Introduction

Lundi 15 septembre

Option Informatique

Ecole Alsacienne

Programme de la séance

1. Présentation du cours
2. Au programme de cette année
3. Un nouveau langage

1. Présentation du cours

Même fonctionnement que l'année dernière

- Une séance d'introduction
- Une alternance entre cours magistraux et travaux dirigés
- Salle 124 (à côté du laboratoire de physique)
- Un ou plusieurs devoirs sur tables
- Un ou plusieurs projets de programmation

Site internet mis à jour

- Site Internet dédié à ce cours :

  andre.lovichi.free.fr/teaching/ea/

- Contenu :
  - Calendrier des séances à venir
  - Slides présentés en cours
  - Sujets des TD
  - Aide-mémoire
  - Guide d'installation
  - Quelques logiciels utiles
  - Pistes pour approfondir

Toujours les mêmes règles

- Cours basé sur le volontariat :

  Vous avez choisi de participer à ce cours,
  donc si vous êtes là, c'est pour suivre.

- Evaluation:
  - Devoirs sur table
  - Quelques TD relevés à chaque séance
  - Assiduité et participation
  - Projets de programmation

- Absence / problèmes divers :

  Un prof prévenu à l'avance et à qui on fournit un justificatif
  est un prof plus conciliant.

Petite parenthèse électorale

- Le délégué de classe, c'est l'élève qui, plus que les autres, pourra dire :
  - Ca va trop vite / trop lentement !
  - Il y a une conférence / un voyage à Berlin / un bac blanc / une après-midi à la ferme la semaine prochaine
  - On n'a pas compris ce que vous nous avez raconté la dernière fois à propos de …
  - Vous avez (encore) oublié de mettre les slides en ligne.
  - Bob, il ose pas trop venir vous parler, mais en fait…
  - Monsieur, on a déjà trois devoirs pour ce jour là, y vraiment pas moyen de repousser un peu le projet ?

- Délégué(e) :

Choix de l'horaire

- Horaire défini pour l'instant : 17h00 – 18h45

- Autres horaires possibles
  - 17h15 – 19h00
  - 17h30 – 19h15
  - autre ?

- Horaire définitif :

2. Au programme de cette année

Séquence 1 : Python et Labyrinthes

Séance 2 : *A la découverte de Python 1/2*

- Manipulation des types de base
- Afficher une valeur
- Construction classiques :
  - Si… alors… (sinon…)
  - Pour i allant de 0 à n, faire …
  - Tant que …, faire …
- Description des erreurs les plus fréquentes

Séquence 1 : Python et Labyrinthes

Séance 3 : *A la découverte de Python 2/2*

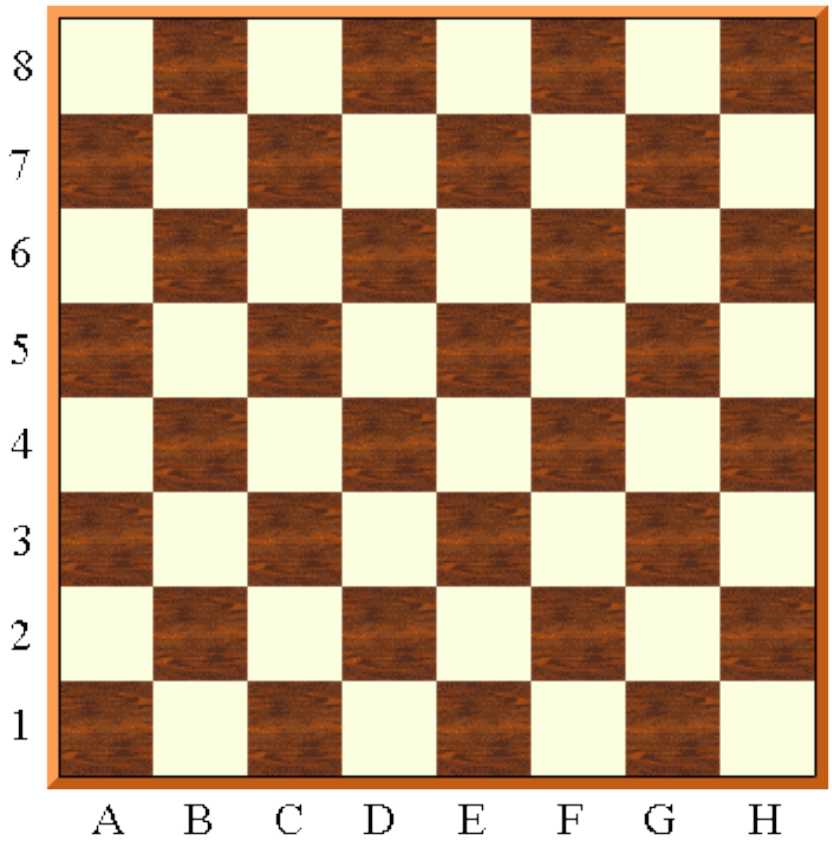
- Lire une entrée de l'utilisateur
- Fonctions
- Tableaux
- Chaînes de caractères

Séance 4 : Structures multidimensionnelles

- Tableaux et dimensions
- Représentation d'une grille
- Le problème des huit dames
- Représentations possibles d'un labyrinthe

Séquence 1 : Python et Labyrinthes

Comment placer huit dames sur un échiquier sans qu'aucune ne soit en prise avec une autre ?



▪▪▫ Au programme de cette année

Séquence 1 : Python et Labyrinthes

Séance 5 : *Labyrinthes simples*

- Affichage d'un labyrinthe
- Génération aléatoire d'un labyrinthe (case par case)
- Génération aléatoire d'un labyrinthe (mur par mur)

Le retour du troll

Début de la saison 2 (en Python)

Séquence 2 : Graphes

- Premiers exemples et définitions formelles

- Premières propriétés
  - Calculs sur les degrés
  - Connexité
  - Distances dans un graphe

- Problèmes célèbres
  - Parcours eulériens et hamiltoniens
  - Graphes planaires
  - Calcul du plus court chemin
  - Coloration

Séquence 2 : Graphes

- On veut organiser des matches entre 5 joueurs d'échecs.
- **Question** : Comment s'arranger pour que chacun joue contre exactement 3 adversaires différents ?
- **Réponse** : C'est impossible !
- **Vraie question** : Pourquoi ?

Séquence 2 : Graphes

- Une soirée est organisée entre 4 personnes :

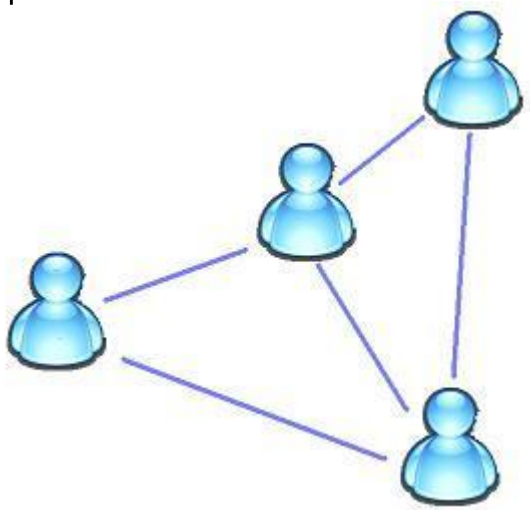
  Albert, Béatrice, Charles et Denise

- On considère que si X connait Y, alors Y connait X.

- **Question** : Y a-t-il forcément deux invités qui connaissent le même nombre de personnes ?

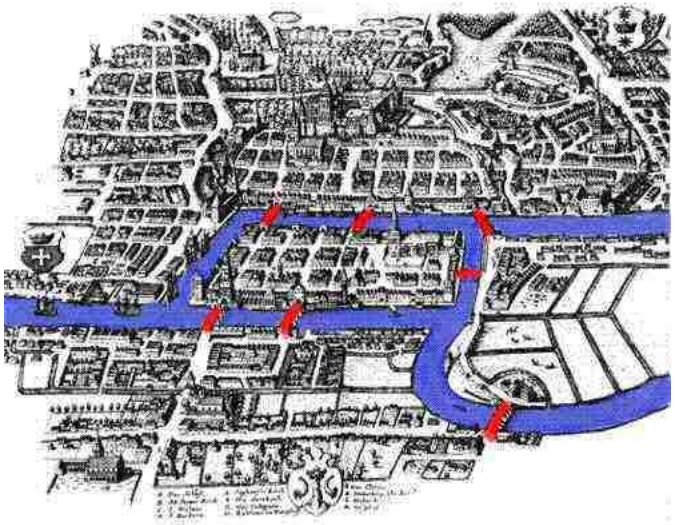
- **Réponse** : Oui, forcément !

- **Vraie question** : Pourquoi ?



Séquence 2 : Graphes

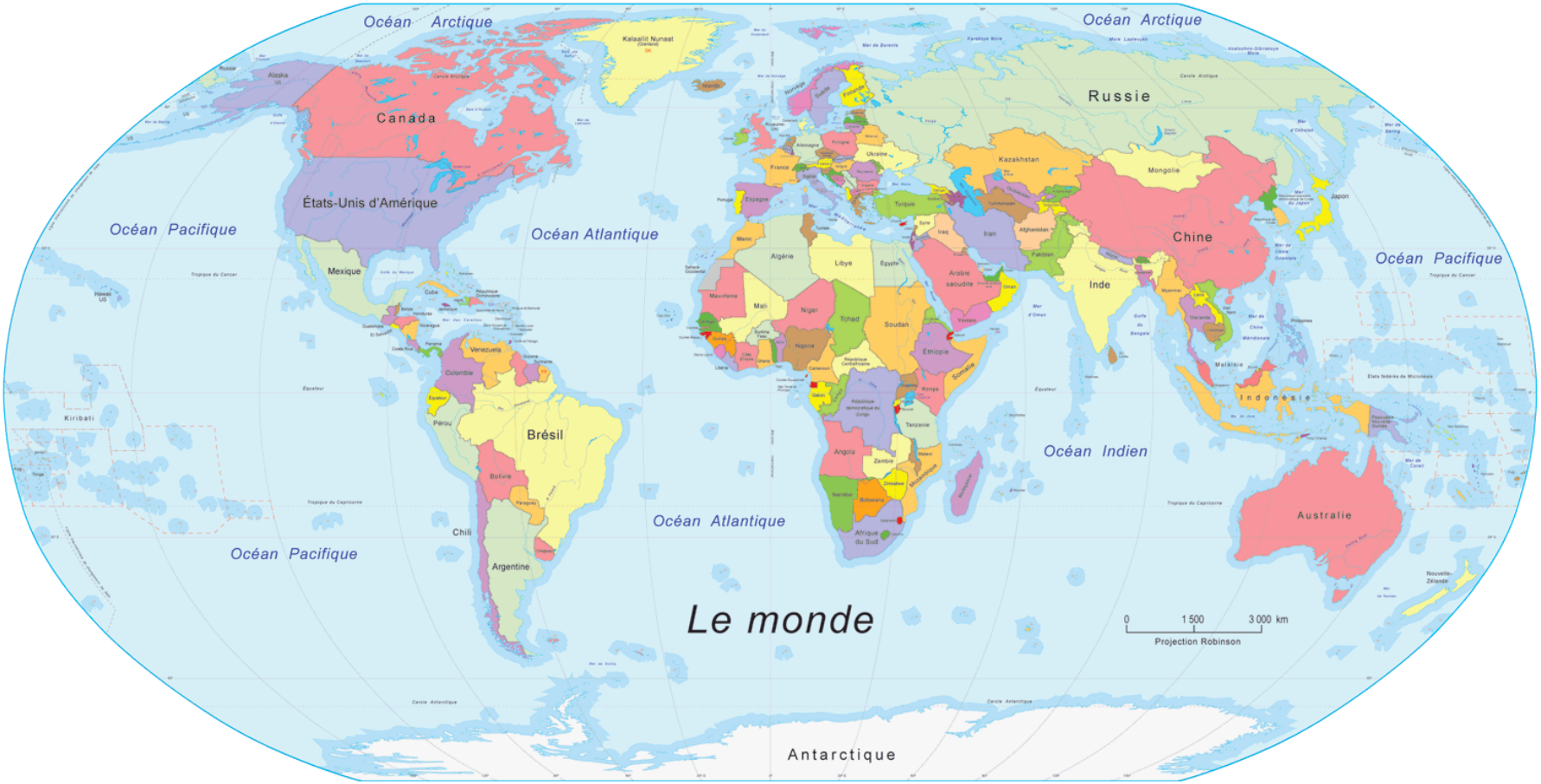
- Les 7 ponts de Königsberg



- Quatre zones géographiques
  - Deux rives
  - Deux îles
- **Questions** :
  - Un habitant peut-il faire un tour de la ville et passer exactement une fois par chacun des ponts ?
  - Même sans revenir à son lieu de départ ?

SÉQUENCE 2 : GRAPHES

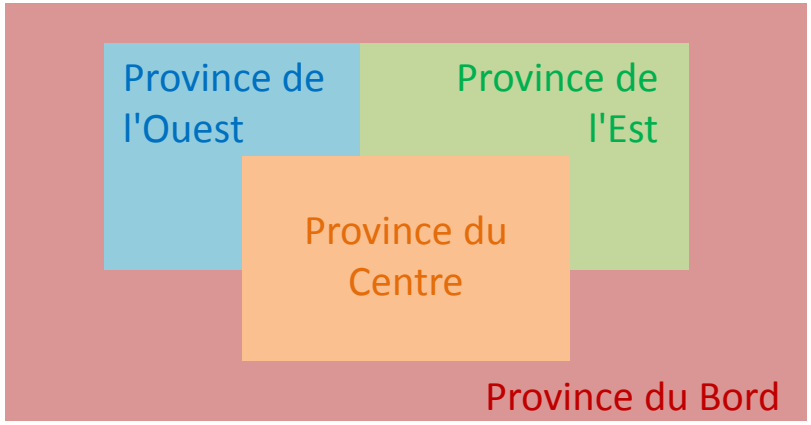
Combien de couleurs faut-il pour tracer une carte sans que deux pays voisins n'ait jamais la même couleur ?



Séquence 2 : Graphes

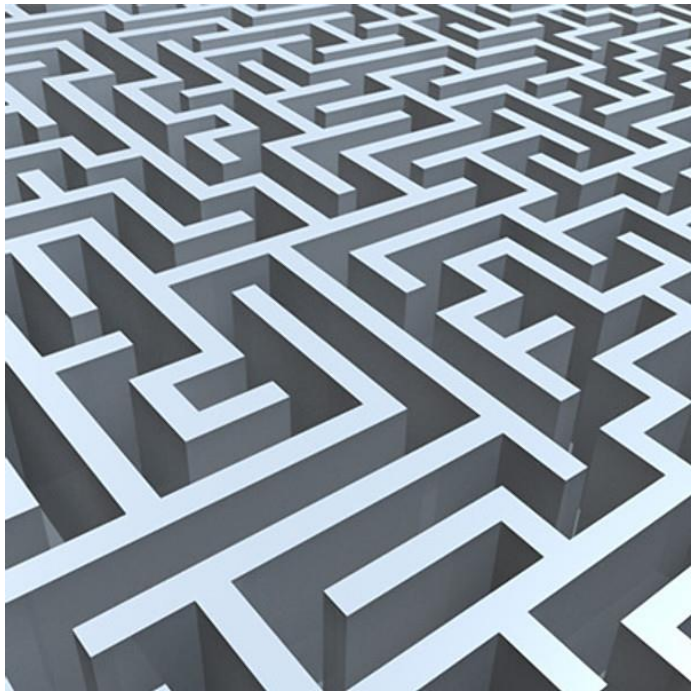
- **Défi** : Tracer une carte du Continent des Cinq Royaumes
  - Un continent composé de 5 pays
  - Chacun pays est voisin des 4 autres (en plus d'un point)

- **Exemple** : La Terre aux Quatre Provinces



Séquence 2 : Graphes

Comment trouver le plus court chemin
vers la sortie dans un labyrinthe ?



Séquence 3 : Automates et langages rationnels

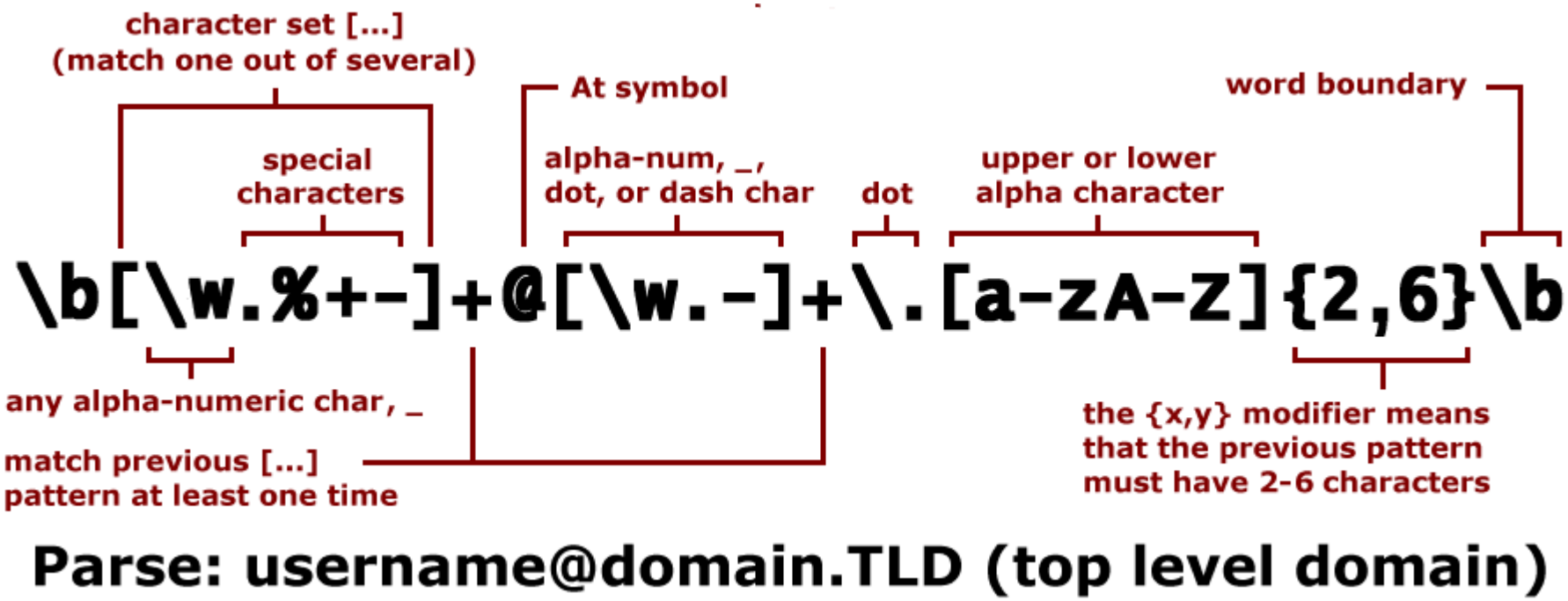
- Alphabets et langages
- Automates finis déterministes
- Expressions rationnelles
- Transformations classiques sur les automates
  - Déterminisation
  - Minimisation
- Lien entre automates et expressions rationnelles

SÉQUENCE 3 : AUTOMATES ET LANGAGES RATIONNELS

Comment vérifier la présence d'un motif dans un texte ?

SÉQUENCE 3 : AUTOMATES ET LANGAGES RATIONNELS

Comment caractériser le format d'une adresse email valide ?



SÉQUENCE 3 : AUTOMATES ET LANGAGES RATIONNELS

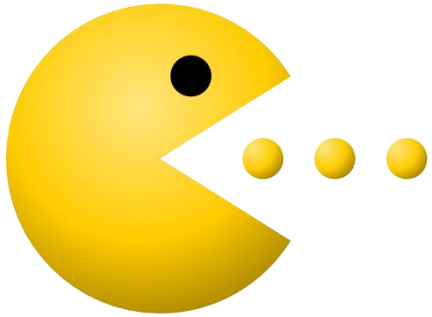
Comment un ordinateur fait-il pour vérifier la syntaxe d'un programme ?

```
def minimum(a,b):
    if a < b:
        return a
    else
        return b


line 4:
else
^
SyntaxError: invalid syntax
```

Séquence 4 : Projet de programmation

- A définir ensemble…
  - Un projet pour la classe ou plusieurs projets ?
  - Travail individuel ou par groupe de 2 ou 3 ?
  - Quelques positions de sujets :



Séquence 5

Interface graphique

Programmation objet

Réseaux

A définir ensemble…

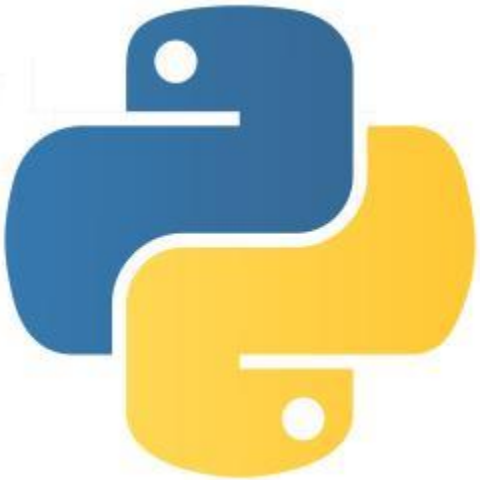
Web

Robotique

3. Un nouveau langage

UN NOUVEAU LANGAGE

Cette année, nous travaillerons en Python !



Depuis 2013, c'est le langage de référence pour la matière « informatique pour tous » en classe préparatoire

WHAT IS YOUR FAVORITE LANGUAGE ?

- Langage créé à la fin des années 90 par Guido van Rossum

- Nom tiré des Monty Python

Avantages de Python

- Conçu pour être lisible
- Syntaxe simple (tout en imposant un minimum de rigueur)
- Gestion automatique de la mémoire (garbage collector)
- Quelques entreprises qui utilisent Python :
  - Google
  - NASA
  - Blender
  - etc.

Et Caml dans tout ça ?

Toujours utilisé en classe préparatoire !
(mais plutôt pour l'option informatique)

VARIABLES

- Déclarer une variable
  - Syntaxe : `nom = valeur`
  - Exemple : `age = 17`

- Accéder à la valeur d'une variable
  - Syntaxe : `nom`
  - Exemple : `age`

- Remarque : Pour afficher à l'écran la valeur d'une variable, on utilise la fonction `print`
  - Exemple : `print(age)`

```
34
```

Variables

- Affectation d'une nouvelle valeur
  - Syntaxe : `nom = valeur` (la même que pour une déclaration)
  - Exemple : `age = 18`

- Remarque : Vous n'avez plus besoin de passer par des références pour modifier une variable
  - Fini les complications avec `ref`, `:=` et `!`
  - Par contre, attention à ne pas écraser des données !

- Remarque : En Python, une variable peut changer de type !

```
x = 3
x = coucou
```

Types

- Il existe différents types de base dans Python
  - `int` : Entiers
  - `float` : Nombre réels
  - `str` : Chaîne de caractère
  - `bool` : Booléens
  - `complex` : Nombres complexes
  - `NoneType` : aucun type (l'équivalent de `unit` ou `void`)

- Remarques :
  - Le type `char` n'existe pas : un caractère est simplement une chaîne de caractères (`str`) de longueur 1
  - Il existe d'autres types (ex : `long`, `dict`, etc.)

Opérateurs

- En Python, il n'y a pas d'opérateurs dédiés à la manipulation des nombres réels
  - Finies les erreurs avec les "opérateurs avec un point"

- Pour les divisions, on distingue trois opérateurs
  - `/` : division "réelle" (nombre réel)
  - `//` : quotient de la division euclidienne (nombre entier)
  - `%` : reste de la division euclidienne (nombre entier)

- Les opérateurs booléens s'écrivent en toutes lettres :
  - `or` : OU logique (disjonction)
  - `and` : ET logique (conjonction)
  - `not` : NON logique (négation)

Séquence d'instructions

- En Python, pour effectuer plusieurs instructions, il suffit de passer à la ligne :

```
instr1
instr2
…
instrN
```

- Exemple :
  - Version OCaml :

```
print_string "age=" ; print_int 3 ;
```

  - Version Python :

```
print("age=")
print(3)
```

  - Finies les erreurs de points virgules !

- ⚠ La mise en forme est très importante ! ⚠

BOUCLE FOR

- Syntaxe classique

```
for i in range(depart, arrivee):
    inst1
    inst2
    …
    instN
```

- Le contenu de la boucle for est caractérisé par son indentation (décalage d'une tabulation)

- ⚠ L'indentation est très importante ! ⚠

BOUCLE FOR

- Variante 1 : boucle de 0 à n

```
for i in range(arrivee):
    inst1
    inst2
    …
    instN
```

- Variante 2 : choisir le pas

```
for i in range(depart, arrivee, pas):
    inst1
    inst2
    …
    instN
```

Boucle While

- Syntaxe

```
while cond:
    inst1
    inst2
    …
    instN
```

- Attention aux boucles infinies

- ⚠ L'indentation est très importante ! ⚠

Conditions : If ... Then ... Else ...

- Syntaxe

```
if cond:
    inst1
    inst2
    …
    instN
else:
    inst1
    inst2
    …
    instP
```

- ⚠ L'indentation est très importante ! ⚠

Conditions : If ... Then ...

- Variante : sans le `else`

```
if cond:
    inst1
    inst2
    …
    instN
```

- Remarque : En Python, Il n'existe pas de `switch … case` (`match … with` en Caml)

FONCTIONS

- Syntaxe

```
def nomDeLaFonction(arg1, arg2, …, argN):
    inst1
    inst2
    …
    instN
```

- ⚠ L'indentation est très importante ! ⚠

- Remarque : sauf indication contraire, une fonction ne renvoie rien
  - Elle peut afficher quelque chose
  - Mais aucun résultat exploitable par une autre fonction

Le mot-clef return

- Syntaxe

```
def nomDeLaFonction(arg1, arg2, …, argN):
    inst1
    inst2
    …
    instN
    return resultat
```

- Le mot clef `return` permet de renvoyer un résultat, qui peut être utilisé plus loin dans le code

Le mot-clef return

- Exemple 1 :

```
def minimum(a,b):
    if(a<b):
        return a
    else:
        return b

x = minimum(3,4)
print(x * 2)
```

```
6
```

Le mot-clef return

- Exemple 2 :

```
def minimum(a,b):
    if(a<b):
        print(a)
    else:
        print(b)

x = minimum(3,4)
print(x * 2)
```

3

```
    print(x * 2)
TypeError: unsupported operand type(s) for *: 'NoneType' and 'int'
```

L'IMPORTANCE DE LA MISE EN FORME

- Une chose à retenir

  ⚠ L'indentation est très importante ! ⚠

- Les séquences d'instructions sont organisés selon les passages à la ligne

- La plupart des constructions classiques se servent de l'indentation pour déterminer les blocs de code à traiter :
  - conditions
  - boucles `for`
  - boucles `while`
  - corps d'une fonction

L'IMPORTANCE DE LA MISE EN FORME

- Exemple 1

```
if(3 > 2):
print("A")
else:
print("B")
```

```
 print("A")
        ^
IndentationError: expected an indented block
```

L'IMPORTANCE DE LA MISE EN FORME

- Exemple 2

```
if(3 > 2):
    print("A")
else:
    print("B")
print("B2")

A
B2
```

L'IMPORTANCE DE LA MISE EN FORME

- Exemple 3

```
if(3 > 2):
    print("A")
   else:
   print("B")
   print("B2")

 else:
       ^
IndentationError: unindent does not match any outer
indentation level
```

L'IMPORTANCE DE LA MISE EN FORME

- Exemple 4

```
for i in range(3):
    print("Je dois faire attention à l'indentation.")
print("Sinon mon code ne fera pas ce que je souhaite")
```

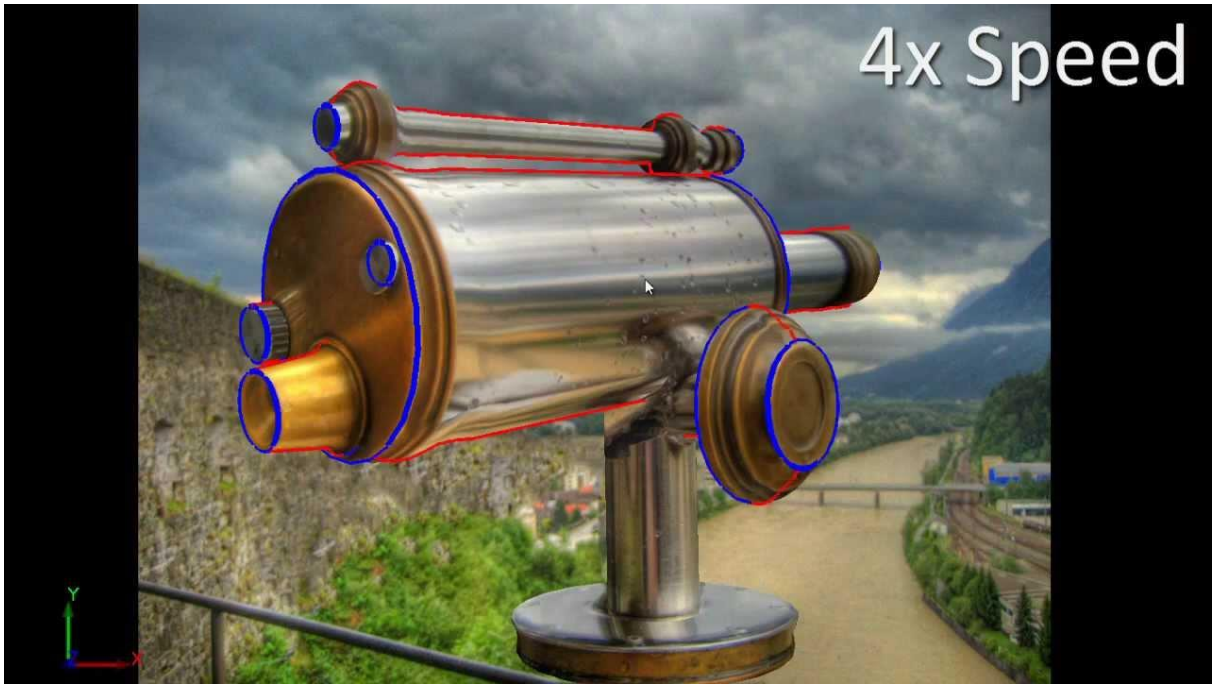
```
Je dois faire attention à l'indentation.
Je dois faire attention à l'indentation.
Je dois faire attention à l'indentation.
Sinon mon code ne fera pas ce que je souhaite
```

- Exemple 5

```
for i in range(10):
print("Je dois faire attention à l'indentation.")
print("Sinon mon code ne fera pas ce que je souhaite")
```

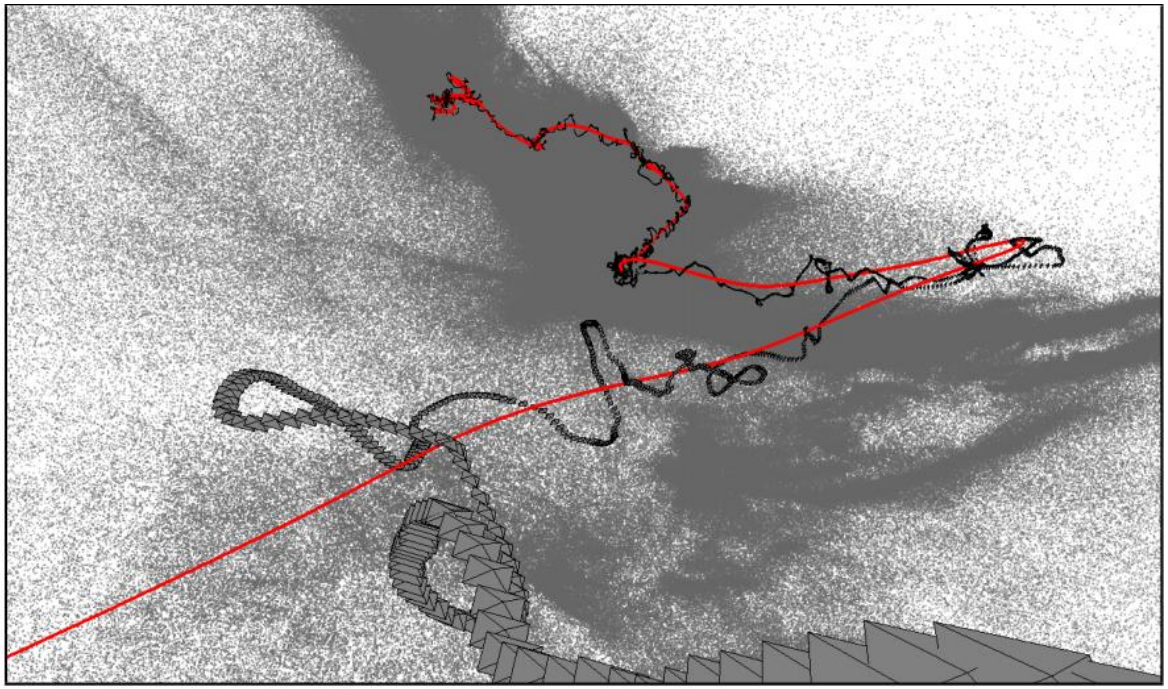
```
 print("Je dois faire attention à l'indentation.")
       ^
IndentationError: expected an indented block
```

POUR FINIR…



Modélisation 3D en temps réel

POUR FINIR…



Hyperlapse

POUR FINIR…

Leap Motion + Oculus Rift

POUR FINIR…

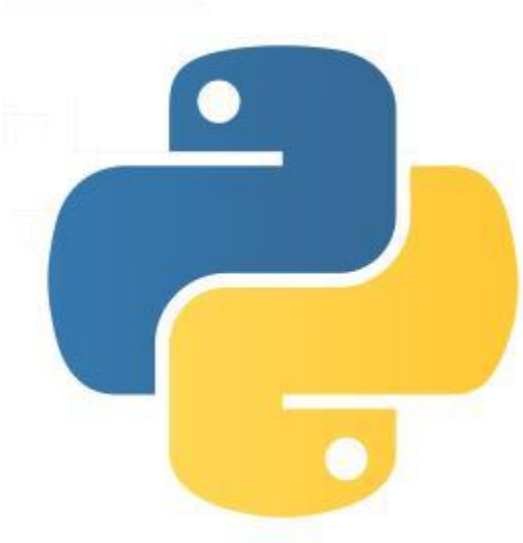
- Connaissez-vous Eugene Goostman ?

- Test de Turing :
  - Une intelligence artificielle (agent conversationnel)
  - Indiscernable d'un interlocuteur humain (pour un juge humain)
  - Juin 2014 : test réussi ?

PROCHAINE SÉANCE

Lundi 22 septembre

A LA DÉCOUVERTE DE PYTHON



⚠ L'indentation est très importante ! ⚠