

# Mise en œuvre d'une classe inversée

**Pierre Poulain**

pierre.poulain@u-paris.fr

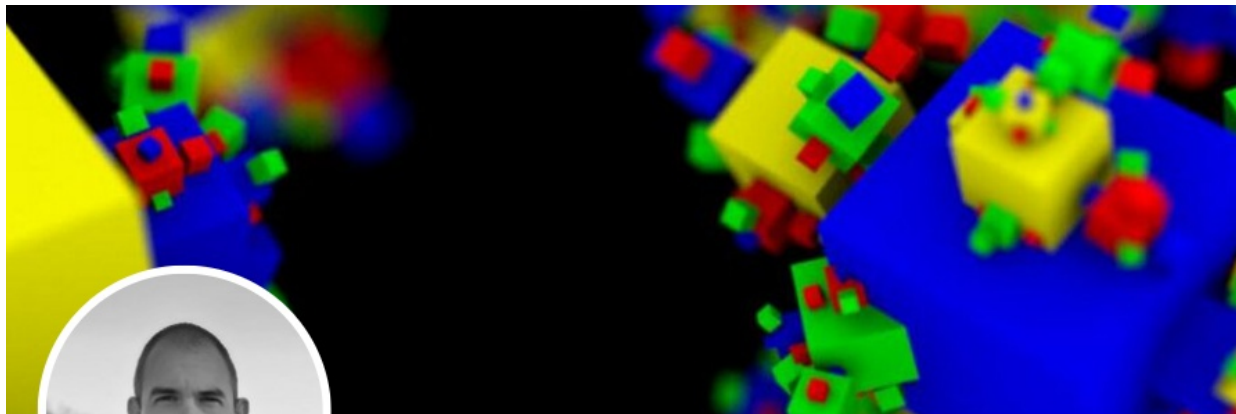
***Journée des initiatives pédagogiques***

**2022-12-08**



D'inversée à renversée,  
la classe dans tous ses états  
comme autant de preuves de concepts  
des **pédagogies actives**

# Bonjour !



**Pierre Poulain**

@pierrepo

Associate prof at [@univ\\_paris\\_cite](#) and [@IJMonod](#)  
[#bioinformatics](#) [#python](#) [#unix](#)  
[#openscience](#) [#opensource](#) [@SWHeritage](#) ambassador  
[#pedagogy](#) 🐘 [@pierrepo@mamot.fr](#)

📍 Paris, France 🔄 [cupnet.net](#) 📅 A rejoint Twitter en mars 2010

✉ [pierre.poulain@u-paris.fr](mailto:pierre.poulain@u-paris.fr)

🐦 [@pierrepo](#)

✉ [@pierrepo@mamot.fr](mailto:@pierrepo@mamot.fr)



SCIENCES DU VIVANT

**Sciences**  
Université Paris Cité

# Au menu

**Pourquoi ?**

**Comment ?**

**Et alors ?**

# Contexte

Enseignement « Programmation Python 2 », 3 ECTS, 30 h.

30-35 étudiants M1 & M2 (bio- & chimie-informatique)  
avec une expérience en programmation

De septembre à novembre :

- **7 CM x 2 h**
- 8 TP x 2 h

# Pourquoi ?

# Points de départ

+15 ans programmation Python

Formé à la pédagogie par SAPIENS  
(CertifiENS 2018)

Besoin de recentrer les cours magistraux  
sur les étudiants.

« Collecter des preuves d'apprentissage »  
Denis Berthiaume



# Matériel de cours

Cours « mature » en ligne (P. Fuchs) :

<https://python.sdv.univ-paris-diderot.fr/>

disponible sous licence CC BY-SA,

50-80 k visites / mois.





# Comment ?

# Les règles du jeu

## Syllabus

Ce document est un mode d'emploi, qui détaille le but et le fonctionnement de l'enseignement « Programmation Python 2 » et précise certains aspects logistiques. C'est aussi un contrat qui indique ce que nous attendons de vous et ce que vous pouvez attendre de l'équipe pédagogique.

Le langage de programmation Python est très utilisé, dans le milieu universitaire, comme dans le secteur privé. À plus court terme, vous serez amenés à utiliser ce langage dans d'autres enseignements, voire plus tard en stage.

### Objectifs d'apprentissage

Ces cours a pour objectifs de vous donner de solides compétences au langage de programmation Python. À l'issue de cet enseignement, vous devriez être capables de :

- Décrire l'écosystème Python.
- Manipuler des variables de différents types.
- Extraire et écrire des données depuis et vers des fichiers contenant de l'information biologique.
- Développer un algorithme utilisant des structures itératives, conditionnelles et des fonctions.
- Présenter des résultats d'analyse dans un notebook Jupyter.
- Développer un programme Python qui respecte les bonnes pratiques de programmation.

### Méthodes et moyens pédagogiques mis en œuvre

L'enseignement est composé de sessions de cours, de travaux pratiques et d'évaluations.

Une approche en classe inversée sera mise en place pour les sessions de cours. Cette pédagogie implique un engagement fort de votre part. Avant le cours en préparant le contenu et en participant aux activités préparatoires. Pendant le cours en participant aux activités proposées. Après le cours, en travaillant les exercices demandés et les projets.

Les travaux pratiques auront lieu dans les salles informatiques du bâtiment Lamarck.

Les sessions d'évaluation seront l'occasion d'évaluer vos compétences théoriques comme pratiques. Une partie d'entre elles se feront sur machine.

Compte-tenu de la situation sanitaire, cette organisation pourra être modifiée très rapidement, notamment vers des enseignements en distanciel.

### Ressources

Le site Moodle du cours contient toutes les informations dont vous aurez besoin pour cet enseignement : <https://moodle.u-paris.fr/course/view.php?id=4101> ou en plus court : <https://huit.re/PYTHON2>

Vous devez impérativement :

1. vous y inscrire (la clé d'inscription est PYTHON2022),
2. le consulter régulièrement,
3. participer aux activités proposées (forum, quiz, devoirs à rendre, QCM...)

Le poly de cours est disponible à la scolarité dans le bureau de Cédric de Cassan (bâtiment Lamarck B, bureau RH 44). Vous retrouverez l'intégralité du contenu du cours et même plus sur la page : <https://python.sdv.univ-paris-diderot.fr/>

### Déroulement de l'enseignement et planning

Le planning est disponible sur le site du cours sur Moodle. Il est susceptible de modifications, donc pensez à le consulter régulièrement.

L'enseignement compte 15 séances de 2 heures, avec une alternance de cours et de TP, réparties sur tout le semestre. À cela s'ajoute une dernière séance d'évaluation mi-décembre.

Cet enseignement compte pour 3 crédits ECTS, ce qui représente 30 heures de travail en présentiel et **au minimum 45 heures de travail personnel** (soit entre 3 et 4 h de travail personnel par semaine).

### Modalité d'évaluations

Tout au long du semestre, des évaluations formatives vous permettront de vous situer dans l'acquisition des connaissances. Ces évaluations ne sont pas prises en compte dans la note finale.

Plusieurs évaluations sommatives contribueront à la note finale de l'enseignement. Leur répartition est la suivante :

- 15 % Préparer les séances de cours en réalisant les activités préparatoires (répondre à un QCM, créer des questions de quiz...).
- 25 % Réaliser un projet (à partir de début octobre) en groupe et participer à son évaluation (fin novembre).
- 20 % Répondre à des questions à choix multiples (QCM).
- 40 % Répondre à une mise en situation lors de l'examen sur machine (en décembre).

Cette répartition pourra être modifiée en cas d'événement exceptionnel.

L'examen de 2<sup>e</sup> session qui a lieu en mai-juin est composé d'un unique examen sur machine qui compte pour 100 % de la note.

À la fin du semestre, nous vous demanderons de répondre anonymement à un questionnaire d'évaluation de l'enseignement. Vos réponses sont essentielles pour que nous puissions l'améliorer.

### Code de conduite

La préparation des séances de cours est obligatoire. Toute activité non faite sera pénalisée de 3 points sur la note de préparation (15 % de la note finale).

La présence en cours, en TP et aux sessions d'examen est obligatoire. En cas d'absence justifiée, le justificatif doit m'être adressé ainsi qu'au secrétariat pédagogique au plus tard 72 h après l'absence. Le cas contraire, l'absence sera considérée comme injustifiée. Toute absence injustifiée sera pénalisée de 5 points sur la note de préparation.

Soyez ponctuel et respectez l'horaire de début de l'enseignement. 8h30 le lundi, c'est tôt pour tout le monde, donc nous devons tous faire un effort. Tout retard sera pénalisé (par une exclusion de la session de cours ou de TP, voire un 0 pour la note globale de préparation de cours).

### Communication

Posez vos questions relatives à l'organisation ou le contenu du cours sur le forum (sur Moodle) prévu à cet effet. Vous devez y participer, c'est-à-dire poser des questions, mais aussi y répondre.

Je vous enverrai des messages via Moodle. Ces messages sont relayés sur votre adresse de messagerie de l'Université. Pensez à la consulter régulièrement.

Pour des questions personnelles uniquement, vous pouvez me contacter par e-mail à l'adresse [pierre.poulain@u-paris.fr](mailto:pierre.poulain@u-paris.fr). Je vous invite néanmoins à consulter ces conseils avant de m'écrire.

### Fraude et triche à l'université

Toute tentative de fraude ou de triche est sévèrement punie à l'Université. La peine encourue peut aller jusqu'à 5 ans d'exclusion de tout établissement universitaire. Pour plus d'information, consultez <https://u-paris.fr/sdv/sanctions-disciplinaires/>

Tous les scripts que vous soumettez seront systématiquement analysés par un outil de détection de plagiat.

### Relais handicap

Si vous êtes en situation de handicap, que ce soit de façon permanente ou temporaire, le Relais handicap peut vous accompagner pour la réussite de vos études, notamment lors des examens. Le Relais handicap se situe au rez-de-chaussée du bâtiment La Halle aux Farines, sur le campus des Grands Moulins. Vous pouvez également les contacter par téléphone ou par e-mail. Toutes les informations sont sur cette page : <https://u-paris.fr/etudes-et-handicap/>

Notez que si vous souhaitez un accompagnement particulier pour vos examens (par exemple du temps supplémentaire), vous devez vous inscrire au Relais handicap le plus rapidement possible et ne pas attendre les premiers examens. Vous devez également me prévenir au moins 2 semaines avant le premier examen et le Relais handicap doit me notifier que vous bénéficiez d'un aménagement.

### Étudiants non-francophones

Pour les évaluations, les étudiants non-francophones peuvent apporter un dictionnaire bilingue papier. Tout objet électronique est proscrit.



Lorena Barba (CC BY)

# Objectifs d'apprentissage

À l'issue de cet enseignement, vous devriez être capables de :

- Décrire l'écosystème Python.
- Manipuler des variables de différents types.
- Extraire et écrire des données depuis et vers des fichiers contenant de l'information biologique.
- Développer un algorithme utilisant des structures itératives, conditionnelles et des fonctions.
- Présenter des résultats d'analyse dans un *notebook* Jupyter.
- Développer un programme Python qui respecte les bonnes pratiques de programmation.

# Activités en amont, avant le CM

Travailler 3-4 chapitres de cours (poly ou en ligne).

Rédiger une « fiche mémo » = un résumé par chapitre (A4, recto, manuscrit) à déposer sur la plateforme en ligne (Moodle) de l'Université avant le cours (lundi 8h30).

# Fiches mémo

# Chapitre 7: les fichiers

Leo B

**brice.txt** = *robot*  
*ca fait*  
*gros cours!*

*n* : le cours *seule*

*open* ouvrir le fichier

**filin** = objet type *fichiers ouvert*

**close** = fermer le fichier

» **filin** = *open* ("brice.txt", "r")

» **filin** = *readlines()*

① *robot*, "ca fait", "n", "gros cours!" *→ liste des lignes du fichier*

» **filin.close()**

⚠ Impossible de lire un fichier fermé

**READLINE()**

**WITH** *open* ("brice.txt", "r") as **filin**:

**lignes** = *filin.readlines()*

for ligne in lignes } *print ligne*  
                                  *print ligne*

**FERMETURE AUTO**

*robot*  
*ca fait*  
*gros cours!*

**WITH**

**WITH** *open* ("brice.txt", "r") as **filin**:

*filin.read()*

*robot*, "ca fait", "n", "gros cours!" *→ chaîne de car. unique!*

**READ()**

**WITH** *open* ("brice.txt", "r") as **filin**:

*ligne* = *filin.readline()*

while *ligne != ""*:

*print* (*ligne*)

*ligne* = *filin.readline()*

*robot*  
*ca fait*  
*gros cours!*

**Ligne par ligne**

**READLINE()**

*amman* = ["pisson", "abell", "cho"]

**WITH** *open* ("gros.txt", "w") as **filout**: *⇒ on crée gros.txt*

for animal in *amman*:

*filout.write* ("{} "n".format(animal))

**8** } *nb octets dans gros.txt*      *gros.txt* = *pisson*  
**5**      *abell*  
                                  *cho*

**WRITE**      ⚠ On ne lit/écrit que des str!

**OUVRIR DEUX FICHIERS**

**WITH** *open* ("gros.txt", "r") as **fichier1**, *open* ("gros2.txt", "r") as **fichier2**:

for ligne in *fichier1*:

*fichier2.write* (" " + ligne)

*Donc gros2.txt* = *pisson*  
                          *abell*  
                          *cho*

**CH 15 : Bonnes pratiques en Python**

**Leo B**

- ④ Indentation
  - 4 espaces
- ② Modules
  - plutôt importer module que from module import \*
- ③ Nommage
  - \* variables : min - variable
  - \* constantes : MA - CONSTANCE
  - \* classes : Classer
  - \* Module : Ma Classe
- ④ Espaces
  - \* entouner ( $\pi$ , -, /, \*, \*\*, %, >, <=, >=, not, in, and, ...)
  - \* d'après des [], {}, {} et ()
  - \* après : et, mais pas avant
- ⑥ Commentaires
  - = x + 1 # In english please
- ⑦ Docstrings
  - """ Docstring simple """
  - ou
  - ''' Docstring long
  - Se termine les détails.
  - ~~~~~
  - Se termine !
  - "" "
- ⑧ Certeils certifié qualité
  - \* pydocstyle : conformité avec la PEP 8 ?
    - ↳ 4 pydocstyle script - doctrine .py
    - ⇒ donne un rapport
  - + pydocstyle : conforme avec la PEP 257 et docstrings
    - ↳ donne un rapport
  - \* pylint : tente de comprendre le code et propose des améliorations
- ⑨ Organiser son code
  - ① Docstring décrivant global<sup>ité</sup> le script
  - ② Métadonnées (copyright)
  - ③ Modules (importer, puis utiliser)
  - ④ Constantes
  - ⑤ Classe
- ⑥ Docstring décrivant la classe
  - ⑦ Lignes vides avant chaque méthode de classe
  - ⑧ Fonctions classiques, 2 lignes vides avant
  - ⑨ Programme principal

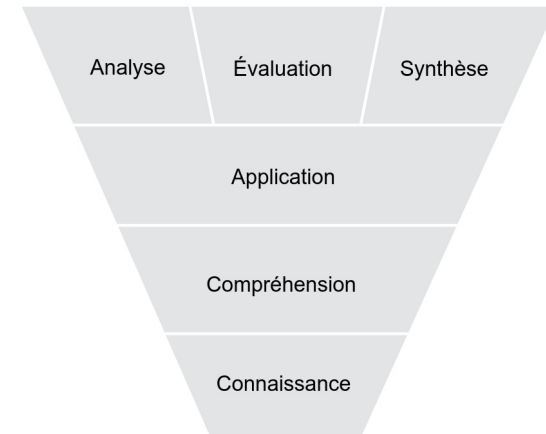
# Activités en CM

## Principes :

- Développer des compétences en programmation Python mais **sans ordinateur**.
- Remédier, discuter et renforcer les notions abordées en autonomie.

# Activités en CM

- Fiches mémo : comparer les fiches, discuter entre pairs, rentrer dans le CM.
- **Quiz en ligne** (Moodle) : lister, résumer, appliquer.
- **Penser, comparer, partager** : élaborer, comparer, argumenter.
- Script à trous : contextualiser, résoudre.
- Script tournant : créer, contextualiser.



# Quiz en ligne (WooClap)

QROC

1

Listez les méthodes disponibles pour les chaînes de caractères.  
Sans point ni parenthèse.

REPLACE DEL SLIP SPLIT(MAXSPLIT COUNT(X) MUTATION POWER  
CAPITALIZE LOWER UPPER SPLIT APPEND DEEPCOPY  
SORT X) STRIP STARTSWITH 6 /N REPLACE = FIND COUNT  
REPLACE(X) DECOPY STARTSWITH() LIST COUNT() REMOVE REVER  
SPLIT(X) APITALISE LOWER() STATSV



# Quiz en ligne (WooClap)

## QCM

2

Que renvoie le script toto.py ?

```
import sys
print(sys.argv[1:])
```

lancé avec la commande :

```
python toto.py 1 8 souris 3.14
```

<code>['toto.py', 1, 8, 'souris', 3.14]</code>	<div><div></div></div>	0%	0 votes
✓ <code>['1', '8', 'souris', '3.14']</code>	<div><div></div></div>	35%	7 votes
<code>['toto.py', '1', '8', 'souris', '3.14']</code>	<div><div></div></div>	55%	11 votes
<code>[1, 8, 'souris', 3.14]</code>	<div><div></div></div>	10%	2 votes

# Penser, comparer, partager

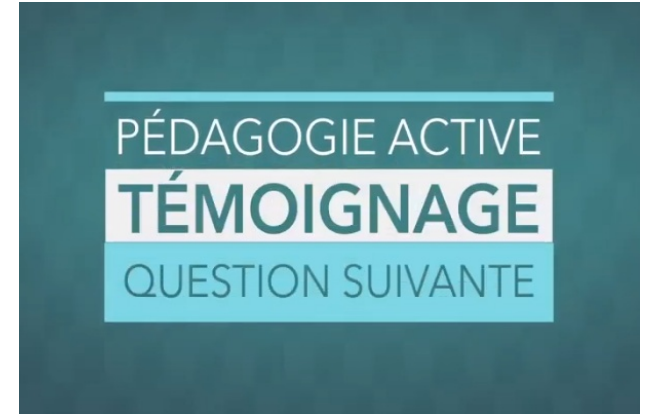
## *Think-pair-share*

*« Je suis un nombre compris entre 100 et 500. La somme de mes chiffres vaut 15. Le produit du chiffre des centaines et de celui des unités vaut 10. Qui suis-je ? »*

- Mise en situation, un petit problème à résoudre en Python ;
- Travail seul (sur feuille),
- Échange avec 1 ou 2 voisins,
- Échange et discussion tous ensemble.

# Après le CM : concevoir des QCM

Activité « StudentQuiz » sur Moodle,  
~ version numérique de « [Question suivante](#) »



Chaque étudiant va :

- Concevoir 3 questions ;
- Tester et commenter les questions des autres.

Pour l'examen, sélection de 5 questions (/15)

# Et alors ?

# Évaluation de l'enseignement (EEE)

Parmi les activités suivantes, évaluez leur utilité pour votre apprentissage de Python.

1 : pas du tout, 5 : beaucoup

	Rang moyen ↓				
	1	2	3	4	5
Les fiches mémo.				■	
Les quiz et nuages de mots sur WooClap.				■	
Le script à trous (l'imprimante a "mangé" certaines parties du script).				■	
Le script tournant (vous commencez à écrire un script puis vous l'échangez avec votre voisin).				■	
La conception de QCM sur Moodle.					■
La pair programming (en TP).				■	

# Évaluation de l'enseignement (EEE)

Temps de travail personnel moyen par semaine : 2h20  
(50 min → 3h)

Entre 3 à 4 h annoncées sur le syllabus.

- 1 ECTS → 25 à 30 h de travail
- 3 ECTS → 75 h de travail
- 30 h de présentiel, reste 45 h de travail personnel
- semestre de 13 semaines → 3,5 h travail personnel / semaine

# Évaluation de l'enseignement (EEE)

Très diligent et dédié aux élèves. Keep it up !

J'ai aimé ce cours. je trouve que malheureusement, on a pas de cours sur Pandas et d'**analyse de donnée** de Python. je pense que pour l'an prochain il faudrait faire un projet sur **MDAnalysis** car c'est un outil qui est très utile pour le Master ISDD.

Le cour est très bien pour le temps que l'on a. Je ne vois pas trop ce qui pourrait être ajouté avec seulement deux heures de cours par semaine. Peut-être **réduire le nombre de projets a corrigé**, 4 c'est sacrément chronophage.

L'enseignement est très bien et nous pousse a être **assidu** tout au long de l'UE. Les cours magistraux étaient bien structurés (**wooclap**, exercice sur feuille, ...) Les Travaux pratiques sont aussi correctes. De plus les compétences apprises et le cours en ligne m'ont été **utiles pour mon stage**.

# Conclusion



# La classe inversée (renversée) c'est :

- L'occasion de mettre en place des **pédagogies actives**.
- Chronophage pour l'enseignant, pas nécessairement pour les étudiants (si structuré).
- Responsabiliser et motiver les étudiants.  
*From Sage on the Stage to Guide on the Side* (King, 1993)
- Scalable (Math, L1 SDV, 250 étudiants)

# Merci !

# <https://colibris.link/jip22ci>

**SAPIENS**

Antoinette Bouziane

Marine Lanteri

**Pôle innovation  
pédagogique UPCité**

Barbara Pacé

Cloé Delévaque

## A structured guide for implementing a flipped classroom

Pierre Poulain<sup>1\*</sup>, Mickael Bertrand<sup>2</sup>, Héloïse Dufour<sup>3</sup>, Antoine Taly<sup>4,5\*</sup>,

<sup>1</sup> Université de Paris, CNRS, Institut Jacques Monod, F-75013, Paris, France

<sup>2</sup> Éducation Nationale, Académie de Dijon, Lycée Anna Judic, F-21140, Semur-en-Auxois, France

<sup>3</sup> Cercle FSER, F-75007, Paris, France

<sup>4</sup> CNRS, Université de Paris, UPR 9080, Laboratoire de Biochimie Théorique, F-75005, Paris, France

<sup>5</sup> Institut de Biologie Physico-Chimique, Fondation Edmond de Rothschild, PSL Research University, F-75005, Paris, France

\* taly@ibpc.fr, pierre.poulain@u-paris.fr

## Abstract

The way flipped classrooms are perceived and even practised by teachers is sometimes approximative. For example, while the Covid-19 pandemic has pushed many universities to adopt distance learning, flipped classrooms have often been mentioned as a solution in that context. This inducement maintains a confusion between flipped classrooms and distance learning that might be detrimental for both students and teachers. Moreover, embarking on a new pedagogical practice such as flipped classroom could be intimidating and time-consuming.

For these reasons, this article aims to share some tips for implementing a flipped classroom, with examples from biology and biochemistry. Based on our own experiences but also on the current scientific literature, we propose to structure the advises around three phases: Preparation, Implementation, and Follow-up.

In the preparation phase it is advised to not only plan to invert time but also say it, as well as to identify (or create) resources for learning in autonomy. In the Implementation phase it is suggested to i) Be explicit in the acquisition of knowledge, and foster students' autonomy; ii) Explore active learning in class; iii) Develop skills of cooperation and sharing; and iv) Differentiate. In the follow-up phase it is proposed to i) Evaluate; ii) Take care of the logistics and your posture; iii) Document your flipped classroom and iv) Share.

DOI [10.20944/preprints202007.0030.v3](https://doi.org/10.20944/preprints202007.0030.v3)

**LA BOÎTE À OUTILS  
DU PROFESSEUR**

**ENSEIGNER  
ET FORMER**  
**PSYCHOLOGIE APPLIQUÉE  
ET PÉDAGOGIES ACTIVES**  
Secondaire, supérieur, formation professionnelle

- Bien construire son cours
- Accueillir ses élèves et gérer sa classe
- Choisir les méthodes pédagogiques efficaces
- Hybrider intelligemment son enseignement



Jean-François Parmentier  
et Quentin Vicens

**DUNOD**

HAL Id [hal-03737367](https://hal.archives-ouvertes.fr/hal-03737367)

# Bonus tracks

# Implémentation

Classe inversée de « type I » de Lebrun :  
l'enseignant fournit le contenu (vidéo, poly...)

avec un peu de « type II » :  
les étudiants créent le contenu (projet tutoriel)

M Lebrun, C Goffinet & C Gilson, Vers une typologie des classes inversées.  
Education & Formation, Vol. e-306, no.2, p. 125-146, 2016.

# Évaluation de l'enseignement (EEE)

Évaluez les propositions suivantes :

1: vous n'êtes pas d'accord, 5 : vous êtes tout à fait d'accord

	Rang moyen					↓
	1	2	3	4	5	
La classe inversée m'a aidé dans l'appropriation des concepts de programmation en Python.				■		4.4
Vous êtes capable de "Décrire l'écosystème Python".				■		4.1
Vous êtes capable de "Manipuler des variables de différents types".				■		4.4
Vous êtes capable d' "Extraire et écrire des données depuis et vers des fichiers contenant de l'information biologique".				■		4.3
Vous êtes capable de "Développer un algorithme simple utilisant des structures itératives et conditionnelles".				■		4.5
Vous êtes capable de "Présenter des résultats d'analyse dans un notebook Jupyter".				■		4.5
Vous êtes capable de "Développer un programme Python qui respecte les bonnes pratiques de programmation".				■		4.3

# Que dit la science ?

PNAS PNAS PNAS PNAS



## Active learning increases student performance in science, engineering, and mathematics

Scott Freeman<sup>a,1</sup>, Sarah L. Eddy<sup>a</sup>, Miles McDonough<sup>a</sup>, Michelle K. Smith<sup>b</sup>, Nnadozie Okoroafor<sup>a</sup>, Hannah Jordt<sup>a</sup>, and Mary Pat Wenderoth<sup>a</sup>

<sup>a</sup>Department of Biology, University of Washington, Seattle, WA 98195; and <sup>b</sup>School of Biology and Ecology, University of Maine, Orono, ME 04469

Edited\* by Bruce Alberts, University of California, San Francisco, CA, and approved April 15, 2014 (received for review October 8, 2013)

To test the hypothesis that lecturing maximizes learning and course performance, we metaanalyzed 225 studies that reported data on examination scores or failure rates when comparing student performance in undergraduate science, technology, engineering, and mathematics (STEM) courses under traditional lecturing versus active learning. The effect sizes indicate that on average, student performance on examinations and concept inventories increased by 0.47 SDs under active learning ( $n = 158$  studies), and that the odds ratio for failing was 1.95 under traditional lecturing ( $n = 67$  studies). These results indicate that average examination scores improved by about 6% in active learning sections, and that students in classes with traditional lecturing were 1.5 times more likely to fail than were students in classes with active learning. Heterogeneity analyses indicated that both results hold across the STEM disciplines, that active learning increases scores on concept inventories more than on course examinations, and that active learning appears effective across all class sizes—although the greatest effects are in small ( $n \leq 50$ ) classes. Trim and fill analyses and fail-safe  $n$  calculations suggest that the results are not due to publication bias. The results also appear robust to variation in the methodological rigor of the included studies, based on the quality of controls over student quality and instructor identity. This is the largest and most comprehensive metaanalysis of undergraduate STEM education published to date. The results raise questions about the continued use of traditional lecturing as a control in research studies, and support active learning as the preferred, empirically validated teaching practice in regular classrooms.

225 studies in the published and unpublished literature. The active learning interventions varied widely in intensity and implementation, and included approaches as diverse as occasional group problem-solving, worksheets or tutorials completed during class, use of personal response systems with or without peer instruction, and studio or workshop course designs. We followed guidelines for best practice in quantitative reviews (*SI Materials and Methods*), and evaluated student performance using two outcome variables: (i) scores on identical or formally equivalent examinations, concept inventories, or other assessments; or (ii) failure rates, usually measured as the percentage of students receiving a D or F grade or withdrawing from the course in question (DFW rate).

The analysis, then, focused on two related questions. Does active learning boost examination scores? Does it lower failure rates?

### Results

The overall mean effect size for performance on identical or equivalent examinations, concept inventories, and other assessments was a weighted standardized mean difference of 0.47 ( $Z = 9.781$ ,  $P < 0.001$ )—meaning that on average, student performance increased by just under half a SD with active learning compared with lecturing. The overall mean effect size for failure rate was an odds ratio of 1.95 ( $Z = 10.4$ ,  $P < 0.001$ ). This odds ratio is equivalent to a risk ratio of 1.5, meaning that on average, students in traditional lecture courses are 1.5 times more likely to fail than students in courses with active learning. Average failure rates were 21.8% under active learning but 33.8% under traditional lecturing—a difference that represents a 55% increase

Freeman, PNAS, 2014. DOI 10.1073/pnas.1319030111