Introduction

Qu'est-ce qu'un programme ? Qu'est-ce qu'un algorithme ?

L'objectif de la programmation est d'apprendre à l'ordinateur comment accomplir de nouvelles tâches. Ainsi, un programme est caractérisé par :

- un objectif
- un ensemble d'opérations à effectuer
- un contexte d'application.

Un algorithme désigne une suite finie de règles à appliquer dans un ordre déterminé, à un nombre fini de données pour arriver, en un nombre fini d'étapes à un certain résultat et cela indépendamment des données (Encyclopédia Universalis)

Exemple : L'algorithme « prendre un fil, répéter dix fois l'opération consistant à enfiler une perle bleue puis une perle rouge, nouer les deux bouts du fil » permet à des enfants de quatre ans de construire un bracelet à partir de fil et de perles.

Remarques :

- Les opérations doivent être déterministes : cela signifie que la même opération effectuée sur les mêmes données doit rendre le même résultat.
- L'ordre des instructions est important.
- Un bon algorithme est un algorithme qui se termine, qui donne le résultat attendu, qui nécessite un temps d'exécution raisonnable.

Qu'est-ce qu'un langage de programmation ?

L'ordinateur ne « comprend » que deux états : le 0 et le 1. En effet, dans le fond, l'ordinateur ne fait que manipuler des signaux électriques fonctionnant sur une logique de tout ou rien. Le langage de l'ordinateur est donc un langage binaire. Chaque objet, chaque opération est codée par des 0 et par des 1 (ce sont les fameux « bits » réunis par 8 « octets », 16 ou 32).

Afin de pouvoir communiquer avec l'ordinateur, des programmeurs ont créé des langages de programmation qui sont des ensembles de mots-clés et de règles de rédaction conventionnelles qui peuvent être traduits en langage binaire par des systèmes qu'on appelle interpréteur ou compilateur.

Il existe plusieurs systèmes d'interprétation et plusieurs langages de programmation. Cette année nous travaillerons avec le langage de programmation PYTHON.

Les programmes peuvent être écrits directement en ligne dans le terminal du

compilateur. Mais cela n'est généralement pas très pratique. On utilise donc un éditeur de texte : c'est un logiciel qui nous permet de rédiger le programme proprement, de l'enregistrer et de le modifier facilement. Il suffit ensuite de faire compiler ce programme rédigé dans l'éditeur de texte par le compilateur. Pour l'éditeur de texte nous utiliserons Pyzo qui est bien adapté pour écrire des programmes en Python.

Qu'est-ce qu'un bug ?

Lorsqu'on écrit un programme, il existe plusieurs sources d'erreurs, qu'on appelle bugs en anglais.

- les erreurs de syntaxe (la forme) : si les règles de rédaction ne sont pas respectées, le logiciel ne « comprend » pas le programme et s'arrête automatiquement. Chaque règle est primordiale. En particulier soyez vigilant :
  - aux minuscules et majuscules (casse)
  - à la ponctuation (virgule, point, apostrophe, deux points, point virgule, parenthèses)
  - à l'orthographe des mots
  - à l'ordre des mots

  Si vous faites une erreurs de syntaxe, le programme va « planter » et un message d'erreur va s'afficher pour vous expliquer où se trouve le problème. Vous devez apprendre à déchiffrer ces messages d'erreurs pour corriger rapidement et efficacement vos programmes.
- Les erreurs sémantiques (le fond) : dans ce cas le résultat n'est pas celui que vous attendiez, même si le programme s'exécute sans erreur. Dans ce cas, les instructions que vous avez tapées dans votre programme ne sont pas adaptées. Le meilleur moyen pour essayer de comprendre où se situe l'erreur est de reprendre l'exécution de la machine en se mettant à sa place, étape par étape.

Dans tous les cas, rappelez-vous que l'ordinateur n'a pas de cerveau mais vous si. Donc les expressions comme « cet ordinateur fait n'importe quoi ! » ou « il ne fait pas ce que je lui demande » ne veulent rien dire : bannissez-les de votre vocabulaire.

Qu'est-ce qu'une variable ?

Comment l'ordinateur stocke-t-il les données que nous souhaitons manipuler ? Il utilise des variables. En informatique, une variable peut être vue comme une case mémoire:elle est repérée par un nom (c'est l'adresse à laquelle la donnée est rangée dans la mémoire de l'ordinateur) et elle possède un contenu (la donnée en elle-même).

Lorsqu'on enregistre une donnée dans une variable, on dit qu'on affecte la donnée à la variable. Pour cela, on utilise la syntaxe :

```
>>> x=y
```

où x est le nom de la variable et y le contenu.

Remarques :

- Le nom de la variable :
  - est une séquences de lettres et de chiffres qui doit toujours commencer par une lettre.
  - ne peut pas utiliser les accents ou les caractères spéciaux (&,$,@, etc...) sauf le symbole _.
  - est sensible à la casse.
  - Doit être court et explicite (par exemple `alt` pour altitude).
  - Ne peut pas être l'un des mots-clés réservés par le langage (`and, del, from, none, true, false, as, elif, global, nonlocal, try, assert, else, if, not, while, break, except, import, or, with, class, False, in, pass, yield, continue, finally, is, raise, def, for, lambda, return`).
- On remarque que le symbole = dans Python n'a pas le même sens qu'en mathématiques. Dans Python le symbole égal a un sens de lecture particulier.
  Exemples :
  - >>> a=3
    >>> a=6
    Que fait l'ordinateur ? Quelle remarque peut-on faire ?
  - `>>> a=a+1`
    Que fait l'ordinateur ? Quelle remarque peut-on faire ?

Dans le cas de données numériques, Python évalue la quantité à droite de l'égalité avant de l'affecter à la variable. Ainsi, en tapant :

```
>>> x=2+3
```

si on demande au langage d'afficher ce qui est contenu dans x, par exemple en tapant

```
>>> print(x)
```

on obtient 5.

On peut également effectuer des affectations multiples, ou en parallèle comme dans l'instruction suivante :

```
>>> a , b = 4 , 8.33
```

Quels sont les différents types de variables ?

Cette question peut se poser autrement : que peut-on enregistrer comme donnée dans une variable ?

En Python, on peut trouver (entre autre) :

- des données numériques : on distinguera le type *integer* (les entiers) et le type *float* (les réels ou nombres à virgule flottante). Attention, en Python, le séparateur des décimales est le point et non la virgule (convention anglaise comme dans beaucoup de logiciels). Pour ces données, on pourra utiliser les opérations de somme (+), de produit (*) , différences, division (/) et de division euclidienne dans le cas des entiers (// pour le quotient et % pour le reste) et puissance (**).
- Des données alphanumériques : il s'agit des lettres, mots, phrases et ensembles de symboles quelconques. On parle de « chaînes de caractères (type *string*). Une chaîne de caractères est délimitée par des quotes (simples ou doubles ou triples).
  Exemples :
  ```
  >>> phrase1='coucou'
  >>> phrase2='' comment t'appelles-tu ?''
  >>> phrase3=''25''
  ```
  Dans ce dernier exemple, on a bien affaire à une chaîne de caractères et non un nombre. On peut néanmoins transformer le type de cette donnée grâce à la fonction int
  ```
  >>> n=int('25')
  ```
- des listes : ce sont des collections d'objets séparés par des virgules, l'ensemble étant entouré par des crochets.
  Exemple :
  ```
  >>>     jour=['lundi','mardi','mercredi',     'jeudi',
  'vendredi', 'samedi', 'dimanche']
  ```
- des booléens : dans cette catégorie, on ne trouve que deux données, vraie (True ou 1) et faux (false ou 0). Ce type de donnée permet de créer des conditions. Les symboles pour créer des données booléennes sont : == (condition d'égalité), != (différent), > (strictement plus grand), < (strictement plus petit), <= (inférieur ou égal), >= (supérieur ou égal), `or` (la condition est vraie si l'une des deux condition est vraie), `and` (la condition est vraie si les deux conditions sont vraies).
  Des fonctions : certaines fonctions sont prédéfinies dans PYTHON. Par exemple : `input(), int() et print()`. La première permet d'afficher un message dans une boite externe, demandant à l'utilisateur d'entrer une donnée et d'enregistrer cette donnée dans une variable. La syntaxe est la suivante :
  ```
  >>> <variable>=input('<message>')
  ```
  Attention, la variable aura alors le type chaîne de caractères. Pour transformer un nombre qui est du type chaîne de caractères en type numérique, on utilise la fonction int(), de la manière suivante :

```
>>> <variable_numerique>=int(<variable_caractere>)
```
la fonction print() permet d'afficher le message écrit entre les parenthèses.
Il est également possible de définir des fonctions originales comme nous le verrons plus tard.

Dans certains programmes, le programmeur doit indiquer au logiciel le type de la variable quand il la définit (par exemple en indiquant que la variable x sera un entier et que la variable y sera une chaîne de caractère). PYTHON est un langage de typage dynamique c'est à dire qu'il se charge lui-même de définir le type de la variable, en fonction du contenu ou de l'utilisation qui en est faite.

Comment écrire un programme ?

Voici quelques étapes schématiques de la conception d'un programme :

- Identifier clairement l'objectif, les données à manipuler.
- Déterminer l'ensemble des étapes nécessaires pour obtenir le résultat à partir des données.
- Traduire ces différentes étapes en les exprimant à l'aide du langage de programmation
- faire tourner le programme pour vérifier qu'il n'y a pas d'erreurs de syntaxe ou de sémantique.

Pensez à toujours commenter le programme en commençant le commentaire par le symbole # (après ce symbole, le compilateur ignore ce qui est écrit). Il faut commenter tout ce qui est possible afin de rendre le programme facile à lire pour vous-même et pour d'autres. Cela vous oblige aussi à expliciter tout ce que vous faites.

Qu'est-ce qu'un module ?

La version de base de Python contient certaines fonctions usuelles déjà programmées pour faciliter la vie des programmeurs. Par exemple, des fonctions pour communiquer avec l'utilisateur d'un programme :

- `x=input(message)` : cette instruction affiche le `message` puis affecte la valeur entrée par l'utilisateur dans la variable `x`.
- `print(message)` : cette instruction affiche à l'écran le `message`.

Mais il est possible d'utiliser d'autres fonctions qui sont programmées dans des « bibliothèques » que l'on appelle modules en Python, c'est à dire des fichiers qui regroupent des ensembles de fonctions et des constantes. Concrètement, on indique avec la ligne de commande :

```
>>> from nom_du_module import *
```

Par exemple

```
>>> from math import *
```

permet d'accéder à des fonctions mathématiques comme `sqrt()` (la racine carrée) ou `sin()` (le sinus) et à des constantes comme `pi` (le nombre pi).

Nos premiers programmes !

Programme 1 :
Ecrire un programme qui échange les valeurs de deux variables a et b (on pourra utiliser les affections simples puis les affectations parallèles dans un second temps).

Programme 2 (conversions) :

- Écrire un programme qui demande à l'utilisateur d'entrer un angle dont la valeur est exprimée en degrés et qui renvoie la valeur de l'angle en radians.
- Écrire un programme qui demande à l'utilisateur d'entrer une température θ en degré Celsius et qui renvoie la température t en degré Fahrenheit (utiliser la relation : $t(°F) = \theta(°C) \times 1{,}8 + 32$).
- Écrire un programme qui demande à l'utilisateur d'entrer un nombre entier de secondes en un nombre d'années, de mois, de jours, de minutes et de secondes (utiliser l'opérateur %)
- Écrire un programme qui calcule le volume d'un parallélépipède rectangle dont la largeur, la longueur et la hauteur sont fournis en mètre par l'utilisateur.

Programme 3 :
Écrire un programme qui demande à l'utilisateur son année de naissance et qui lui donne son age en 2015.

Programme 4 (utile pour les soldes):
Écrire un programme qui demande à l'utilisateur d'entrer le prix initial d'un produit, le pourcentage de réduction et qui annonce le prix à payer pour le produit soldé.

Programme 5 :
Ecrire un programme qui demande à l'utilisateur de calculer 12*56 et qui renvoie True si le résultat est correct, False sinon.

Programme 6 :
Ecrire un programme qui demande à l'utilisateur de rentrer un nombre entier et qui renvoie True si ce nombre est divisible par 13 et False sinon.

Programme 7 :

« Choisissez un chiffre en 0 et 9, multipliez-le par 9 puis retranchez-le à votre âge multiplié par 10. Si vous me donnez le nombre que vous obtenez, je vous donnerai votre âge».

Essayer de comprendre ce « tour de magie » puis écrivez un programme qui détermine l'âge quand on lui donne le résultat des calculs.
Attention : il y a une limite dans le couple (chiffre,âge) choisi… Si l'âge dépasse 10 ans, pas de souci...