la fonction del , par exemple « del liste[4] »

6) Expliquer en détails, ce qu'il se passe entre les commandes 103 et 106.

Expliquer en détails, ce qu'il se passe

Entre les commandes 107 et 110.

Y a-t-il des différences de traitement entre les objets de type nombre et les objets de type liste?

```
In [103]: c=5

In [104]: d=c

In [105]: d=d+1

In [106]: print(c,d)
(5, 6)

In [107]: c=[1,2,3]

In [108]: d=c

In [109]: d.append(2019)

In [110]: print(c,d)
([1, 2, 3, 2019], [1, 2, 3, 2019])
```

```
In [114]: c=[1,2,3]

In [115]: d=c[:]

In [116]: d.append(2019)

In [117]: print(c,d)
([1, 2, 3], [1, 2, 3, 2019])
```

A gauche on a résolu le problème précédent, expliquer la commande 115.

Quelques listes de base

7) en python 2 range(n) renvoie tous les entiers de 1 à n-1.

```
In [34]: print(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

pour avoir le même résultat sous python 3, on utilisera la commande list(…) qui convertira l'objet obtenu en liste utilisable.

Qu'obtiendra t on en tapant : list(range(7))

8) Range peut prendre 1,2 ou trois paramètres.

Après avoir testé range(5,10) , range(20,52) , range(20,52,2) et range(20,52,5)

Expliquer à quoi correspond range(i,j) , range(i,j,k)

Approche alternative :

en important la librairie numpy,

la liste arange(a,b,c) permet de générer

la liste des nombres entre a et b-c, avec

un pas de c

```
In [38]: from numpy import *

In [39]: print(arange(10,15,0.1))
[ 10.   10.1  10.2  10.3  10.4  10.5  10.6  10.7  10.8  10.9  11.   11.1
  11.2  11.3  11.4  11.5  11.6  11.7  11.8  11.9  12.   12.1  12.2  12.3
  12.4  12.5  12.6  12.7  12.8  12.9  13.   13.1  13.2  13.3  13.4  13.5
  13.6  13.7  13.8  13.9  14.   14.1  14.2  14.3  14.4  14.5  14.6  14.7
  14.8  14.9]
```

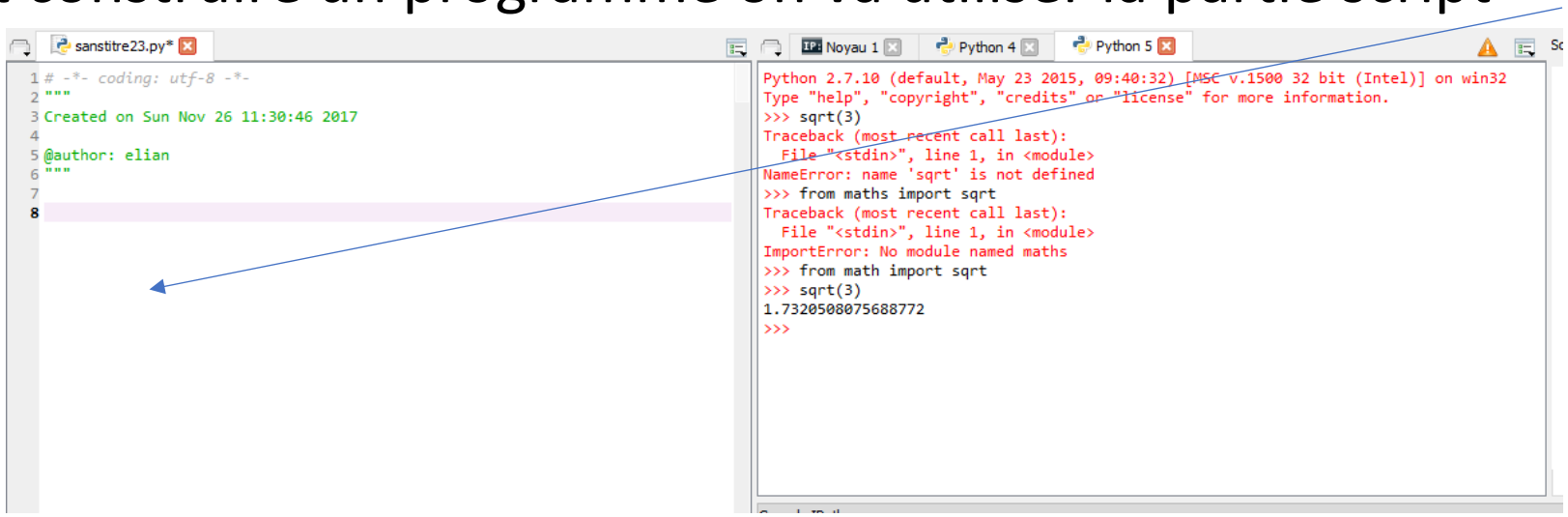
9) Comment créer la liste suivante : [-5,-4.8,-4.6,-4.4, … , 17.6] ?

10) proposer une liste des entiers de 13 à 955

11) proposer une liste des nombres décimaux arrondis au dixième allant de 147 à 3201

IV Les programmes

Si on veut construire un programme on va utiliser la partie script

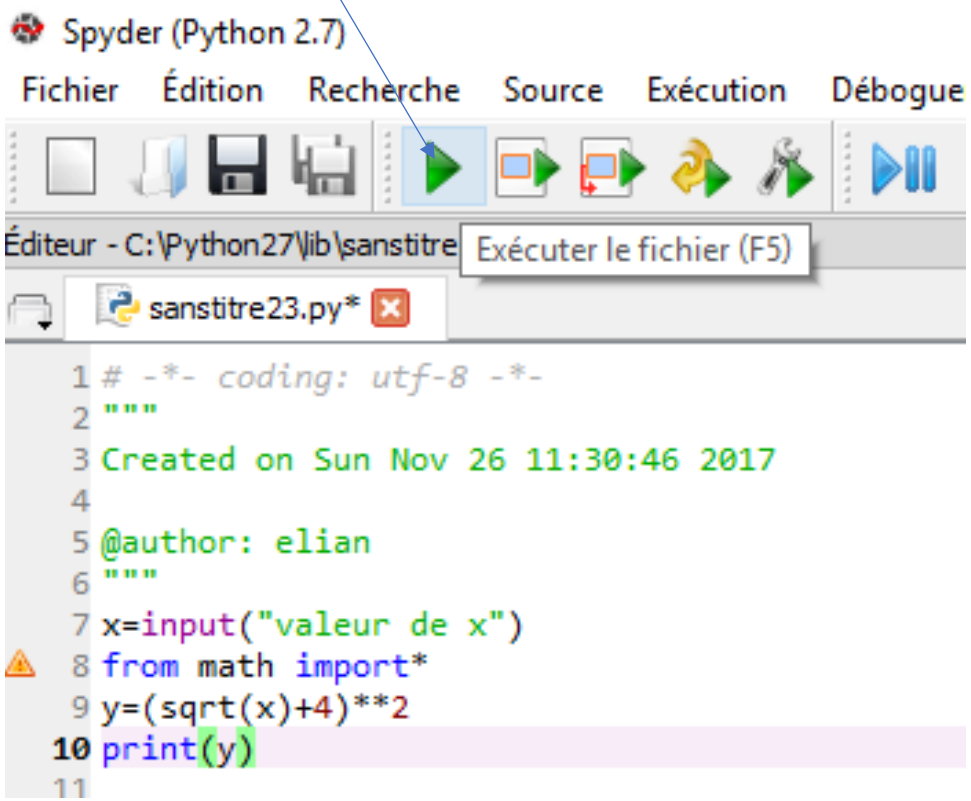


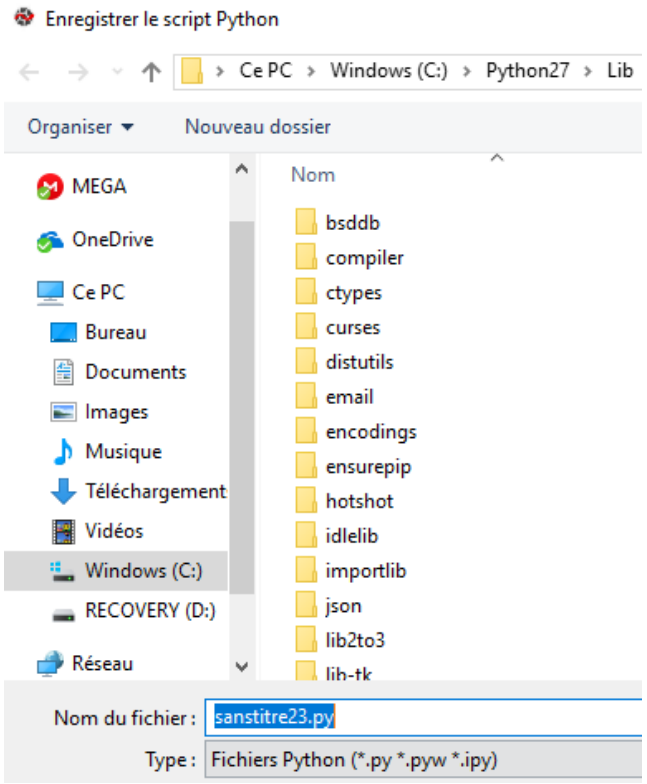
Une fois que le programme est écrit, on l'enregistre, puis on clique sur le bouton Play (flèche verte) présent dans la bande supérieure (s'il vous pose des questions, appuyez juste sur OK).

L'exécution (question, affichage) se fait dans la console.

En cas de problème on peut interrompre l'exécution du programme en appuyant sur le carré bleu.

Pour faire fonctionner le programme (script) il suffit de taper sur la flèche d'exécution le logiciel propose d'enregistrer le script. Pour pouvoir vous y retrouver, il est conseillé de créer un répertoire où vous allez stocker tous les programmes fait durant la séquence





Ton premier programme

Généralement un programme est composé de trois types d'instructions :

- **Entrée** : on va récupérer de l'information extérieure à travers le clavier, la souris, l'écran (quand il est tactile), la caméra, le micro.
- **Traitement** : on utilise des formules et des instructions sur les informations récupérée avant de le rendre.
- **Sortie** : on va offrir à l'utilisateur une production au travers de l'écran, des enceintes…

Exemple :

```
nom = input(“ quel est ton nom ?  “)

phrase = “ Bonjour “ +nom+ “, j’espere que tu vas bien “

print(phrase)
```

Copier/coller ce programme et vérifiez son bon fonctionnement.

Exercice 1 : A toi de jouer : *dans fichier, sélectionne nouveau , pour créer un nouveau programme*
Ecris un script (programme) demandant à une personne son nom, son prénom, puis le parfum de sa glace préférée et qui en retour affichera une phrase du type « Bonjour, Julien Kergot, ton parfum de glace préféré est chocolat »

Programme utile en mathématiques

Le professeur voudrait que les élèves de sa classe tracent la courbe de la fonction f qui à tout réel positif x associe le nombre : $(\sqrt{x}+4)^2$

Pour commencer il demande à ses élèves de remplir un tableau de valeurs.

Cette situation est un brin répétitive, et l'informatique peut nous aider à remplir le tableau plus rapidement.

Dans un premier temps crée un programme qui va demander à l'utilisateur la valeur de x et qui lui donnera en retour la valeur de f(x)

Remarque : tu pourras t'aider de ce qui a été vu sur les opérations de base et la fonction sqrt() découverte précédemment.

*Correction : voir deux diapos plus haut ! M. Bellante a choisi d'utiliser import * plutôt que import sqrt, avec ce choix il peut utiliser n'importe quelle fonction de la librairie math.*

Exercices

```
8 x=input('lettre')
9 texte=input("texte")
10 n=texte.count("x")
11 print(n)
```

2) On a envie de faire un programme comptant le nombre de fois où une lettre est répétée dans un texte.

Un élève propose le script ci-dessus.

Recopiez le et testez le avec les deux configurations suivantes :

Lettre : « e » et texte : « xavier est gentil »

Lettre : « w » et le texte « le rappeur xxxtentation est mort en juin 2018 à l'âge de 20 ans»

Quelle réponse obtenez vous pour les deux cas ?

Quelle lettre est ce que le programme a compté (au lieu de « e » et « w »)?

Déterminez où est l'erreur et corrigez la.

247	22
−22	11
27	
−22	
5	

3) ci contre vous avez la division de 247 (dividende) par 22 (le diviseur) qui nous donne un reste de 5 et un quotient de 11.

Dans la console taper : int(247/22) puis 247%22

Créer un programme demandant à l'utilisateur un dividende et un diviseur et qui en retour donnera la valeur du quotient et du reste.

Structure conditionnelle : si alors sinon

Exemple : dire d'un triangle dont on connait les trois mesures s'il est rectangle ou non.

On sait que si le carré de la plus grande mesure est égale à la somme des carrés des mesures des deux autres côtés alors le triangle est rectangle (réciproque du théorème de Pythagore) et sinon il n'est pas rectangle (contraposée du théorème de Pythagore)

On se propose d'écrire un programme demandant à l'utilisateur les trois mesures du triangle et de tester l'égalité citée plus haut, et suivant si elle est vraie ou non de dire si le triangle est rectangle ou non.

On va donc être amené à faire un test, et si celui-ci est concluant d'exécuter certaines instructions et dans le cas contraire d'en exécuter d'autres.

Voici le script du programme répondant à la consigne de la diapositive précédente.

```
1 a=float(input("le plus grand côté"))
2 b=float(input("autre côté"))
3 c=float(input("dernier côté"))
4 if a**2==b**2+c**2:
5     print("le triangle est rectangle")
6 else :
7     print("le triangle n'est pas rectangle")
```

Décortiquons le :

pour les trois premières lignes, on remarquera qu'en plus de input on utilise la fonction float elle convertit le nombre proposé par l'utilisateur en réel.

À la ligne 4 on pose notre test. « si » se traduit en anglais par « if ».

On remarquera qu'entre les deux membres il y a un double égal : ==

Nous, on veut tester l'égalité, or en Python un signe « = » sert à assigner une valeur à une variable, donc la convention pour tester une égalité sera le double égal : ==

À la fin de l'égalité on a écrit « : », ces deux points servent à introduire un bloc d'informations qui ne sera exécuté que si ce qui précède les deux points est vrai.

Une fois qu'on a écrit les deux points et qu'on appuie sur « entrer » l'éditeur de programme va à la ligne et « indente » le texte (ce qui veut dire qu'il provoque un décalage automatique de quatre espaces vers la droite)

Tant que le texte est décalé on est dans le bloc qui ne s'exécutera que si la condition est vraie,

si on se ramène au bord gauche l'instruction s'exécutera dans tous les cas.

```
1 a=float(input("le plus grand côté"))
2 b=float(input("autre côté"))
3 c=float(input("dernier côté"))
4 if a**2==b**2+c**2:
5     print("le triangle est rectangle")
6 else :
7     print("le triangle n'est pas rectangle")
```

À la ligne 6 on est revenu contre le bord gauche donc elle s'exécute même si la condition n'est pas réalisée d'ailleurs l'instruction est « else » ce qui veut dire « sinon », et elle indique que le bloc qui va suivre ne s'exécutera justement que si la condition de la ligne 4 n'est pas réalisée.

On peut tester autre chose que des égalités.

Nom	Python
est égal à	`a == b`
n'est pas égal à	`a != b`
est plus grand que	`a > b`
est plus petit que	`a < b`
est plus grand ou égal à	`a >= b`
est plus petit ou égal à	`a <= b`
ET logique	`a and b`
OU logique	`a or b`
négation (NOT)	`not a`

Pour des conditions complexes il faudra généralement décomposer en plusieurs conditions simples.

Deux exemples :

- Si on veut savoir si $x \in [8; 12]$, il faut d'abord l'exprimer sous forme d'inégalités : $8 \leq x \leq 12$, puis décomposer cet encadrement : il revient à dire que x vérifie : $8 \leq x$ et $x \leq 12$ sur python on écrira donc `if 8<=x and x<=12 :`
- Si on veut savoir si n'est pas dans $[8; 12]$ on peut utiliser deux approches :

  Ne pas être dans $[8; 12]$ est le contraire d'être dans $[8; 12]$ donc en Python on écrira :
  `if not (8<=x and x<=12) :`

  Ou

  ne pas être dans $[8; 12]$ revient à être soit plus petit que 8 ou strictement plus grand que 12, donc on écrira : `if x<8 or x>12 :`

Exercices

4) On sait que la distance d'arrêt sur sol sec d'un véhicule roulant à la vitesse v(en km/h) est la somme de la distance parcourue durant le temps de réaction $(0{,}3 \times v)$ et de la distance de freinage $(0{,}0075 \times v^2)$

Créer un programme demandant à l'utilisateur la vitesse du véhicule et la distance séparant le véhicule de l'obstacle, et qui en retour affichera après un court calcul et une comparaison « désolé mais ton véhicule a emboutit l'obstacle » ou « tu as réussi à éviter l'obstacle mais reste vigilant! ».

```
In [30]: from random import randint

In [31]: randint(1,6)
Out[31]: 4
```

5)La librairie random permet d'importer la fonction randint

Si on tape « randint(1,6) » , l'ordinateur permet de ressortir un nombre entier aléatoire entre 1 et 6. Bref cela permet de jouer au dé.

On veut programmer un assistant pour jouer aux petits chevaux. A chaque tour le joueur lance un dé à six faces. Si son cheval est dans l'écurie il ne peut l'en sortir que si le dé affiche un 1 ou un 6. Si le cheval est déjà sorti de l'écurie on avance du nombre de case affiché par le dé.

Le programme simule le lancé d'un dé, affiche le résultat obtenu et affichera « si ton cheval est dans l'écurie tu as le droit de le sortir » ou « si ton cheval est dans l'écurie, désolé mon coco mais il va y rester un tour de plus ! »

6) Un jeu de cartes a 32 cartes (autrement dit il y a quatre familles, et on a retiré les cartes 2, 3, 4, 5 et 6.

Construire un programme permettant de simuler une partie de ce jeu. Pour plus de commodité on numérote les 32 cartes, les 8 premières sont les piques, les huit suivantes des cœurs, puis des trèfles, et enfin des carreaux. Elles sont rangées dans l'ordre suivant : as,7,8,9,10,valet, dame, roi. Ainsi la carte numéro 2 est un 7 de pique, la carte numéro 8 est un as de cœur, …

<u>Version facile</u> : Le programme doit simuler un tirage et dire si on a gagné (on a tiré un as) ou si on a perdu (dans le cas contraire).

<u>Version un peu plus difficile</u> : le programme doit aussi indiquer la famille et la nature de la carte

Astuce : on remarquera que si on fait une division euclidienne du numéro de carte par 8 , les cartes de même nature donnent le même reste (par exemple les as sont en position 1,9, 17 et 25, et quand on divise ces nombres par 8, on obtient à chaque fois un reste de 1.

LES BOUCLES POUR

```
In [52]: for k in range(10):
    ...:     print('salut')
    ...: print('aurevoir')
    ...:
salut
salut
salut
salut
salut
salut
salut
salut
salut
salut
aurevoir
```

Une boucle pour permet de recommencer des instructions, un nombre déterminé de fois

Un exemple

Dans l'algorithme ci-contre l'action que l'on

recommence 10 fois est d'afficher le texte 'salut'.

En fait, il y a 10 même action numérotées de 0 à 9

qui se renouvelle. Attention l'instruction qui se renouvelle est mise en retrait sous le k de 'for k in range(10)'

Une fois que les commandes de la boucle sont terminées, le programme passe à la suite des instructions. L'instruction afficher 'aurevoir' est écrite sous le for, donc elle s'effectuera après cette boucle

```
In [53]: for k in range(10):
    ...:     print('salut')
    ...:     print('aurevoir')
    ...:
salut
aurevoir
salut
aurevoir
salut
aurevoir
salut
aurevoir
salut
aurevoir
salut
aurevoir
salut
aurevoir
salut
aurevoir
salut
aurevoir
salut
aurevoir
```

Un autre exemple: dans cet exemple, comme les instructions print(‘salut’) et print(aurevoir’) sont écrites l’une après l’autre, sous le ‘k’ de ‘for k in range(10)’. ON a donc 10 affichages successifs de ‘salut’ et ‘aurevoir’

Exercices

7) Construire un algorithme permettant de simuler 10 lancers d'un dé à six faces (à chaque lancer le programme écrira le résultat du lancer)

8) Reprendre le programme précédent pour qu'à chaque lancer il affiche « gagné » (si on a tiré un 6) ou « perdu » (pour les autres tirages).

9) Analyser le programme ci-contre

```
for k in range(4):
    L=L+[randint(1,6)]
print(L)
```

et indiquer ce qu'on peut s'attendre à avoir en sortie. A quoi pourrait servir un tel programme?

10) Proposer un algorithme permettant de jouer 5000 fois au dé, il doit permettre d'afficher le nombre de fois où on a gagné (obtenu 6)

11) Proposer un programme permettant de simuler 1 tirage du loto (c'est-à-dire 5 nombres pris entre 1 et 49)

Version 1 : on s'autorise à tirer plusieurs fois le même nombre

Fonctionnement interne de la boucle for

On peut avoir l'impression que les boucles précédentes se contentent de répéter 10 fois le bloc qui est indenté.

C'est le cas, mais la situation est plus riche que ça.

range(10) correspond comme on a pu le voir précédemment à une sorte de liste allant de 0 à 9.

Ce que la boucle fait, c'est qu'à chaque tour la variable « i » va parcourire la liste et prendre une valeur différente : 0 puis 1 puis 2 … puis 9.

Si on écrit le code ci-contre

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

, on obtiendra la sortie ci-contre.

```
>>>
0
1
2
3
4
5
6
7
8
9
```

On n'est pas obligé d'utiliser la fonction range pour faire une boucle « for », n'importe quelle liste fera l'affaire.

Exemple : tapez le script ci-dessous et observez l'affichage quand on l'exécute.

```
mois=0
for i in ["janvier", "février", "mars"] :
    mois=mois+1
    phrase=i+" est le mois n°"+mois+" de l'année"
    print(phrase)
```

.

Exercices

12) Reprendre le programme 7 (deux diapos plus haut) pour qu'à chaque lancer il fasse une phrase du type : « le lancer n°5 vaut 4 »

13) En vous inspirant des propriétés de la fonction range, afficher successivement les nombres entre 500 et 563 en allant de 3 en 3 (c'est-à-dire 500 puis 503 puis 506 puis …)

14) Reprendre l'exercice « programme utile en mathématiques » qui s'intéresse à la fonction $(\sqrt{x}+4)^2$ de telle sorte qu'il fasse plus que donner l'image d'un nombre unique (il demandera un x minimum, un x maximum et un pas, puis avec une boucle il va afficher des lignes de la forme « f(0)=16 », « f(1)=25 », etc)

15) Dans une équipe de volley féminin les joueuses ont pour nom : Alice, Nawel, Lou, Alexandra, Candice et Shaïna. Faire un programme affichant consécutivement 6 phrases écrites sur le modèle suivant : « Alice a 5 lettres dans son prénom »

<u>rappel</u> : pour déterminer le nombre de lettre dans le mot stocké dans la variable nom on écrit : len(nom)

Palindromes

Un palindrome est un mot qui se lit de la même manière qu'on aille de la droite vers la gauche ou de la gauche vers la droite. Quelques exemples connus : rotor, xanax, kayak

Le but de l'exercice est d'écrire un programme permettant de dire si un mot est un palindrome. Comme ça n'a rien d'une évidence, on vous donne pour vous guider un algorithme commenté :

Algorithme	**commentaire**
Demander à l'utilisateur mot	mot est le mot dont ton cherche à savoir s'il est un palindrome.
On détermine n le nombre de lettres dans le mot.	Voir dernière ligne de la diapo précédente
On pose palin=True	la variable palin dit si le mot est un palindrome et on considère que jusqu'à preuve du contraire, le mot en est un.
I prendra ses valeurs entre 0 et n	on utilise une boucle pour parcourir les lettres du mot les unes après les autres
Si la lettre de rang i n'est pas la même que la lettre en position symétrique	
Alors palin prend la valeur False	il suffit qu'il y ait un couple de lettres non symétriques pour que mot ne soit pas un palindrome
(pas de Sinon)	si les deux lettres sont identiques, il n'y a rien à faire, ça ne change en rien ce qu'on a déterminé précédemment
On affiche le contenu de la variable palin	Une fois que la boucle est terminée

1) Donner pour un rang i d'une lettre , le rang de la lettre symétrique en fonction de i et de n (le nombre de lettre du mot)

Petit exemple pour y voir plus clair sur les indices des lettres.

« vietnamien » est un mot de 10 lettres, elles seront numérotées de 0 à 9, ainsi mot[1] vaut « i »

La position symétrique de la lettre 0 est 9, la position symétrique de la lettre 1 sera 8 etc

2) Traduire l'algorithme sous la forme d'un programme en Python

Boucles tant que (while)

On rappelle que les instructions déclarées dans la boucle tant que, se feront tant que la condition reste vraie. Une fois que celle-ci n’est plus vrai, on passe à l’instruction suivante

Souvent, on utilise la boucle tant que, pour parvenir à une fin bien précise, dans l’esprit suivant

Tant que je rate j’essaie encore

C’est une boucle non bornée, en effet cette boucle ne peut jamais s’arrêter. Et oui on peut toujours rater le test

```
In [1]: t=5

In [2]: while t<10:
   ...:     t=t+1
   ...:

In [3]: print (t)
10
```

Algorithme 1

Dans l'algorithme 1 :t au départ vaut 5,le but est d'obtenir Une valeur de t plus grande ou égale à 10, Tant que t<10; l'instruction est de remplacer la valeur de t par t+1, ce qui fait donc augmenter t. de ce fait on a toute les chances pour réussir à être plus grand ou égal à 10

```
In [4]: t=5

In [5]: while t>6:
   ...:     t=t+1
   ...:

In [6]: print(t)
5
```

Algorithme 2

Le but est d'être inférieure ou égal à 6, or t est au départ égal à 5.

Donc comme la condition d'entrée dans la boucle tant que, n'est pas vérifiée, la valeur de t ne bouge pas. On passe donc à l'instruction suivante, c'est-à-dire écrire t qui vaut 5

Petite devinette

Expliquer ce qui se passe dans cet affichage

```
In [8]: while t<10:
   ...:     print('boujour')
   ...:

boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
boujour
Traceback (most recent call last):

  File "<ipython-input-8-1dfa3bde9ddb>", line 2, in <module>
    print('boujour')

  File "C:\Python27\lib\site-packages\IPython\kernel\zmq\iostream.py", line 202, in write
    string = string.decode(self.encoding, 'replace')

  File "C:\Python27\lib\encodings\utf_8.py", line 16, in decode
    return codecs.utf_8_decode(input, errors, True)

KeyboardInterrupt
```

Exercices

16) Reprendre l'exercice « programme utile en mathématiques » (déjà repris dans la section « for ») qui s'intéresse à la fonction $(\sqrt{x}+4)^2$ de telle sorte qu'il fasse plus que donner l'image d'un nombre unique (il demandera un x minimum, un x maximum et un pas, puis avec une boucle il va afficher des lignes de la forme « f(0)=16 », « f(1)=25 », etc) Cette fois ci, on n'utilisera pas la fonction range.

17) Créer un programme de devinette. Il fonctionnera de la manière suivante : le programme choisit aléatoirement un nombre entier entre 1 et 100. le programme demande alors de deviner la valeur du nombre. Tant que la proposition n'est pas bonne, le programme affichera « trop petit » ou « trop grand » selon le cas rencontré, puis reposera la question. Une fois que l'on aura obtenu la bonne réponse, le programme affichera : « bravo, tu as gagné en …. coups » (vous utiliserez un compteur, qui va augmenter de 1 à chaque fois que l'utilisateur fera une proposition, et qui permettra donc de compléter les …. de la phrase de félicitations.)

18) Reprendre l'exercice de tirage de numéros du loto, cette fois ci on n'aura pas le droit de tirer deux fois le même nombre.

V Les fonctions

Une fonction est un procédé dépendant ou non d'une variable, ou de plusieurs variables permettant de retourner un calcul ou autre

Un premier exemple

C'est l'exemple classique mathématique qui à toute valeur x on fait correspondre le réel $f(x) = 2x + 1$.

Evidemment , $f(2) = 2 \times 2 + 1 = 5$

```
In [9]: def f(x):
   ...:     return 2*x+1
   ...:

In [10]: print(f(2))
5
```

Un deuxième exemple

Ici la fonction permet de générer un nombre x de parties demandées par l'utilisateur . A l'issue de ces x parties la fonction retourne le nombre de fois où on a gagné

```
In [14]: def f(x):
    ...:     L=[]
    ...:     for k in range(x):
    ...:         if randint(1,6)==6:
    ...:             L=L+['gagné']
    ...:         else:
    ...:             L=L+['perdu']
    ...:     c=L.count('gagné')
    ...:     return c
    ...:
```

```
In [15]: f(10)
Out[15]: 1

In [16]: f(10)
Out[16]: 1

In [17]: f(10)
Out[17]: 1

In [18]: f(20)
Out[18]: 4

In [19]: f(20)
Out[19]: 7

In [20]: f(30)
Out[20]: 4
```

Exercice

1) Proposer la fonction lotoGen(x) permettant de générer un nombre x de tirages du loto. (la sortie sera donc une liste contenant x listes de 5 éléments)
2) Créer la fonction palin(mot) qui détermine si un mot est un palindrome ou non, elle retournera alors la valeur True ou False
3) Créer la fonction summ(min, max) qui détermine la somme de tous les entiers compris entre les entiers min et max.
4) Créer la fonction facto(nombre) pour déterminer la factorielle d'un nombre. (pour information : la factorielle de 5 se note 5! et elle vaut $5 \times 4 \times 3 \times 2 \times 1 = 120$, et de manière générale $n! = n(n-1)(n-2) \dots 3 \times 2 \times 1$)