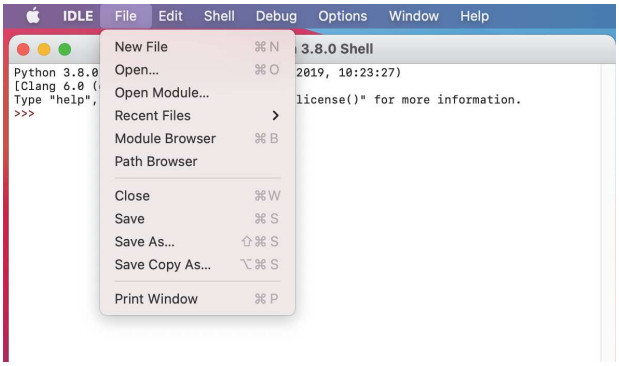
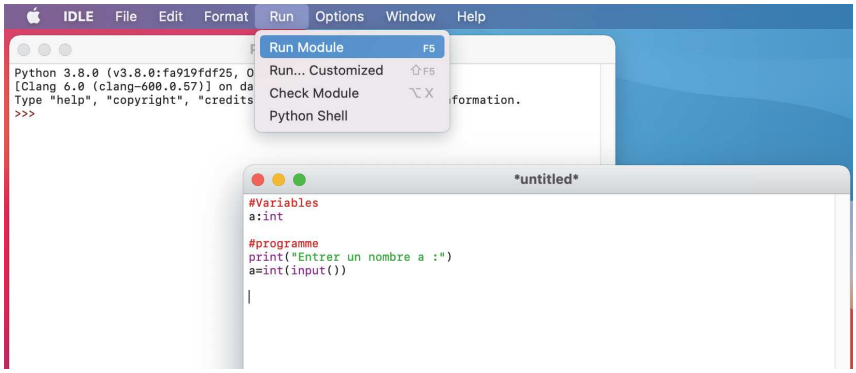
Environnement de travail

Le langage Python servira de support au module.

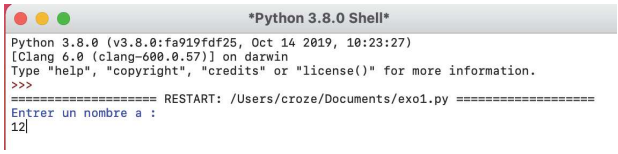
Vous avez deux possibilités pour écrire vos programmes :

1/ Soit vous utilisez l'outil Anaconda Prompt :
Dans le menu démarrer, choisissez **Anaconda Prompt**. Une fenêtre va s'ouvrir. Ecrivez **idle**
Cet environnement rassemble un éditeur de code et un compilateur qui vérifie la syntaxe et exécute le programme.

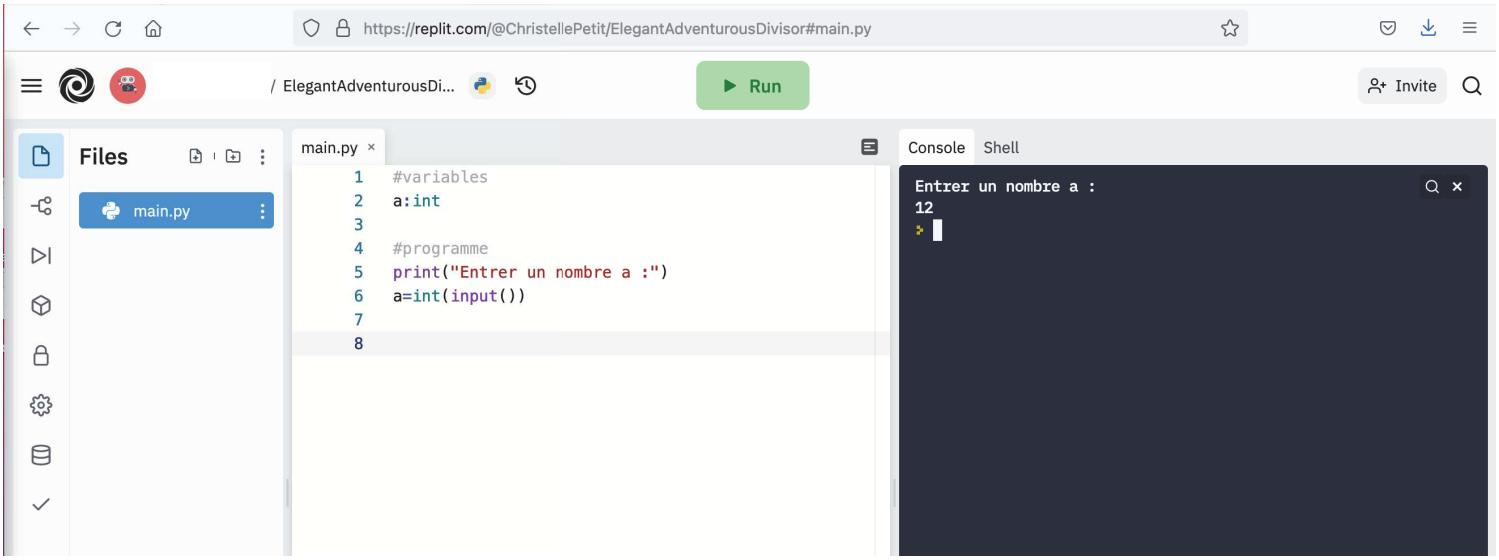




Créer un nouveau programme : Onglet File/New File - vous pouvez écrire le code de votre application dans ce fichier. *Vous créerez un fichier par exercice (que vous nommerez Nom_prenom_exo1.py)*
Tester le programme : Onglet Run / Run module - La fenêtre Console vous permet d'interagir avec votre application.



2/ Si anaconda n'est pas installé sur l'ordinateur, utilisez l'outil en ligne : repl.it (www.replit.com)
Créer un nouveau programme : "Create repl" puis choisissez Python. Vous écrirez votre code dans la partie centrale.
Tester votre application : Cliquez sur le rectangle vert "RUN" et regardez ce qui se passe dans la partie de droite.
Récupérez votre fichier en le téléchargeant à partir des . Sauvegardez le et renommez le *Nom_prenom_exo1.py.*



Exercice 0. Prise en main de Python

Le programme suivant illustre la façon d'écrire en Python les instructions de base (lire, écrire et affectation).

Ecrire et exécuter le programme suivant :

```
# déclaration des variables
a:int;
b:int;

#programme principal

print("Entrer le premier entier : ") ;
a = int(input());
print(a) ;

b = int(input("Entrer le second entier :"));
print(b) ;
```

En python, vous pouvez regrouper les 2 instructions écrire puis lire en une seule instruction.

Il correspond à une version de l'algorithme suivant en pseudo-langage.

```
programme exemple_simple
Var a, b : ENTIER ;
début
    écrire("Entrer le premier entier :");
    lire(a) ;
    écrire(a);
    écrire("Entrer le second entier :");
    lire(b) ;
    écrire(b) ;
fin
```

L'**alternative** repose sur le schéma suivant : si une condition est vérifiée alors on exécute une (ou plusieurs) actions ; sinon on exécute une (ou plusieurs) autres actions.

Exemple en pseudo-langage :

```
programme exemple_if
Var a, b : ENTIER ;
début
    lire(a) ;
    lire(b) ;
    si (a > b) alors
       écrire(a, b) ;
    sinon
       écrire(b, a) ;
    fin si
fin
```

Exemple en python :

```
a:int;
b:int;

a=int(input());
b=int(input());
if (a>b):
  print(a,b);
else:
  print(b,a);
```

En python, le mot clé "alors" est représenté par les : Les instructions qui suivent sont toutes décalées à droite par rapport au début de la ligne.
Le fait de revenir au début de la ligne marque la fin.

Le programme suivant illustre la façon d'écrire en python la boucle **tant que**.

Exemple en pseudo-langage :

```
programme exemple_while
Var i : ENTIER ;
début
 i <- 20;
 tant que (i > 0) faire
    écrire("i = ", i) ;
    i <- i-1;
 fin tant que ;
fin
```

Exemple en python :

```
i:int;
i=20;
while (i>0) :
    print("i = ",i);
    i = i-1;
```

Le programme suivant illustre la façon d'écrire en python la boucle **pour**.

Exemple en pseudo-langage :

```
programme exemple_for
Var i : ENTIER ;
début
 pour i de 0 à 10 pas 1 faire
   ecrire(i) ;
 fin pour ;
 pour i de 0 à 10 pas 2 faire
   ecrire(i) ;
 fin pour ;
 pour i de 10 à 0 bas 1 faire
  ecrire(i) ;
 fin pour ;
fin
```

Exemple en python :

```
i:int;
for i in range(0,11,1):
    print()
for i in range(0,11,2):
    print(i, end=" ");
print()
for i in range(10,-1,-1):
    print(i, end=" ");
print()
```

En python, la valeur max de la boucle POUR n'est pas comprise.

Exercice 1 : Calcul des moyennes

Écrire un algorithme permettant à l'utilisateur de choisir s'il souhaite effectuer une moyenne classique ou une moyenne pondérée de deux notes (lues) d'un étudiant. Le programme calculera et affichera la moyenne choisie.

Exercice 2. Conception d'algorithme, codage et test

On donne l'algorithme suivant.

```
programme affichage_de_valeurs
var choix : ENTIER ;
var borne_inf, borne_sup : ENTIER ;
début
  choix <- -1 ;
  écrire("Entrer la valeur de la borne inférieure : ") ;
  lire(borne_inf) ;
  écrire("Entrer la valeur de la borne supérieure : ") ;
  lire(borne_sup) ;
  tant que (choix ≠ 0) faire
     écrire("0 – Quitter le programme") ;
     écrire("1 – Afficher les valeurs comprises entre les bornes") ;
     écrire("2 – Afficher les valeurs paires comprises entre les bornes") ;
     écrire("3 – Afficher  les valeurs impaires comprises entre les bornes") ;
     écrire("Entrer votre  choix : ") ;
     lire(choix) ;
     si (choix = 1) alors
         écrire("Valeurs entre les bornes") ;
     fin si ;
     si (choix = 2) alors
         écrire("Valeurs paires entre les bornes") ;
     fin si ;
     si (choix = 3) alors
         écrire("Valeurs impaires entre les bornes") ;
     fin si ;
  fin tant que ;
fin
```

1. Coder l'algorithme en Python.
2. Compléter l'algorithme pour réaliser les traitements demandés.
3. Tester le code.

Exercice 3 : Jeu

Le joueur doit deviner un nombre tiré au hasard dans un intervalle [0,100].
A chaque étape du jeu, le joueur propose un nombre puis le programme affiche : trop grand", "trop petit" ou "gagné". A la fin, le programme affiche le nombre d'essais nécessaires à l'utilisateur afin qu'il trouve (cf exemple ci-dessous).

NB : EN python, pour utiliser la fonctionnalité qui tire au hasard un nombre entre 2 bornes vous devez au début de votre programme écrire :

```
from random import *
```

puis vous devez utilisez la fonction randint pour qu'il génère un nombre entre 0 et 100 :

```
nb=randint(0,100)
```

```
Essai n° 1  : Entrer un nombre entre 0 et 100 :
50
Trop grand.
Essai n° 2  : Entrer un nombre entre 0 et 100 :
25
Trop grand.
Essai n° 3  : Entrer un nombre entre 0 et 100 :
12
Trop petit.
Essai n° 4  : Entrer un nombre entre 0 et 100 :
18
Trop petit.
Essai n° 5  : Entrer un nombre entre 0 et 100 :
20
Trop petit.
Essai n° 6  : Entrer un nombre entre 0 et 100 :
22
Gagné en  6  essais.
```

Exercice 4 : Nombre parfait

Un nombre est parfait s'il est égal à la somme de ses diviseurs propres, c'est-à-dire autres que lui-même.
Par exemple 6 est un nombre parfait car 6 = 1 + 2 + 3
Ecrire un programme qui affiche tous les nombres parfaits compris entre 1 et 1000.