EPFL

ICC

Programmation

CGC/SIE, **Cours 1**

19 février 2024

Jean-Philippe Pellet

Présentation

- Jean-Philippe Pellet
- Informatique à l'EPFL (2006)
- Doctorat en statistique/machine learning
  à l'ETHZ et IBM Research, Zurich (2010)
  - Mais vraie passion: les langages de programmation
- Actuellement:
  - HEP Vaud à Lausanne,
    développement (logiciels pédagogiques et didactiques)
    didactique de l'informatique & recherche

Partie Programmation d'ICC

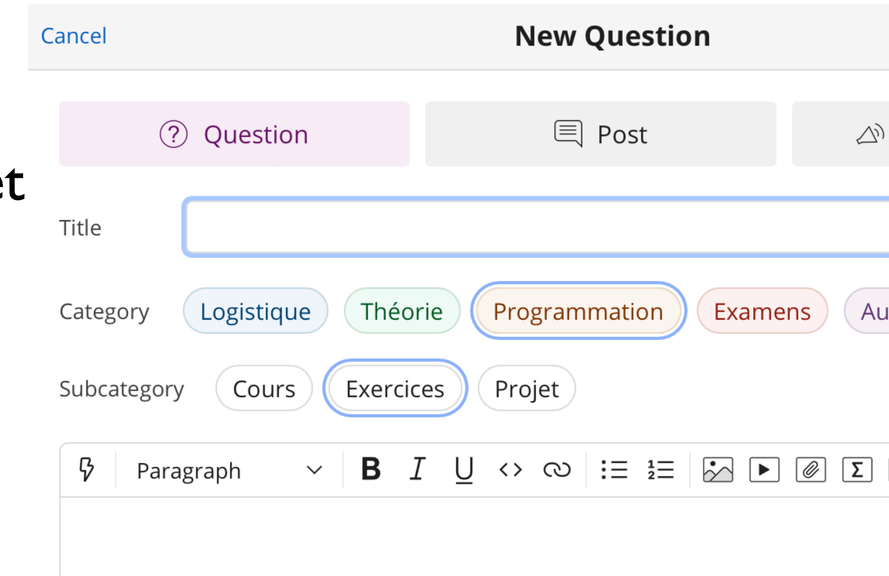
- Approche pragmatique: *make things work*
- Peu de théorie, davantage de pratique: programmation comme *outil*

- Cible:
  - Concepts les plus importants en programmation
  - Bases du langage Python
  - Débrouillardise et «savoir-chercher»

- On commence de zéro… mais on avance assez vite

Documents de cours

- Toutes les informations et liens vers les documents de cours sont sur **Moodle**, cours CS-119(k)
  - https://moodle.epfl.ch/course/view.php?id=15817
- Tous les cours sont diffusés en live via **Zoom**
  - **Enregistrement** à retrouver sur Moodle après
  - Mais… venez au cours autant que possible!
- Pour vos questions: **Ed Discussion**
  - Posez vos questions sur le cours, les exercices, le miniprojet
  - Catégorisez votre question correctement



Programme du cours

8 avril

Examen intermédiaire programmation

15%

Session d'examen

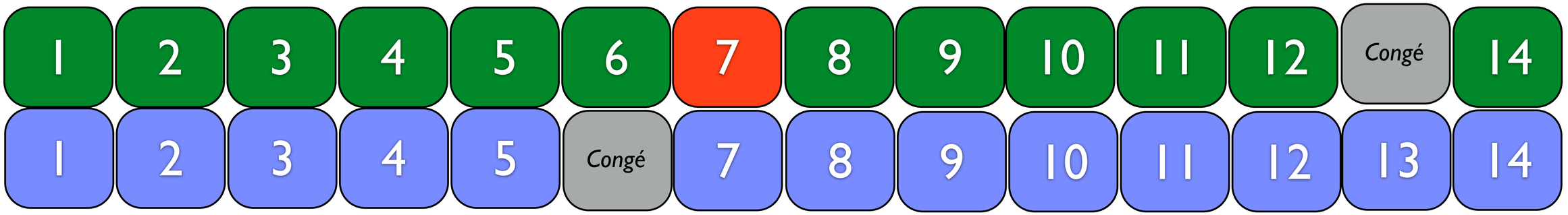
Examen final

75%

Cours & séries, partie programmation

10%

*Cours, séries & **mini**projet, partie programmation*



Cours & séries, partie théorique

Cours & séries, partie théorique

*L'examen final porte sur la partie théorique (⅔) **et** la partie programmation (⅓)*

Exercices

- **Méthode recommandée:** postes de travail virtuels
  - Directement en salles **CO020, CO021, CO023**
  - Sur votre machine via https://vdi.epfl.ch
  - Machine virtuelle: **IC-CO-IN-SC-INJ-2024-Spring** (Linux)
- **Autre possibilité:** installation personnelle de Python et Visual Studio Code sur votre propre machine
  - Nous vous aidons pour l'installation
  - Batterie et état de marche de votre machine: *votre responsabilité*
  - Je n'ai pas testé toutes les configurations…

La programmation

Programmation

l'ordinateur:

la machine universelle du monde de l'information

la programmation:

une technique de communication structurée avec la machine;
l'art d'exprimer de façon élégante un processus

l'approche computationnelle:

un «nouveau» moyen d'opérer en tant que scientifique

acquisition de données (capteurs)

simulation de modèles
(météo, cerveau, …)

traitement des signaux

interfaces graphiques

calcul numérique

automatisation de tâches répétitives

bases de données (enquêtes)

Python — un langage de programmation

«Grammaire»
Syntaxe

«Vocabulaire»
API/Bibliothèques

Comment mettre
les mots ensemble
de manière correcte

**Ensemble de
règles à savoir**

Quels sont les mots qui ont un sens particulier
dans ce langage, et que signifient-ils exactement?

**En Python: énorme volume de
bibliothèques**, à apprendre petit à
petit en fonction des besoins

Exemples de bibliothèques: communication réseau,
manipulation d'images, cryptographie, machine
learning, manipulation de code, etc., etc.

Ce cours: syntaxe de base de Python; exploration de la bibliothèque standard de
Python et outils pour rechercher et utiliser des bibliothèques existantes

Écrire un programme

- Comme une recette (que vous écrivez)
  - Une série d'instructions à exécuter dans un certain ordre
  - Une instruction se composent de mots du «vocabulaire» du langage, assemblés selon la syntaxe du language («grammaire»)
  - *L'ordre des instructions est bien sûr important!*

Faire des calculs en Python

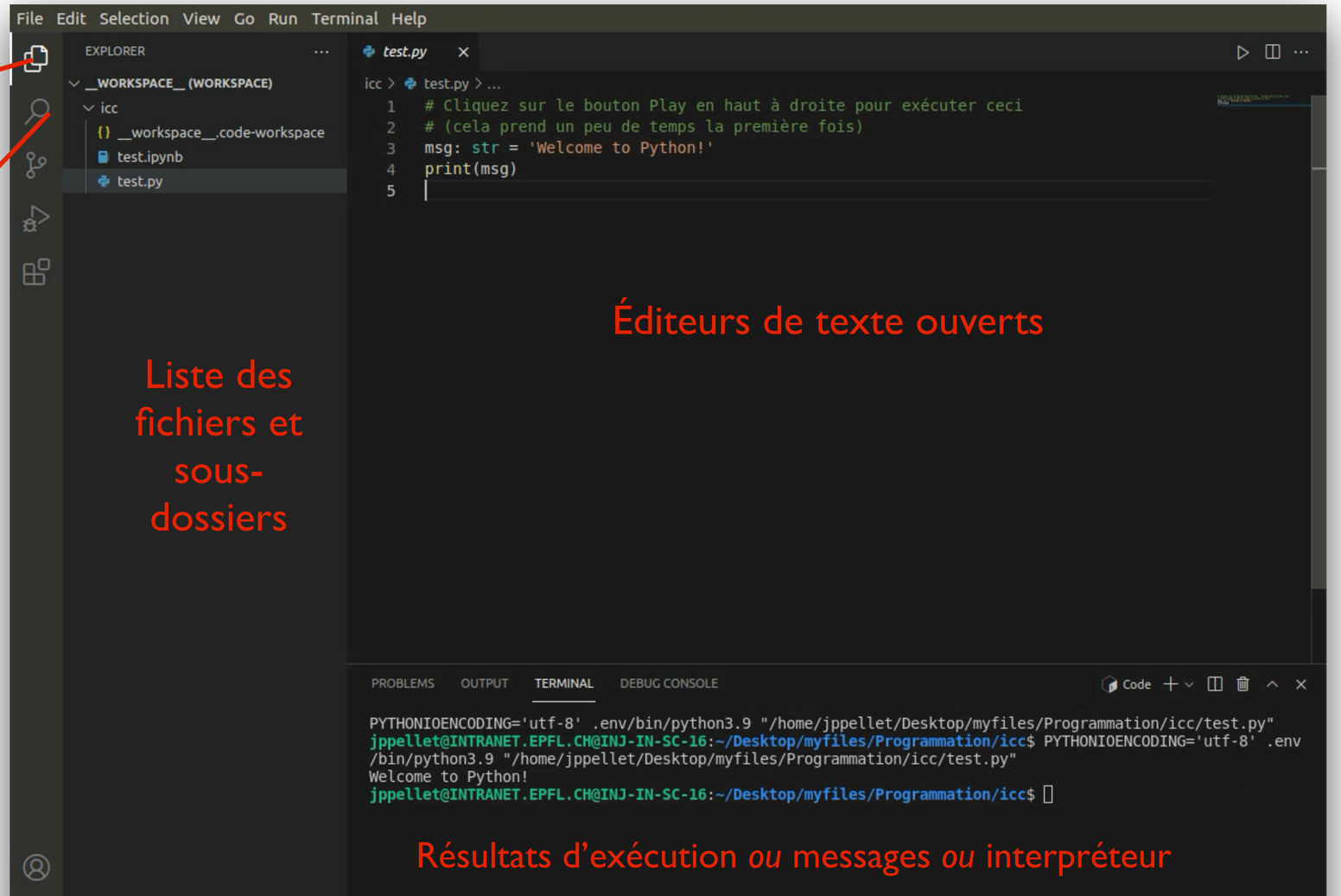
Démo

```
side = 4
area = side * side

print(area)
```

Travail avec Visual Studio Code

- *Première fois:* suivre consigne des exercices pour créer un *workspace*
- *Chaque fois:* File → Open Workspace…, puis sélectionner *icc_prog.code-workspace*

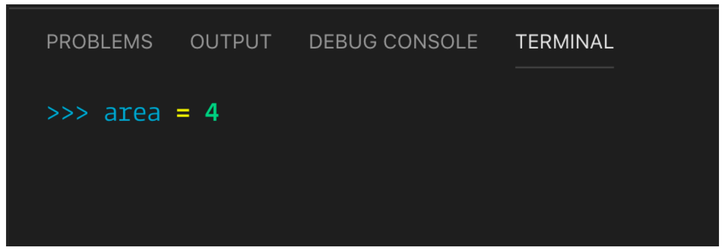


Faire tourner du code

Interpréteur

Menu ***Terminal → New Terminal*** *→ tapez* ***python3***

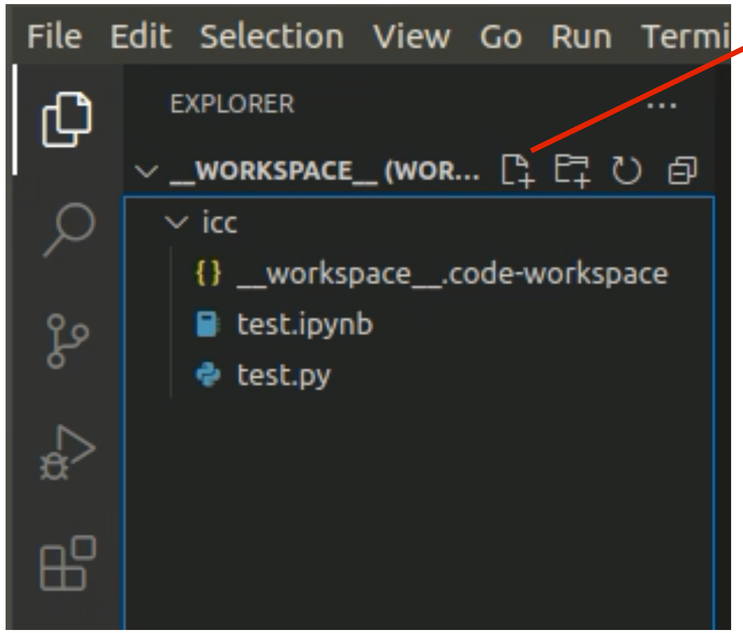
Puis: *taper des lignes dans le terminal qui s'ouvre en bas à droite*



Terminez par ***Ctrl-D***

Exécution ***ligne par ligne***

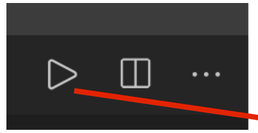
Via fichier



Bouton *New File…*

Puis: donner un nom qui se termine par *.py*

Puis: *éditer le fichier dans l'éditeur qui s'ouvre à droite*



Bouton *Run* en haut à droite

Exécution de ***tout le fichier***

Faire des calculs en Python

Variante:

```
side: int = 4
area: int = side * side

print(area)
```

- Déclarer les types est en principe optionnel en Python
- Mais: faites-le autant que vous pouvez
  - Plus de vérifications par le compilateur (ou linter)
  - Code plus expressif et plus facile à (re)lire pour vous
  - Force à mieux réfléchir à ce qu'on écrit

Interprétation de ces trois lignes

```
side: int = 4
```

nom *type* *valeur*

«Prends un bout de mémoire, rappelle-toi que je vais y faire référence avec le nom `side`*, et stockes-y la valeur 4, sachant que c'est un nombre entier»*

```
area: int = side * side
```

«Prends un autre bout de mémoire, que je vais appeler maintenant `area`*, et stockes-y le produit de ce qui est dans l'emplacement mémoire qui s'appelle* `side` *avec lui-même, sachant que le résultat est un nombre entier.»*

```
print(area)
```

«Imprime sur la console ce qu'il y a dans l'emplacement mémoire que j'ai appelé `area`*.»*

Représentation de la machine


```
side: int = 4
```

```
area: int = side * side
          = 4    * side
          = 4    *  4
          = 16
```

Les principaux types de données

```
side: int = 4

length: float = 3.5

my_name: str = "Jean-Philippe"

my_last_name: str = 'Pellet'
```

int — Un **nombre entier**

float — Un **nombre à virgule**

str — Du **texte**; une *chaîne de caractères* (string = chaîne), définie avec " ou ' au début et à la fin

```
# Commentaire
```

Pas un type de données, mais un **commentaire** à vous, pas interprété comme code

Conversion float/int/String

On ne peut pas faire toutes les opérations avec tous les types, mais on peut convertir d'un type à un autre

```
# Conversion depuis un int vers un float ou vers du texte
some_int: int      = 34
some_int_as_float  = float(some_int) # 34.0, mais souvent inutile en Python
some_int_as_string = str(some_int)   # "34"

# some_int_as_string + 2, ne marche pas! On ne peut pas ajouter un nombre à du texte
# some_int_as_string + "2", on peut “additioner” deux str: concaténation

# Conversion depuis un float
import math
some_float: float       = 0.182
some_float_rounded_up   = math.ceil(some_float)  # 1
some_float_rounded_down = math.floor(some_float) # 0
some_float_as_int       = int(some_float)        # 0
```

Fonctions, valeurs dérivées

```
math.ceil(some_float)
print(some_variable)
```

Forme générique: *NomDeFonction*(*argument*)

Ressemble à des choses bien connues… $\sin(\alpha)$, par exemple fonction dans Excel

Pas la seule manière d'obtenir des ***valeurs dérivées*** *dans Python*

Autres manipulations utiles

```
my_string = "programmation"
# Vous choisissez le nom de la variable;
# la valeur est toujours entre "" ou ''

# la fonction len() retourne la longueur d'un string
length = len(my_string)

# la méthode upper() s'écrit après un point et
# crée une version tout en majuscules de la valeur
# indiquée avant le point
my_string_upper = my_string.upper()

# le slicing (indexage d'une variable entre [])
# permet d'extraire une partie du string
my_substring = my_string[1:4]
```

Se documenter sur Python

«**python** convert int to string»
«**python** get string length»
«**python** check if string contains other string»

http://stackoverflow.com/

Site spécialisé en programmation, questions avec réponses triés par ordre de pertinence selon votes de la communauté.

Vérifiez bien que votre question n'a pas encore été posée avant d'en poser une — c'est très probable que quelqu'un ait déjà eu votre problème!

Résumé Cours 1

- Python est un langage moderne avec une syntaxe minimale
- VS Code est un IDE pour Python (notamment) qui permet d'éditer les fichiers et d'exécuter le programme
- L'interpréteur permet de facilement tester de petits bouts de code
- En Python, on peut déclarer le type des variables. Les types aident à vérifier que le programme est correct
- Des notations précises permettent de calculer de nouvelles valeurs (*fonctions, méthodes, slicing — on en reparlera!*)

Les assistant·e·s

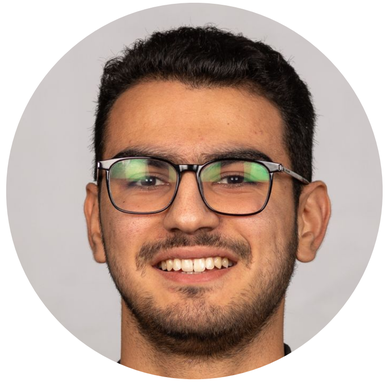
Puck van Gerwen
doctorante

Amer Lakrami

Axel Giboulot

Adèle Soulier

Fanny Bitoun



Yanis Seddik

Elie Bruno

Othmane Idrissi Oudghiri

Emily Heer

Syrine Noamen

Ludovic Pujol