Labo d'introduction à l'informatique

pour les mathématiques

Yann Thorimbert

UNIVERSITÉ DE GENÈVE
CENTRE UNIVERSITAIRE D'INFORMATIQUE

Semaine 1
Variables et expressions

Qu'est-ce qu'une variable ?

- Une variable permet à la machine de mémoriser une valeur.

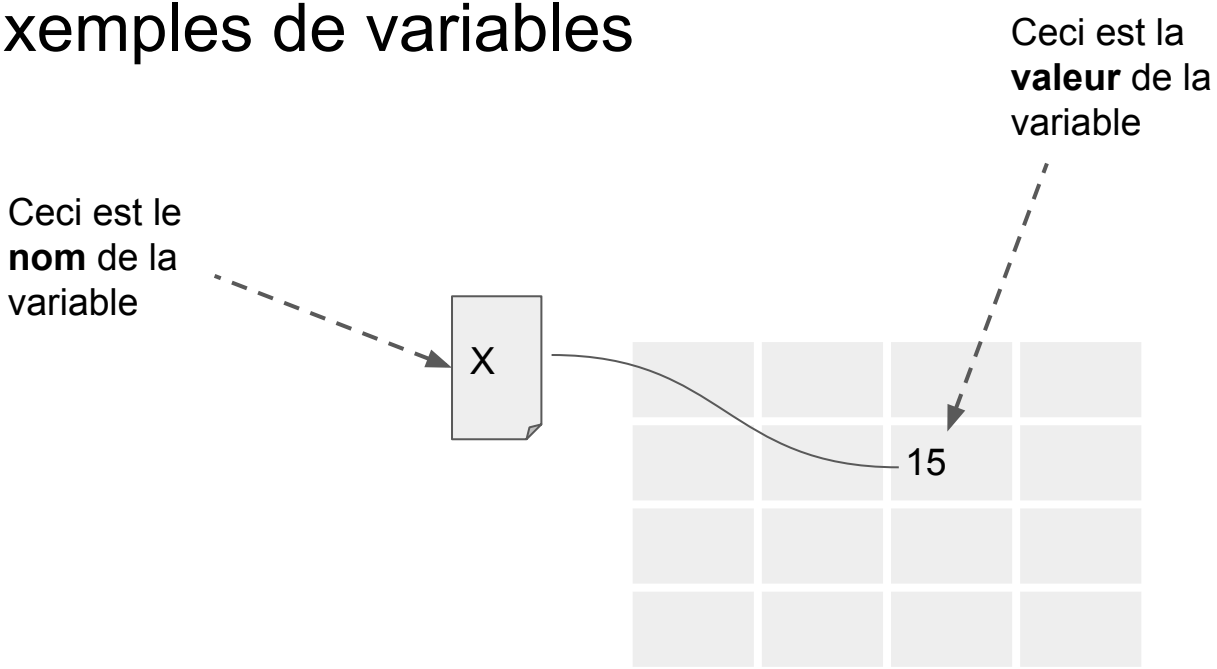
Qu’est-ce qu’une variable ?

Durant l’exécution d’une application, les données utiles sont stockées dans la mémoire centrale (mémoire vive).

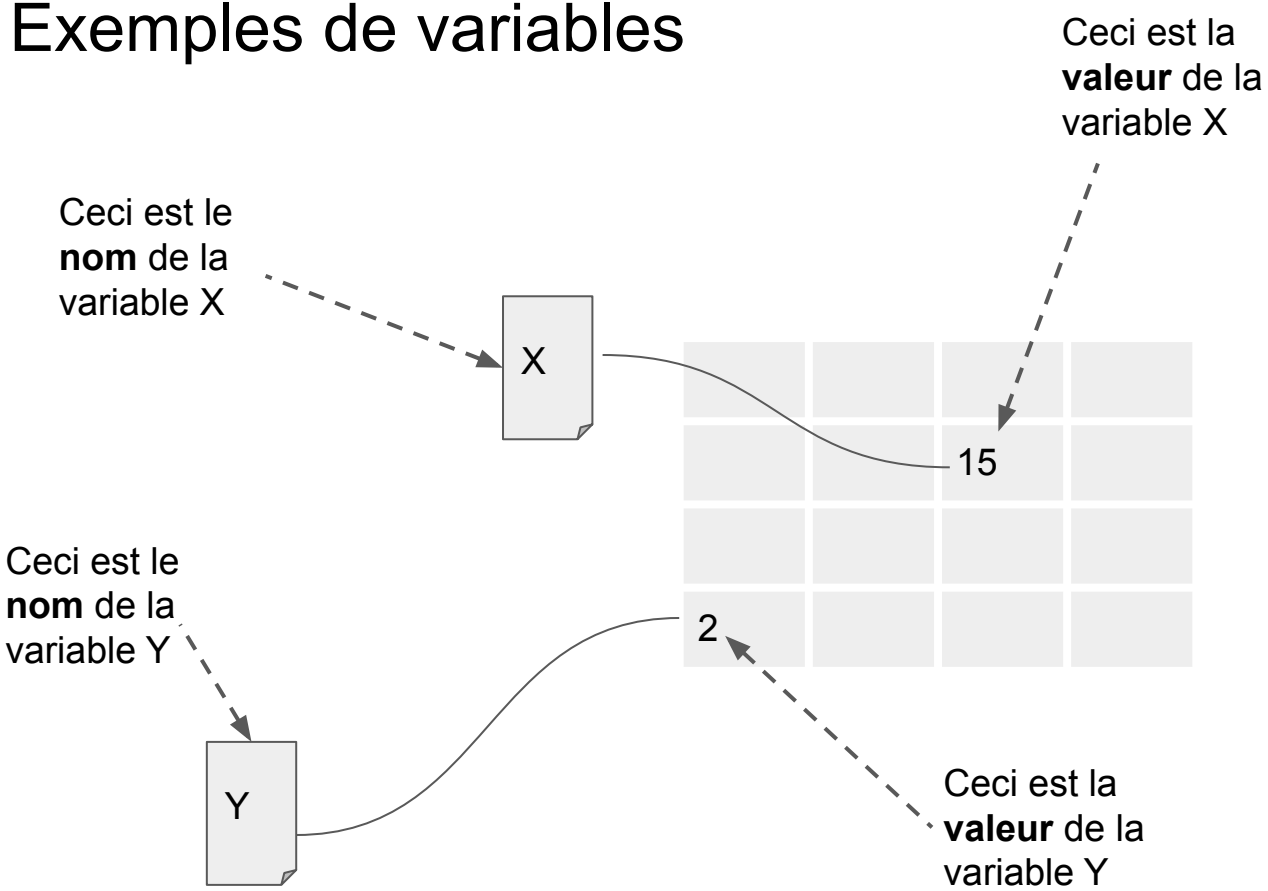
La mémoire vive peut être imaginée comme un ensemble de cases dans lesquelles on peut écrire des valeurs.

Lorsqu’on a besoin d’une valeur, on demande à l’ordinateur d’aller la chercher dans sa mémoire.

Exemples de variables



Exemples de variables



Comment peut-on nommer les variables ?

Dans la plupart des langages de programmation, le nom d'une variable ne peut pas **commencer** par un chiffre.

De même, on ne met pas d'**espace** vide dans le nom d'une variable.

Dans ce cours, on prend l'habitude de toujours commencer le nom d'une variable par une minuscule, et d'utiliser soit l'écriture "camel case", soit des underscores pour les espaces. Exemples d'utilisation du camel case et des underscores :

ceciEstUnNomDeVariable ceci_est_un_nom_de_variable

nomDuPersonnage

ageDuPersonnage

Comment peut-on nommer les variables ?

Règles de nommage strictes (contrôlées par l'interpréteur Python) :

- Ne pas commencer par un chiffre.
- Ne pas inclure d'autres caractères que des lettres majuscules et minuscules, les chiffres et le caractère "_".

Règles de nommage conventionnelles :

- Éviter les accents.
- Éviter les majuscules au début du nom des variables.
- Éviter les noms trop peu précis ou vagues.

Si la variable est destinée à garder la même valeur sans jamais changer :

- On peut l'écrire tout en majuscule.

Déclaration de variable

- À **gauche** de l'égalité se situe le **nom** de la variable.
- À **droite** de l'égalité se situe la **valeur** de la variable.
- Exemple A:

```
x  = 3
```

- Exemple B:

```
x = 8.4
y = 0.2
z = x + y
```

Valeur issue de l'évaluation de l'**expression** `x + y`

Que peut-on mettre “dans” une variable ?

La valeur d’une variable peut être un nombre, un texte, ou encore d’autres quantités dont on parlera plus loin dans le cours.

Le **type** de valeur que contient la variable est une donnée importante.

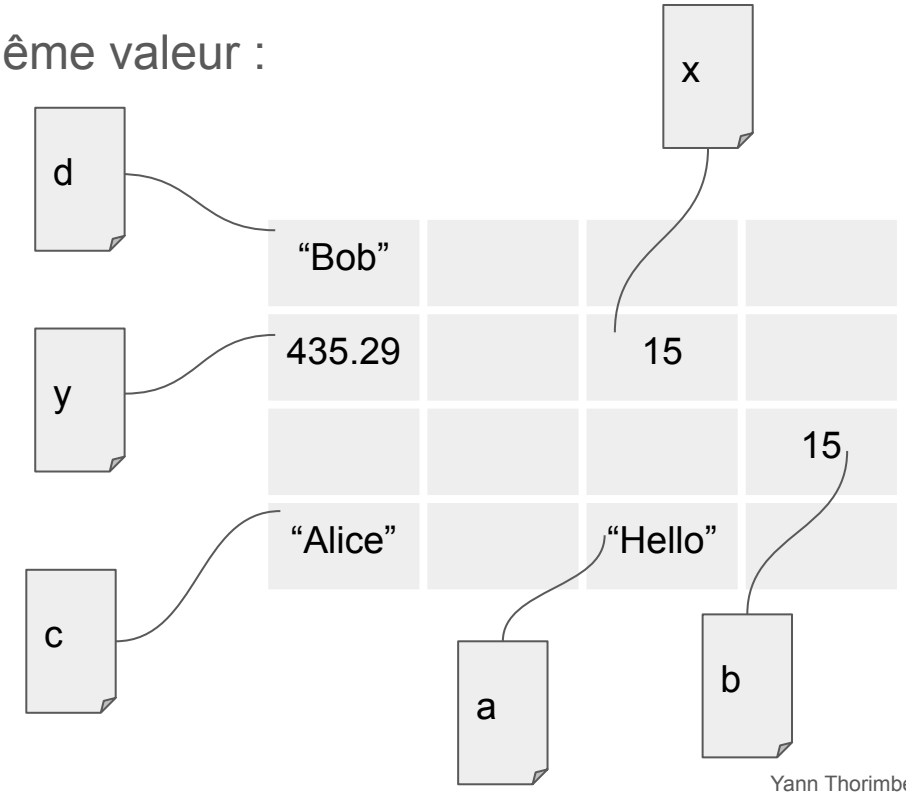
```
a = “Bonjour à tous”
b = 15
```

Dans le code ci-dessus, a et b ne sont visiblement pas du même type.

Que peut-on mettre “dans” une variable ? (suite)

Plusieurs variables peuvent avoir la même valeur :

```
a = “Hello”
b = 15
c = “Alice”
d = “Bob”
x = 15
y = 435.29
```

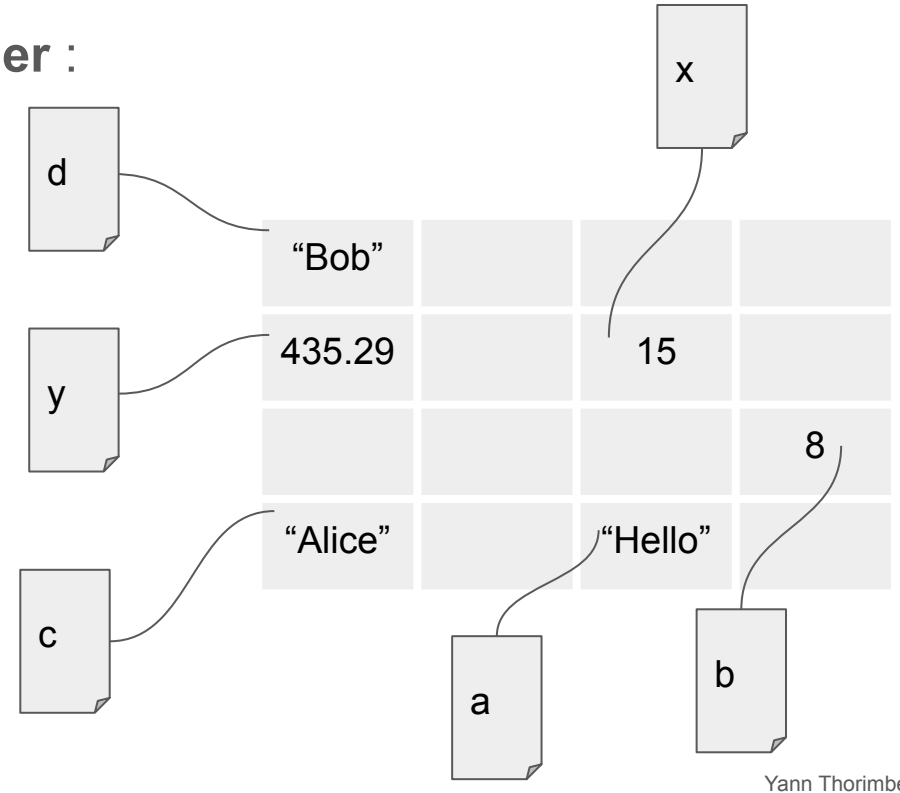


Que peut-on mettre “dans” une variable ? (suite)

La valeur d’une variable peut **changer** :

```
a = “Hello”
b = 15
c = “Alice”
d = “Bob”
x = 15
y = 435.29
b = 8
```

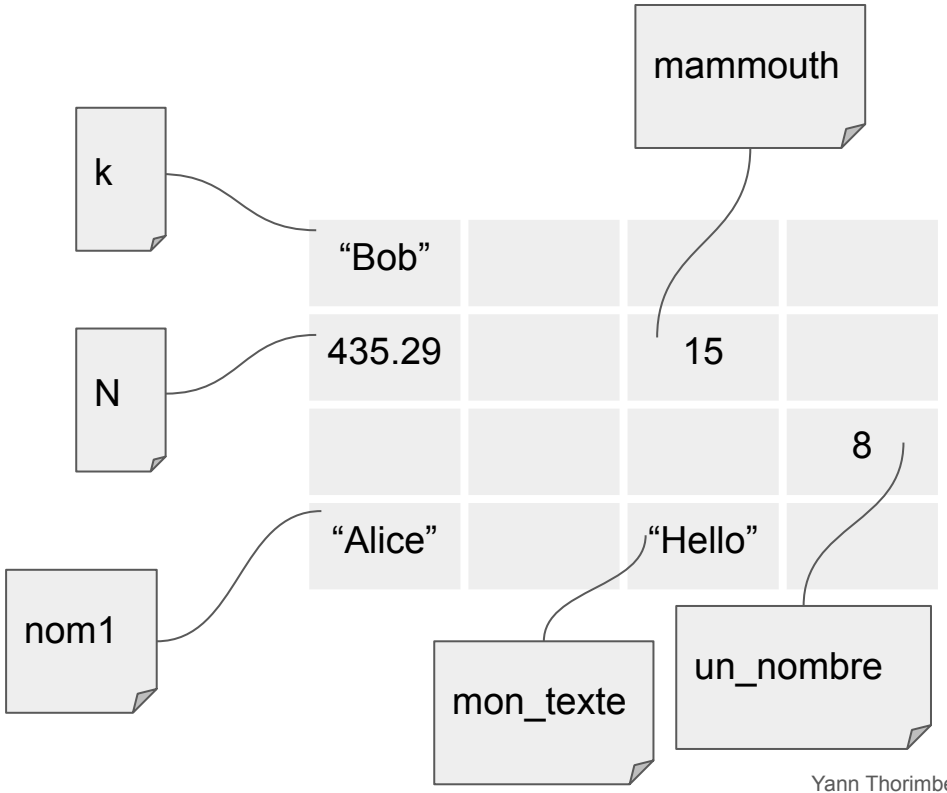
Ici, la variable b est modifiée dans la dernière ligne.



Que peut-on mettre “dans” une variable ? (suite)

C’est le programmeur qui **décide** du nom des variables.

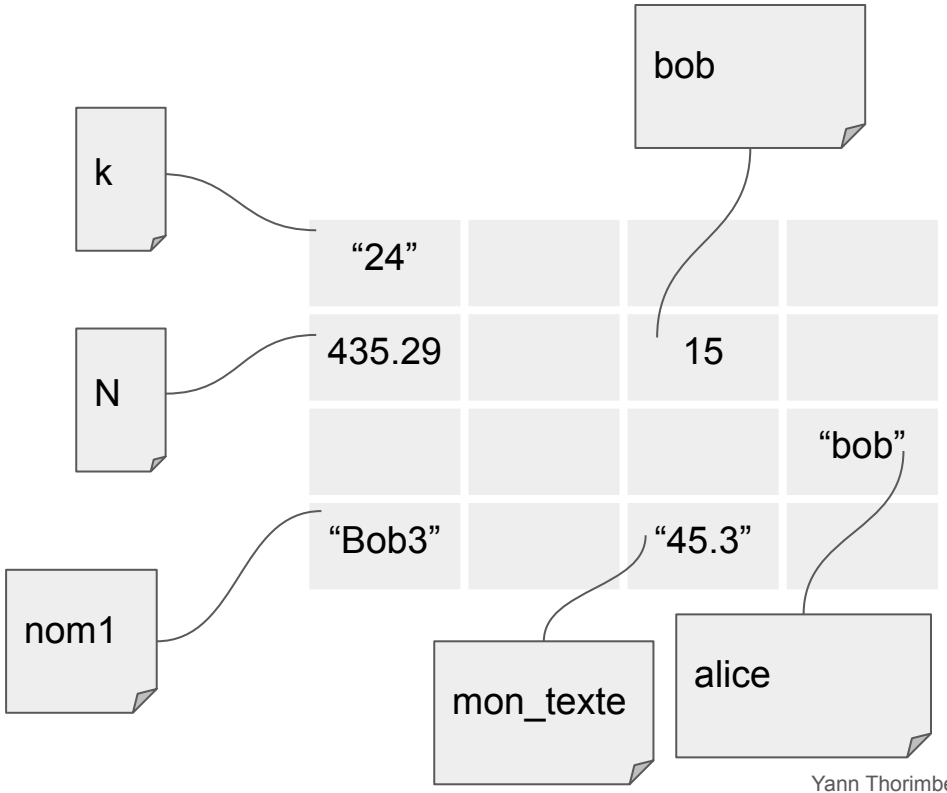
Dans l’exemple ci-contre, toutes les variables n’ont pas des noms très explicites, ce qui est à éviter !



Que peut-on mettre “dans” une variable ? (suite)

Attention, un texte (string) peut contenir des chiffres !

Texte : **chaîne** de caractères



Variables (rappel)

- Le nom de la variable est indiqué à gauche du signe d'égalité.
- La valeur de la variable est indiquée à droite du signe d'égalité.
- Exemples :

```
mon_nombre = 3.14;
un_autre = -12;
x = 2^4;
y = mon_nombre + un_autre;
z = sqrt(36);
```

Expressions

- Désigne ce qui peut être **évalué** pour en obtenir une valeur.
- Une variable est une expression.
- Un littéral est une expression qui est la représentation textuelle d'une valeur.
  Exemples : `3`, `"salut"`
- Une opération qui combine des variables et des littéraux pour retourner une valeur est elle-même une expression.
  Exemple : `y = x + 8`

  Valeur issue de l'évaluation de l'expression `x + 8`, où l'opérateur d'addition est utilisé en combinant la valeur de la variable `x` et d'un littéral.

Variables (suite)

- Le signe d'égalité ne s'entend pas comme en mathématiques !
- Le signe d'égalité s'entend comme "devient…", "se transforme en…", "prend la valeur de…".
- *Exemple A :*

```
x = 3;
x = x + 2;
disp(x) % ceci affiche 5
```

- *Exemple B :*

```
y = 4
y = 1
disp(y) % ceci affiche 1
```

Les variables prédéfinies

- En Matlab, certaines variables sont déjà définies.
  Exemples : `pi, i, …`

- Pour réinitialiser la valeur des variables, utiliser la commande (mot-clé) :
  `clear`

- Attention : par défaut, l'espace de travail (les valeurs en mémoire) est conservé d'une exécution à l'autre du script.
  Exemple : si vous écrivez pi = 8, cela affectera les exécutions futures du script !

Les types de variables vus dans ce cours

Type d'information	**Alias en Matlab**	**Exemples d'assignation**
Nombre entier relatif	`int8, ..., int64`	`x=int8(-5), y=int64(12)`
Nombre à virgule	`single, double`	`x=-2.76, y=single(3.141), z=5`
Texte	`string`	`x="Ciao!", y=":)", z="32", t=" "`
Booléen	`logical`	`x=1, y=0`

Par défaut, Matlab utilise le type double même pour les entiers !

NB : il faut différencier string et char, mais le sujet sera traité plus loin.

Obtenir le type d’une variable

- La fonction `class(x)` permet d’obtenir le type de `x`.
  Exemples :

```
a = -8;
disp(class(a)); % double
b = “Salut”;
disp(class(b)); % string (ou char en Octave)
```

- Quand on programme, il est impératif de connaître les types afin de comprendre les erreurs qui s’affichent.

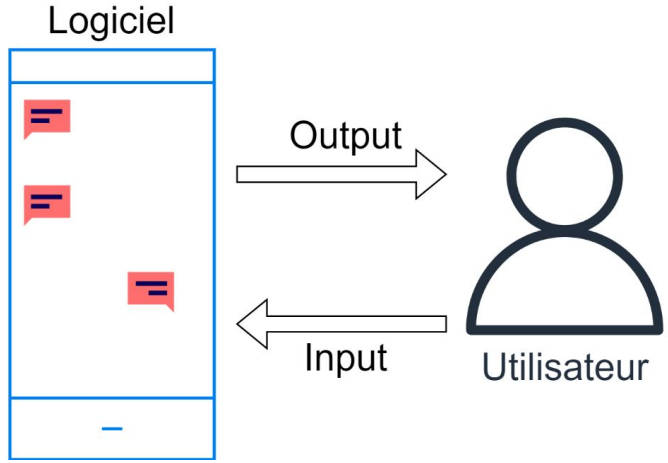
Input, output et rôle de l'utilisateur

- Lorsqu'on **utilise** une application ou un logiciel, le code du programmes nous est caché.

- L'utilisateur :
  - Reçoit des informations (output).

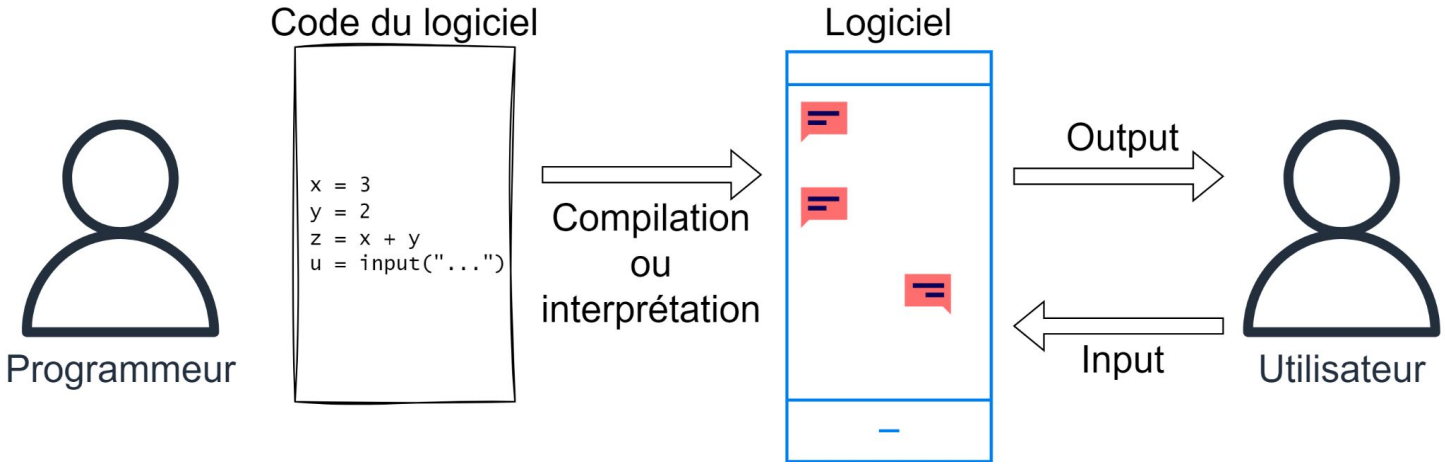
    Par exemple : lire un message affiché.
  - Envoie des informations (input).

    Par exemple : écrire un message.



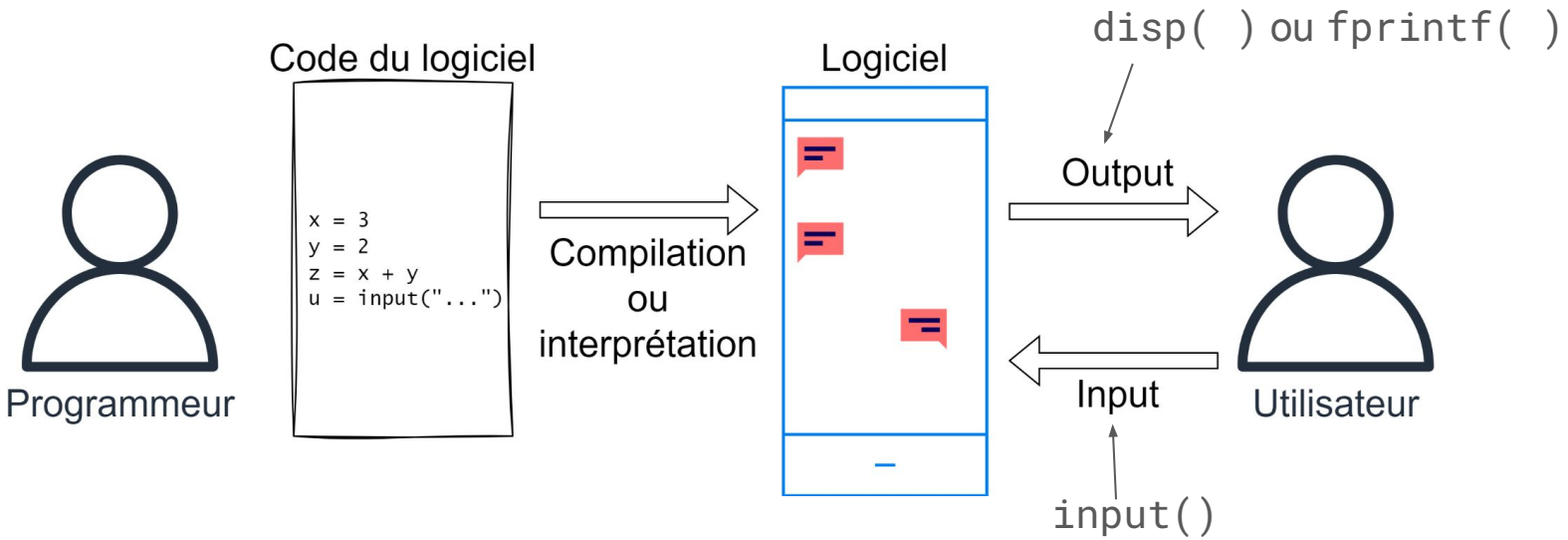
Input, output et rôle de l'utilisateur

- Lorsqu'on **crée** un logiciel, on doit faire **semblant** d'être l'utilisateur lorsqu'on le teste.



Input, output et rôle de l'utilisateur

- Lorsqu'on **crée** un logiciel, on doit faire **semblant** d'être l'utilisateur lorsqu'on le teste.



Input, output et rôle de l'utilisateur

- Certaines variables se voient assigner une valeur qui **dépend de l'input** de l'utilisateur et qui n'est pas connue au moment de l'écriture du programme.
  Exemple : `age = input("Entrez votre âge:");`

- D'autres variables ont une valeur décidée uniquement par le programmeur.
  Exemple : `age_minimum = 18;`

- Dans tous les cas, le type et le nom de la variable sont fixés par le programmeur.

Exemples d'input en Matlab

- Demander l'âge de l'utilisateur (nombre à virgule) :

```
x = input("Quel est votre âge ?");
x = int64(x); % si l'on veut absolument un entier
```

Exemples d'input en Matlab

- Demander l'âge de l'utilisateur (nombre à virgule) :

```
x = input("Quel est votre âge ?");
x = int64(x); % si l'on veut absolument un entier
```

- Demander une distance en kilomètres (nombre à virgule):

```
x = input("Quelle est la distance ?");
```

Exemples d'input en Matlab

- Demander l'âge de l'utilisateur (nombre à virgule) :

```
x = input("Quel est votre âge ?");
x = int64(x); % si l'on veut absolument un entier
```

- Demander une distance en kilomètres (nombre à virgule):

```
x = input("Quelle est la distance ?");
```

- Demander son nom à l'utilisateur (texte):

```
x = input("Quel est votre nom ?", "s");
```

Sert à indiquer que l'on veut récupérer du texte

Et l'output ?

- En Matlab l'output textuel vers la console est en général effectué via la fonction `disp()`, en omettant le point-virgule après une instruction, ou encore via la fonction `fprintf()`.

- En règle générale, les fonctions d'input et d'output dépendent des **périphériques** utilisés. Dans ce cours, on ne considère que les inputs et outputs affichés par l'écran au sein de la **console** (“Command Window”).

Exemple d'affichage de valeurs avec `fprintf()`

```
a = "Bonjour";
b = 12;
c = 2.718;
```

(coloration fantaisiste pour des raisons pédagogiques)

Exemple d'affichage de valeurs avec `fprintf()`

```
a = "Bonjour";
b = 12;
c = 2.718;

fprintf("Texte: %s\n", a);
fprintf("Entier: %d\n", b);
fprintf("Double: %f\n", c);
fprintf('Entier: %d, Double: %f, Texte: %s\n', b, c, a);
```

% se référer à https://ch.mathworks.com/help/simulink/slref/fprintf.html !

Exemple d'affichage de valeurs avec `fprintf()`

```
a = "Bonjour";
b = 12;
c = 2.718;

fprintf("Texte: %s\n", a);
fprintf("Entier: %d\n", b);
fprintf("Double: %f\n", c);
fprintf('Entier: %d, Double: %f, Texte: %s\n', b, c, a);
```

Dans ce cas, "%" n'est pas un commentaire !
C'est un **spécificateur de format**.

% se référer à https://ch.mathworks.com/help/simulink/slref/fprintf.html !

Exemple d'affichage de valeurs avec `fprintf()`

```
a = "Bonjour";
b = 12;
c = 2.718;

fprintf("Texte: %s\n", a);
fprintf("Entier: %d\n", b);
fprintf("Double: %f\n", c);
fprintf('Entier: %d, Double: %f, Texte: %s\n', b, c, a);
```

"\n" désigne un saut de ligne.

% se référer à https://ch.mathworks.com/help/simulink/slref/fprintf.html !

Affichage de valeurs (suite)

Affichage avec `disp` :

```
disp("Pi vaut environ:");
disp(d); % il existe d'autres façons de faire encore, cf. chapitre 6
```

Affichage avec `fprintf` :

```
fprintf("Pi vaut environ (défaut) %f\n", d);
fprintf("Pi vaut environ (8 décimales) %.8f\n", d);
fprintf("Pi vaut environ (2 décimales) %.2f\n", d);
```

Fonctions simples

- “Bout de code” qui peut être réutilisé depuis d’autres scripts.
- Effectue une suite d’instructions sur les paramètres d’entrée (arguments).
- Exemple de **déclaration** de fonction :

```
function val = cube(x)
    val = x * x * x;
end
```

- Exemple **d’utilisation** de fonction :

```
cube(3); % retourne 27
```

Fonctions simples

- En Matlab, **chaque fonction son fichier**.
- Par exemple, la fonction `cube()` définie plus haut est définie dans cube.m
- Si le dossier qui contient cube.m fait **partie de l'espace de travail**, la fonction `cube()` est disponible.
- Exemple d'utilisation depuis un autre fichier du même espace de travail :

```
x = input("Entrez un nombre:");
fprintf("Le cube de votre nombre vaut %f\n", cube(x));
```

- On reviendra plus en détail sur les fonctions (e.g. retours multiples) dans un autre chapitre.

Fonctions simples

```
function val = cube(x)
    val = x * x * x;
end
```

cube.m

Fait référence à du code écrit dans un autre fichier.

```
a = 3;
cube_a = cube(a);
```

monscript.m

Fonctions sans valeur de retour

- Exemple : fonction uniquement destinée à afficher un graphique.
- Dans ne nombreux langages, on parle de **procédure** dans ce cas.
- En Matlab :

```
function ma_fonction()
    disp(“Bonjour”)
end
```

Note sur la différence avec d'autres langages

En Matlab, on ne spécifie pas explicitement à l'aide du mot clé `return` que la fonction retourne une valeur, comme dans d'autres langages (C, Python, …)