Réunion de rentrée option D

28 septembre 2020

La préparation à Paris-Saclay

Préparation des deux épreuves spécifiques de l'option D:

- leçon d'informatique
- modélisation

Contenu:

- Cours, TD de révision, approfondissement du programme
- TP de programmation
- entraînement à l'épreuve de modélisation
- entraînement à l'épreuve de leçon
- développements blancs (décembre, janvier)
- oraux blancs (entre l'écrit et l'oral)

La préparation à Paris-Saclay

Préparation des deux épreuves spécifiques de l'option D:

- leçon d'informatique
- modélisation

Bon, on espère qu'elles auront lieu cette année!

Contenu:

- Cours, TD de révision, approfondissement du programme
- TP de programmation
- entraînement à l'épreuve de modélisation
- entraînement à l'épreuve de leçon
- développements blancs (décembre, janvier)
- oraux blancs (entre l'écrit et l'oral)

Implication attendue

- Au moins 2 développements blancs
- Au moins 2 leçons
- Au moins 1 DM de programmation
- Au moins 2 entraînements de modélisation
- Au moins 1 leçon + 2 oraux blancs de modélisation
- **Intérêt collectif**: assister aux présentations des autres, comparer, s’améliorer
- Dérogations occasionnelles (report, remplacement) si demandé suffisamment tôt et justifié (santé, surcharge de travail en maths)

Implication attendue

- Au moins 2 développements blancs
- Au moins 2 leçons
- Au moins 1 DM de programmation
- Au moins 2 entraînements de modélisation
- Au moins 1 leçon + 2 oraux blancs de modélisation
- **Intérêt collectif**: assister aux présentations des autres, comparer, s'améliorer
- Dérogations occasionnelles (report, remplacement) si demandé suffisamment tôt et justifié (santé, surcharge de travail en maths)

D'un autre côté, il est demandé de « réduire la jauge de 30-40% » — je vous laisse apprécier la situation

Comment travailler

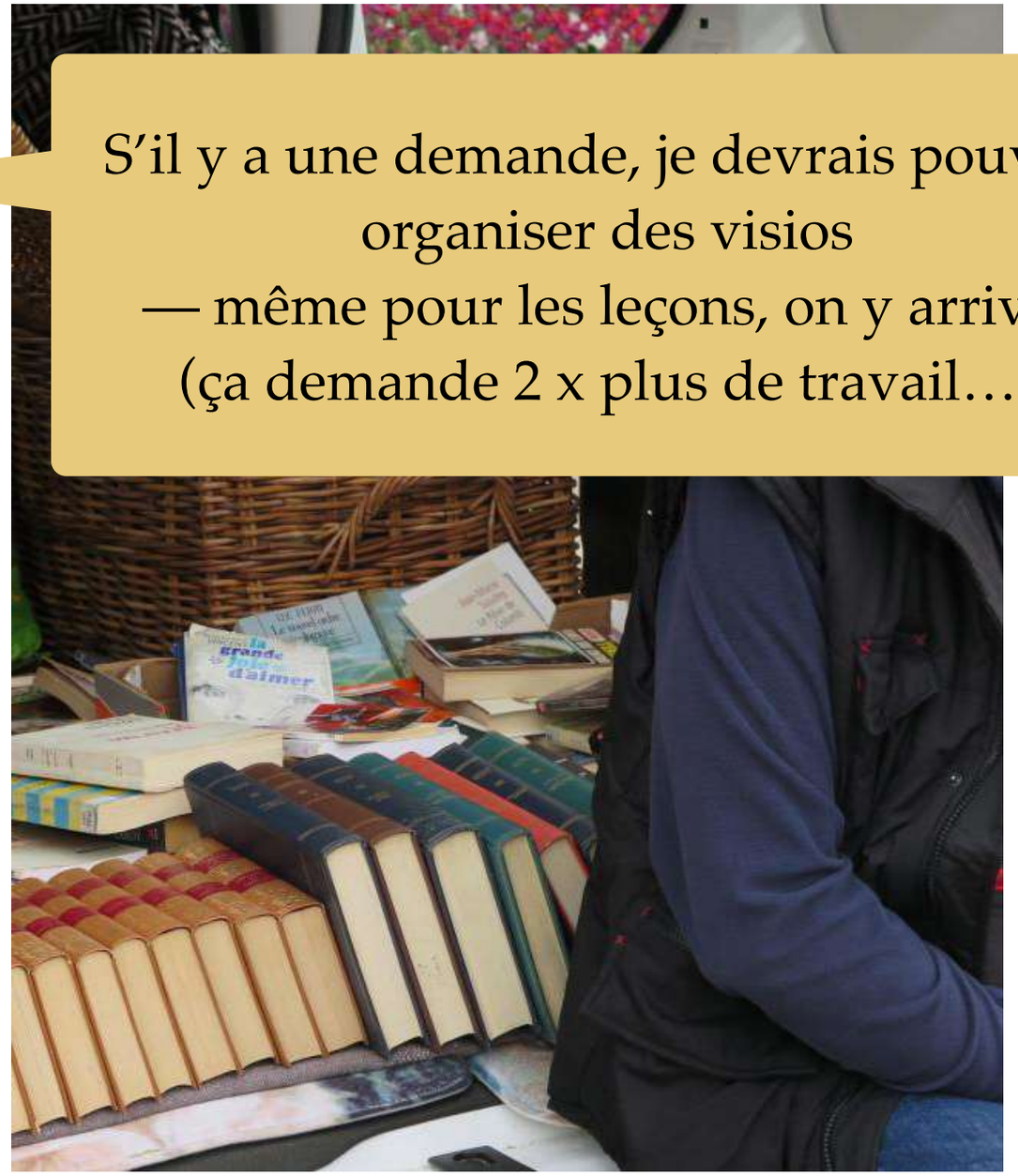
- **Venir**: observer, s’inspirer, s’entraîner
- **Lire**: repérer références, livres, résultats, preuves
- **Echanger**: avec les enseignants, entre vous

- Tout couvrir, au moins sur les <u>bases</u>: pas d’impasse!
- Approfondir ce qui vous plaît davantage
- <u>Connaître son niveau</u>: un exposé doit être maîtrisé

Comment travailler

- **Venir**: observer, s'inspirer, s'entraîner
- **Lire**: repérer références, livres, résultats, preuves
- **Echanger**: avec les enseignants, entre vous

- Tout couvrir, au moins sur les <u>bases</u>: pas d'impasse!
- Approfondir ce qui vous plaît davantage
- <u>Connaître son niveau</u>: un exposé doit être maîtrisé



S'il y a une demande, je devrais pouvoir organiser des visios
— même pour les leçons, on y arrive!
(ça demande 2 x plus de travail…)

Comment travailler

- **Venir**: observer, s'inspirer, s'entraîner
- **Lire**: repérer références, livres, résultats, preuves
- **Echanger**: avec les enseignants, entre vous

- Tout couvrir, au moins sur les <u>bases</u>: pas d'impasse!
- Approfondir ce qui vous plaît davantage
- <u>Connaître son niveau</u>: un exposé doit être maîtrisé

S'il y a une demande, je devrais pouvoir organiser des visios
— même pour les leçons, on y arrive!
(ça demande 2 x plus de travail…)

Ca c'est le point crucial
(et régulièrement souligné par le jury)

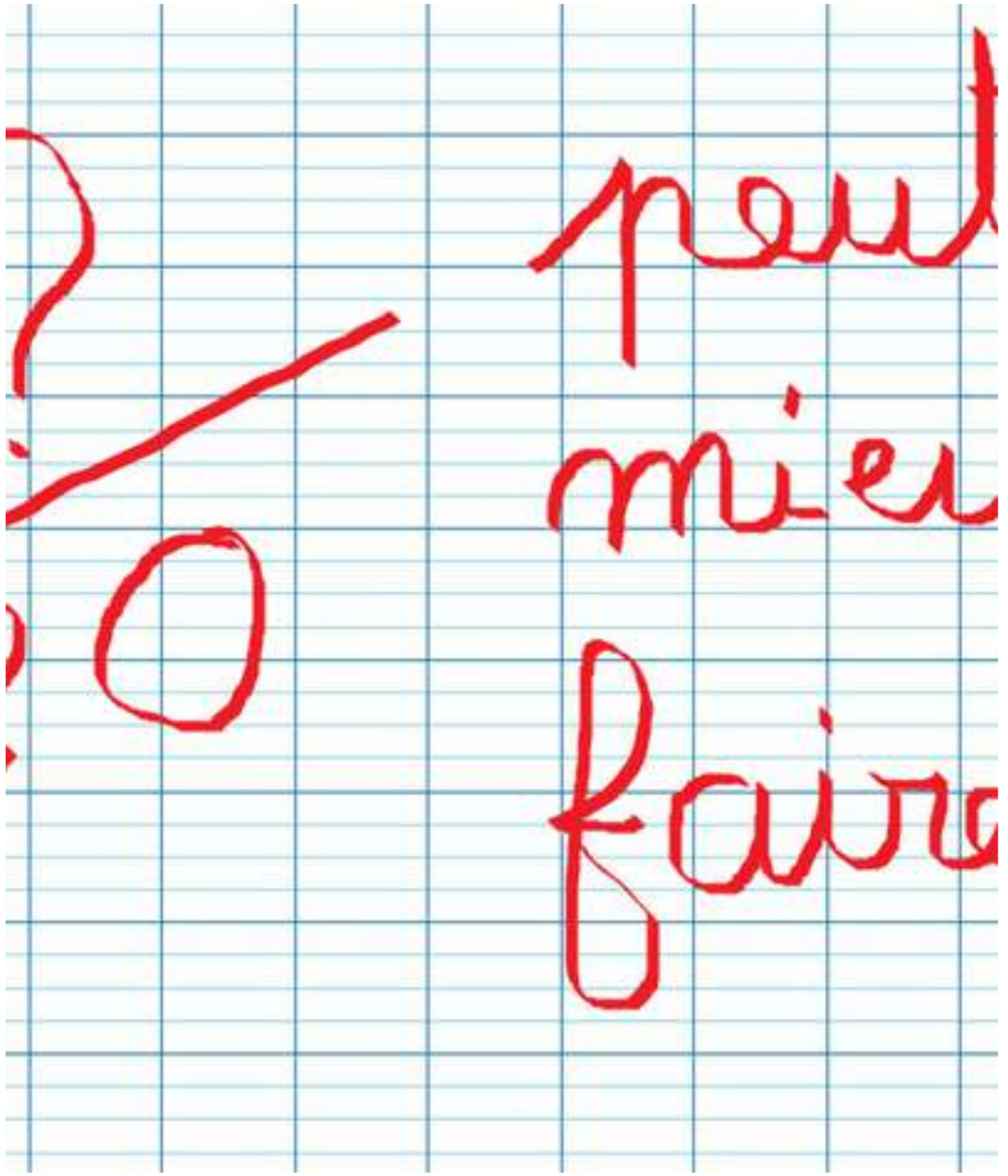
Notes, master FESUP

Premier semestre:

- Leçon d'informatique: notes des développements blancs
- Modélisation: notes du DM de programmation 1

Second semestre:

- Leçon d'informatique: moyenne des notes des leçons préparées
- Modélisation: oraux blancs + DM programmation optionnel



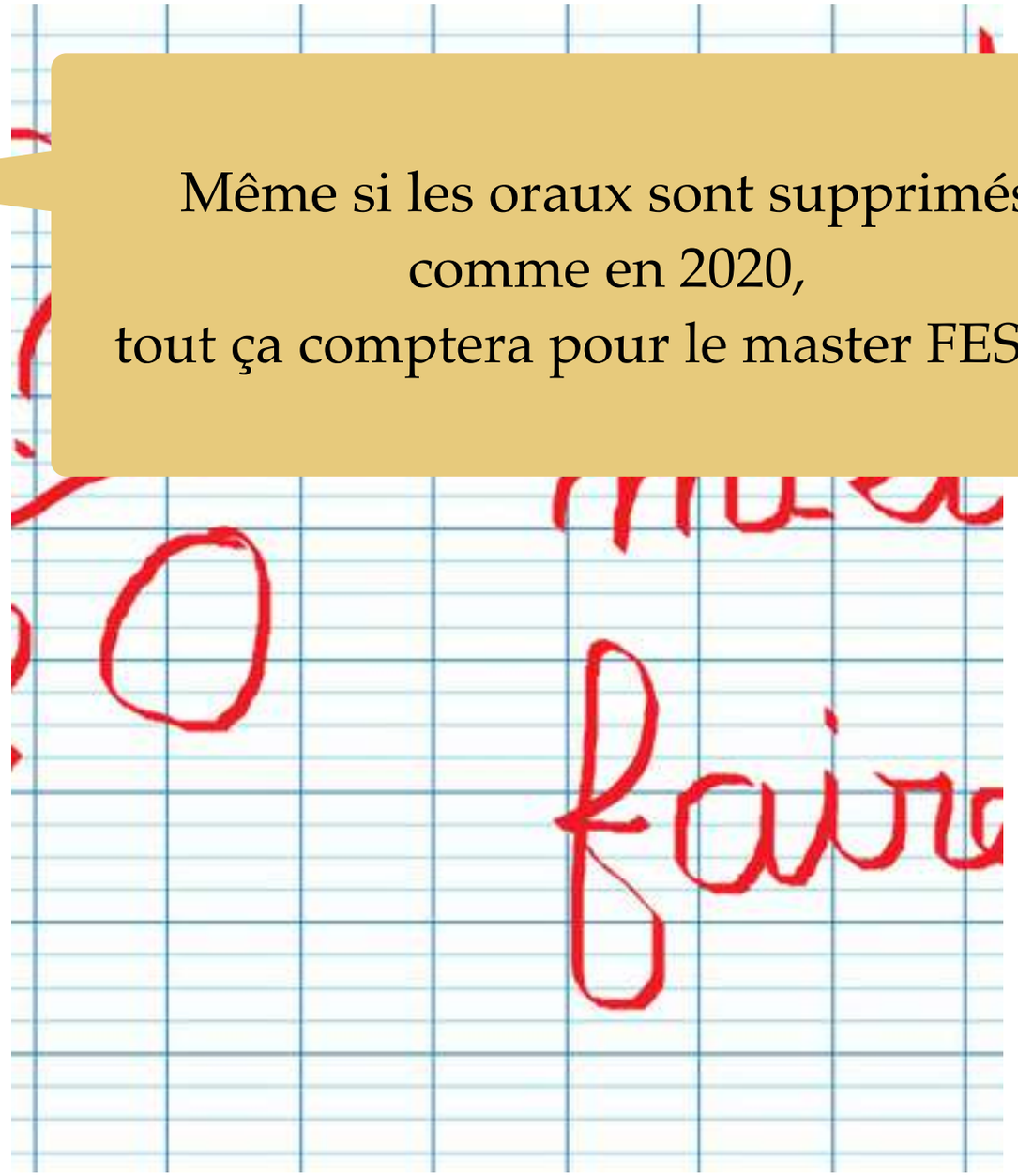
Notes, master FESUP

Premier semestre:

- Leçon d'informatique: notes des développements blancs
- Modélisation: notes du DM de programmation 1

Second semestre:

- Leçon d'informatique: moyenne des notes des leçons préparées
- Modélisation: oraux blancs + DM programmation optionnel



Même si les oraux sont supprimés
comme en 2020,
tout ça comptera pour le master FESUP

Délégué(e)

- Fait remonter les problèmes
- Participe aux réunions du département info (~2/an)
- Informe les autres agrégatifs

Délégué(e)

Qui?

- Fait remonter les problèmes
- Participe aux réunions du département info (~2/an)
- Informe les autres agrégatifs

Programme, épreuves

- Programme 2018-19: http://www.dptinfo.ens-cachan.fr/Agregation/Programme2020.pdf

- A lire absolument, le rapport de l'année précédente: http://www.dptinfo.ens-cachan.fr/Agregation/rapport2019.pdf

- Infos sur: http://www.dptinfo.ens-cachan.fr/Agregation/

Épreuves orales de l'option D : informatique

1re Épreuve : Mathématiques

Le programme de cette épreuve est constitué des titres 1 à 13 ci-dessus. Les candidats se verront proposer deux sujets, dans un corpus d'algèbre, de géométrie, d'analyse et de probabilités.

2e Épreuve : Informatique Fondamentale

Le programme spécifique de cette épreuve est constitué des titres 14 à 17 ci-après.

3e Épreuve : Modélisation

Le programme de cette épreuve est constitué des titres 14 à 17 ci-après. L'épreuve consiste en un exposé de *modélisation informatique* construit en partant d'un texte choisi par le candidat parmi deux qui lui sont proposés par le jury. Le programme fournit les éléments fondamentaux permettant la compréhension du texte, la formalisation des problèmes décrits et leur analyse.

Le texte comporte un exercice de programmation destiné à évaluer la capacité du candidat à construire un court programme informatique, en expliquer la structure et défendre les choix qui auront été faits. Le candidat devra également s'appuyer sur ce programme informatique, ou d'autres à son choix, pour illustrer certains aspects pertinents du texte.

Les langages informatiques C, Caml, Python et Java seront disponibles pour cette épreuve et sa préparation. L'exercice de programmation devra être traité en utilisant l'un de ces langages. Le site de l'agrégation externe de mathématiques (http://agreg.org) précise la liste des logiciels mis à la disposition des candidats et la nature de leur environnement.

Programme spécifique de l'option D.

L'objectif de cette option est de s'assurer que les candidats maîtrisent les fondements essentiels et structurants de la science informatique. Le programme n'est pas rédigé comme un plan de cours, il décrit les notions que les candidats doivent maîtriser. Aucun langage de programmation particulier n'est imposé. Les candidats doivent maîtriser au moins un langage et son environnement de programmation parmi C, Caml, Python ou Java.

14 Algorithmique fondamentale

Cette partie insiste sur les notions de preuve et de complexité des algorithmes. Elle est relativement indépendante de tout langage de programmation, mais le candidat doit être capable de mettre en oeuvre sur machine les structures de données et les algorithmes étudiés.

(a) Structures de données. Types abstraits : définition des tableaux, listes, piles, files, arbres, graphes (orientés et non orientés), ensembles, dictionnaires, file de priorité. Interface abstraite et implantation concrète.

(b) Schémas algorithmiques classiques : approche gloutonne, diviser pour régner, programmation dynamique. Exemples : algorithme de Dijkstra, tri fusion, plus longue sous-séquence commune.

(c) Complexité. Analyse des algorithmes : relations de comparaison O, Θ et Ω. Analyse dans le pire cas. Exemple d'analyse en moyenne : recherche d'un élément dans un tableau.

Programme spécifique de l'épreuve de modélisation, option D.

Sur les sujets suivants, il est attendu moins une capacité à les restituer de manière structurée qu'une connaissance suffisante pour comprendre la problématique d'un texte, en éclairer le contexte, ou encore argumenter (ou critiquer) les solutions que ce dernier propose. *En particulier, aucune expérience de programmation assembleur ou système n'est attendue.*

(a) Eléments d'architecture des ordinateurs : principaux composants et leurs interactions ; principes des langages assembleurs ;

(b) Représentation des nombres entiers et flottants ;

(c) Eléments sur les systèmes d'exploitation : systèmes de fichiers, processus, gestion de la mémoire.

(d) Algorithmes de tri et de recherche. Méthodes de tri par c
rapide), arbre de décision et borne inférieure du tri par c
séquentielle et dichotomique. Arbres binaires de recherche.
entre la taille et la hauteur, maintien de l'équilibre.

(e) Algorithmes de graphes. Parcours de graphes : algorithmes
algorithme de Dijkstra. Arbres couvrants : algorithmes
transitive.

15 Calculabilité, décidabilité et complexité

(a) Définition des fonctions primitives récursives ; schémas prim
des fonctions récursives ; fonction d'Ackermann.

(b) Définitions des machines de Turing. Équivalence avec les

(c) Universalité. Décidabilité, Indécidabilité. Théorème de l'ar
Turing. Définitions et caractérisations des ensembles réc

(d) Lambda-calcul pur comme modèle de calcul : définition,
pressivité.

(e) Complexité en temps et en espace : classe P. Machines de T
Acceptation par certificat. Réduction polynomiale. NP-co

16 Logique et démonstration

(a) Calcul propositionnel : syntaxe et sémantique. Tables de
forme clausale. Théorème de complétude du calcul propos

(b) Logique du premier ordre : aspects syntaxiques. Langages
variables liées, substitutions, capture de variables.

(c) Logique du premier ordre : systèmes formels de preuve. Ca
Algorithme d'unification des termes. Preuves par résolutio

(d) Logique du premier ordre : aspects sémantiques. Interpré
Validité, satisfiabilité. Théories cohérentes, théories complè
Exemples de théories : égalité, arithmétique de Peano. T
prédicats du premier ordre.

17 Théorie des langages de programmation

(a) Preuve de programmes : correction, terminaison. Méthodes
postconditions, invariants et variants de boucles, logique d

(b) Sémantique des langages de programmation : sémantiques
plication à un langage impératif restreint.

(c) Automates finis. Langages reconnaissables. Existence de la
nisation. Propriétés de clôture des langages reconnaissable

(d) Expressions rationnelles. Langages rationnels. Théorème d

(e) Grammaires et Langages algébriques. Existence de langages
des langages algébriques.

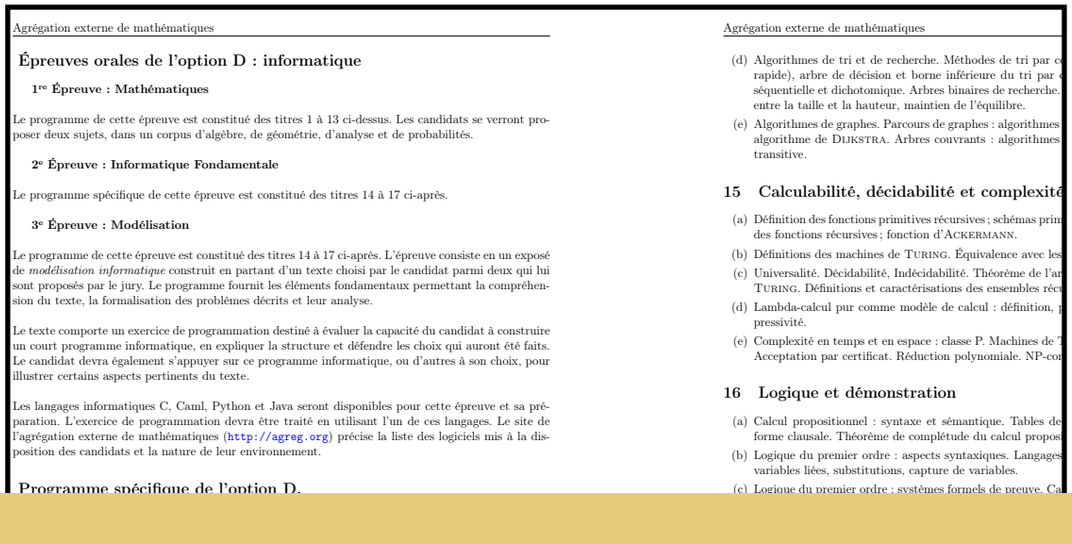
(f) Chaîne de compilation. Analyse lexicale. Analyse syntaxiq
et ascendante). Analyse sémantique élémentaire (arbre de
analyse de portée, typage, ...).

Programme, épreuves

- Programme 2018-19: http://www.dptinfo.ens-cachan.fr/Agregation/Programme2020.pdf

- A lire absolument, le rapport de l'année précédente: http://www.dptinfo.ens-cachan.fr/Agregation/rapport2019.pdf

- Infos sur: http://www.dptinfo.ens-cachan.fr/Agregation/



Agrégation externe de mathématiques

Épreuves orales de l'option D : informatique

1re Épreuve : Mathématiques

Le programme de cette épreuve est constitué des titres 1 à 13 ci-dessus. Les candidats se verront proposer deux sujets, dans un corpus d'algèbre, de géométrie, d'analyse et de probabilités.

2e Épreuve : Informatique Fondamentale

Le programme spécifique de cette épreuve est constitué des titres 14 à 17 ci-après.

3e Épreuve : Modélisation

Le programme de cette épreuve est constitué des titres 14 à 17 ci-après. L'épreuve consiste en un exposé de *modélisation informatique* construit en partant d'un texte choisi par le candidat parmi deux qui lui sont proposés par le jury. Le programme fournit les éléments fondamentaux permettant la compréhension du texte, la formalisation des problèmes décrits et leur analyse.

Le texte comporte un exercice de programmation destiné à évaluer la capacité du candidat à construire un court programme informatique, en expliquer la structure et défendre les choix qui auront été faits. Le candidat devra également s'appuyer sur ce programme informatique, ou d'autres à son choix, pour illustrer certains aspects pertinents du texte.

Les langages informatiques C, Caml, Python et Java seront disponibles pour cette épreuve et sa préparation. L'exercice de programmation devra être traité en utilisant l'un de ces langages. Le site de l'agrégation externe de mathématiques (http://agreg.org) précise la liste des logiciels mis à la disposition des candidats et la nature de leur environnement.

Programme spécifique de l'option D

Agrégation externe de mathématiques

(d) Algorithmes de tri et de recherche. Méthodes de tri par c
rapide), arbre de décision et borne inférieure du tri par c
séquentielle et dichotomique. Arbres binaires de recherche.
entre la taille et la hauteur, maintien de l'équilibre.

(e) Algorithmes de graphes. Parcours de graphes : algorithmes
algorithme de Dijkstra. Arbres couvrants : algorithmes
transitive.

15 Calculabilité, décidabilité et complexité

(a) Définition des fonctions primitives récursives ; schémas prim
des fonctions récursives ; fonction d'Ackermann.

(b) Définitions des machines de Turing. Équivalence avec les

(c) Universalité. Décidabilité, Indécidabilité. Théorème de l'ar
Turing. Définitions et caractérisations des ensembles réc

(d) Lambda-calcul pur comme modèle de calcul : définition,
pressivité.

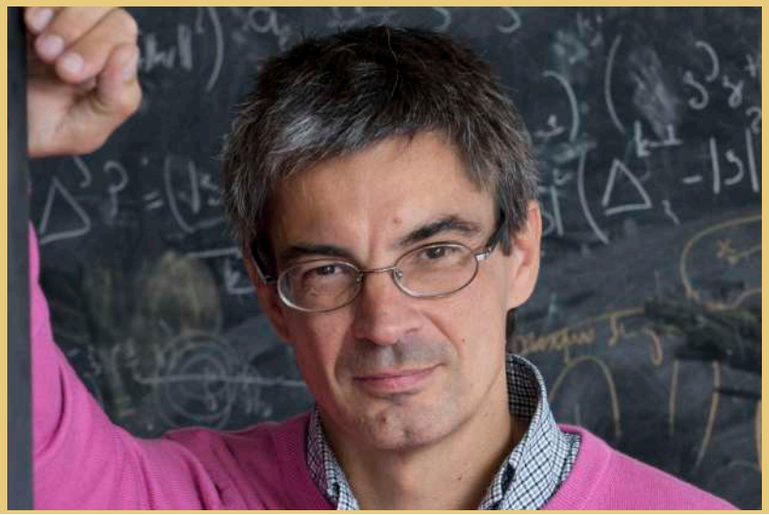
(e) Complexité en temps et en espace : classe P. Machines de T
Acceptation par certificat. Réduction polynomiale. NP-co

16 Logique et démonstration

(a) Calcul propositionnel : syntaxe et sémantique. Tables de
forme clausale. Théorème de complétude du calcul propos

(b) Logique du premier ordre : aspects syntaxiques. Langages
variables liées, substitutions, capture de variables.

(c) Logique du premier ordre : systèmes formels de preuve. Ca



Thierry Goudon, président du jury (jusqu'en 2020)

« Le rapport du jury est à apprendre par cœur

Note

- La présidence du jury change: à partir de 2021, c’est **Claudine Picaronny** (IG)

Leçon: 1. préparation

Préparation: 3h.

- Choix d'un sujet parmi 2 tirés:
  21 leçons, 4 thèmes
  tous couplages possibles

- Rédaction d'un **plan**
  3 pages manuscrites
  1 page de figures (opt.)
  marge 1cm

- identifier (≥) 2 **développements**

- livres **autorisés**
  (personnels, ou dans la malle)

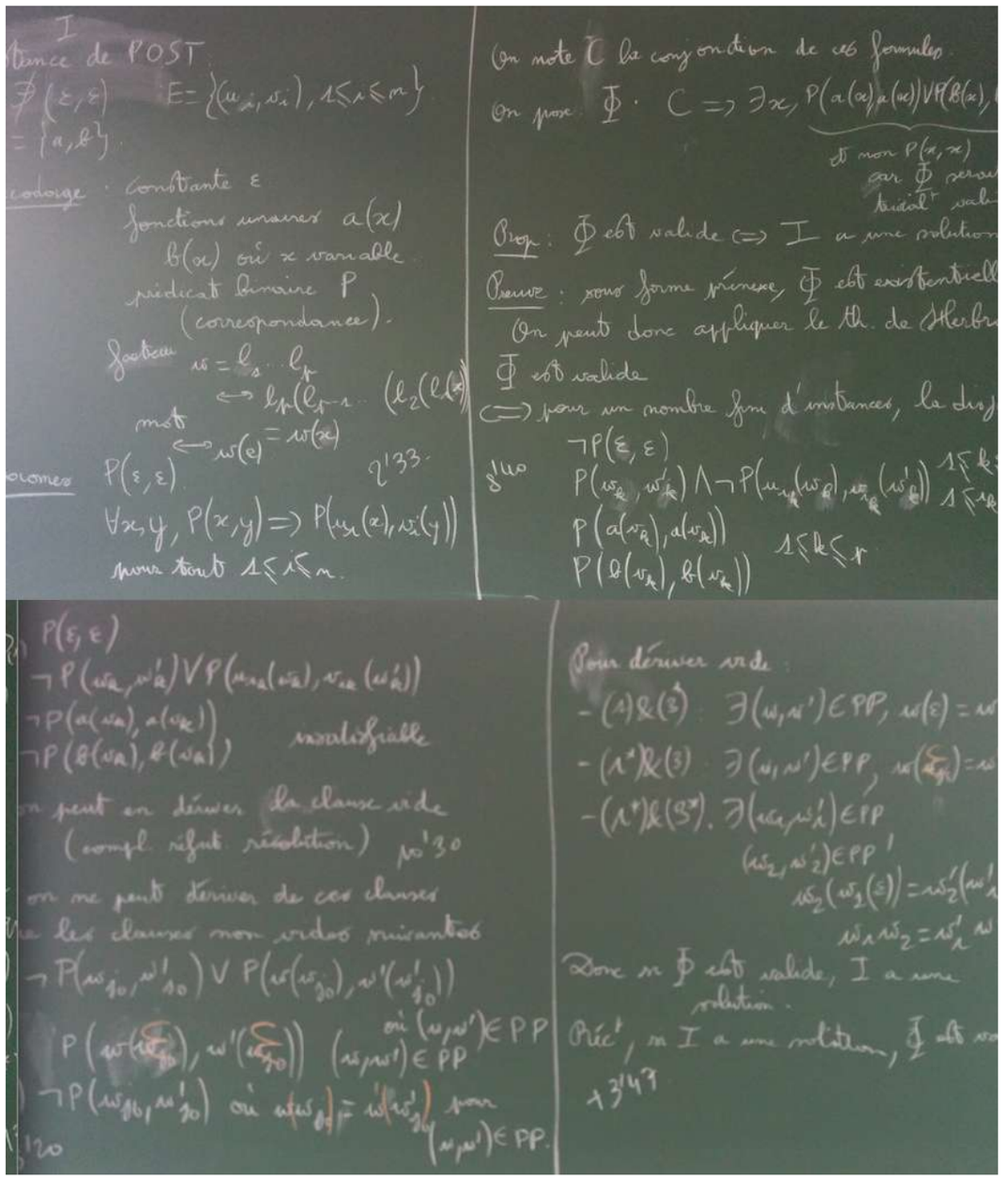
- tous autres documents
  (articles, polys) **interdits**

901 Structures de données. Exemples et applications.
903 Exemples d'algorithmes de tri. Correction et complexité.
907 Algorithmique du texte. Exemples et applications.
909 Langages rationnels et automates finis. Exemples et applications.
912 Fonctions récursives primitives et non primitives. Exemples.
913 Machines de Turing. Applications.
914 Décidabilité et indécidabilité. Exemples.
915 Classes de complexité. Exemples.
916 Formules du calcul propositionnel : représentation, formes normales, satisfiabilité. Applications.
918 Systèmes formels de preuve en logique du premier ordre. Exemples.
921 Algorithmes de recherche et structures de données associées.
923 Analyses lexicale et syntaxique. Applications.
924 Théories et modèles en logique du premier ordre. Exemples.
925 Graphes : représentations et algorithmes.
926 Analyse des algorithmes : complexité. Exemples.
927 Exemples de preuve d'algorithme : correction, terminaison.
928 Problèmes NP-complets : exemples et réduction.
929 Lambda-calcul pur comme modèle de calcul. Exemples.
930 Sémantique des langages de programmation. Exemples.
931 Schémas algorithmiques. Exemples et applications.
932 Fondements des bases de données relationnelles.

Leçon: 2. passage

Durée totale: 50 min.

- Présentation du **plan**: 6 min.
- Le jury choisit un de vos développements
- Présentation du **développement**: 15 min.
- Les temps sont **stricts**
- Questions et dialogue



Thème 1: algorithmique fondamentale

- **Thomas Chatain, Stefan Schwoon**
  ~16 séances

- **901**: structures de données
  exemples et applications
  903: exemples d'algorithmes de tri
  correction et complexité
  921: algorithmes de recherche et
  structures de données associées
  925: graphes
  représentations et algorithmes
  926: analyse des algorithmes
  complexité, exemples
  927: exemples de preuve d'algorithme
  correction, terminaison
  931: schémas algorithmiques
  exemples et applications

Algorithme 18: Algorithme de Kruskal

Données : Un graphe simple pondéré $G = (X, E, W)$
Résultat : Un arbre $A = (X, T)$ recouvrant de poids mi

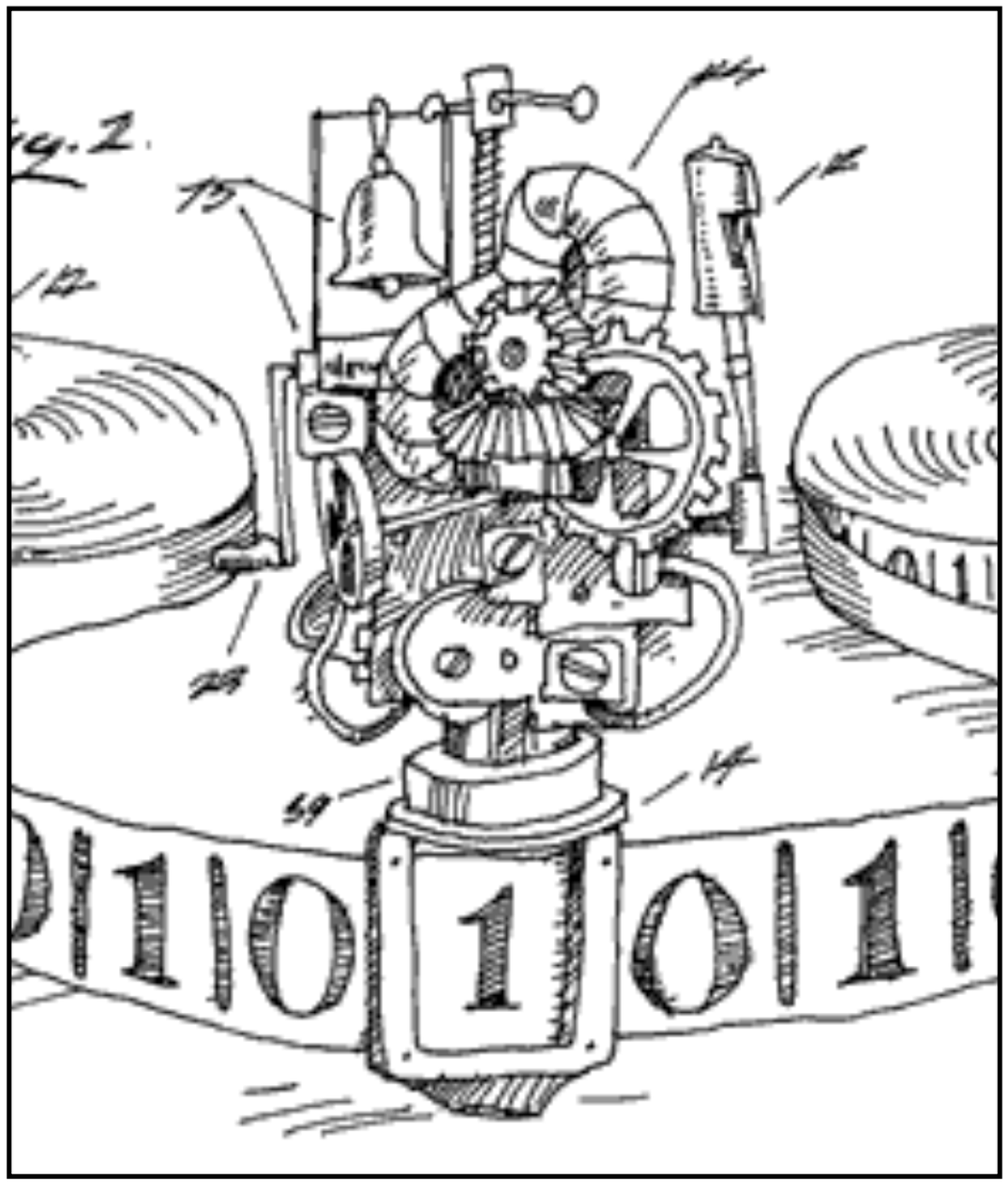
```
// Initialisation
Cout ← 0
T ← ∅
// Boucle principale
tant que |T| < N − 1 faire
    Chercher la plus petite arête [u, v] n'appartenant pas
    si T ∪ [u, v] ne contient pas de cycle alors
        T ← T ∪ [u, v]
        Cout ← Cout + W_{u,v}
    sinon
        Eliminer l'arête [u, v]
retourner A
```

Thème 2: calculabilité, décidabilité, complexité

- **Laurent Rosaz**
  ~11 séances

- **912**: fonctions récursives primitives
  et non primitives
  exemples
  913: machines de Turing
  applications
  914: décidabilité et indécidabilité
  exemples
  915: classes de complexité
  exemples
  928: problèmes NP-complets
  exemples et réduction

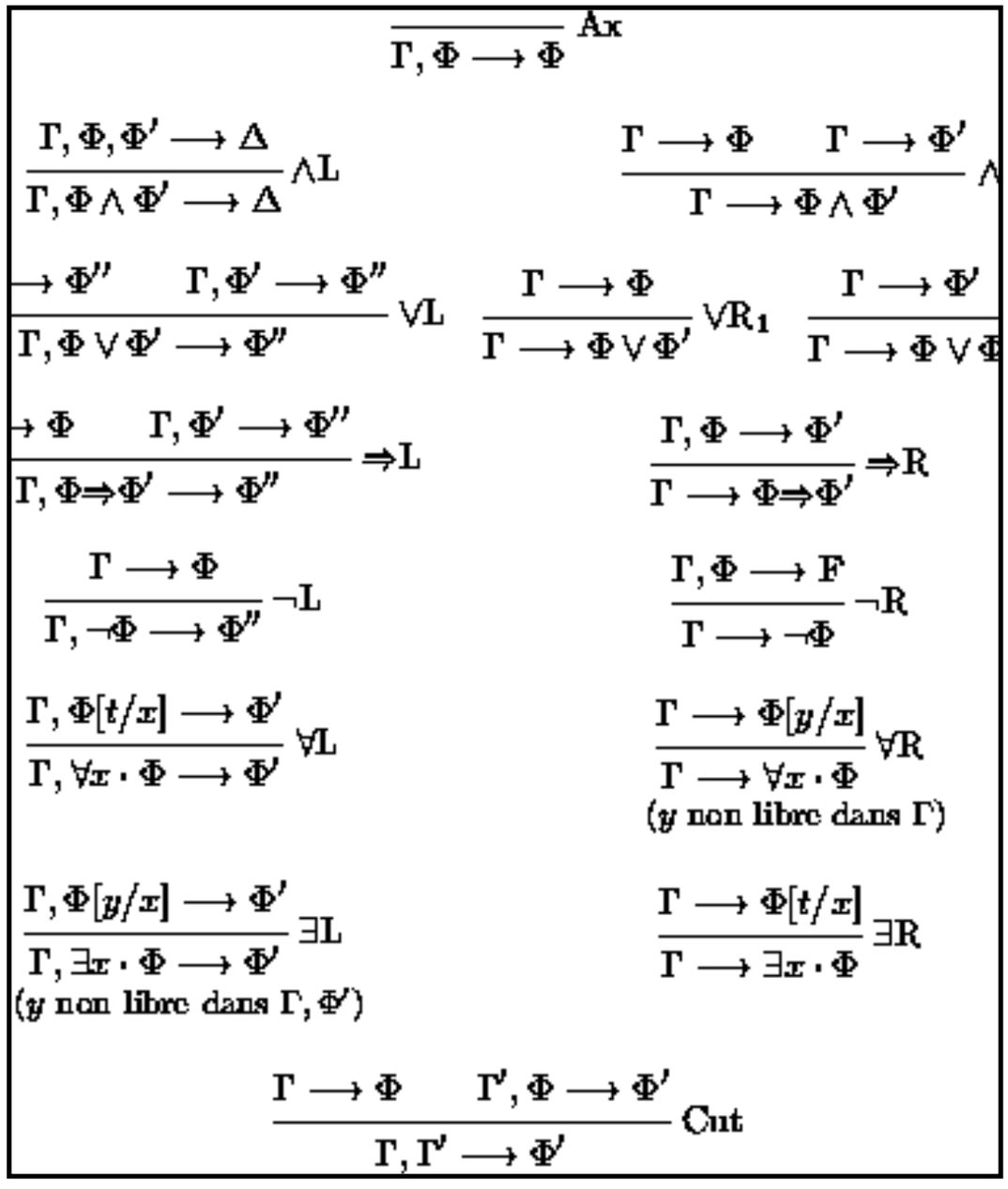
(+ 929: lambda-calcul: voir logique)



Thème 3: logique et démonstration

- **David Baelde, Frédéric Blanqui, Evelyne Contejean**
  ~ 14 leçons

- **916**: formules du calcul propositionnel
  représentation, formes normales, satisfiabilité
  applications
- **918**: systèmes formels de preuve en logique du premier ordre
  exemples
- **924**: théories et modèles en logique du premier ordre
  exemples
- **929**: lambda-calcul pur comme modèle de calcul
- **932**: fondements des bases de données relationnelles



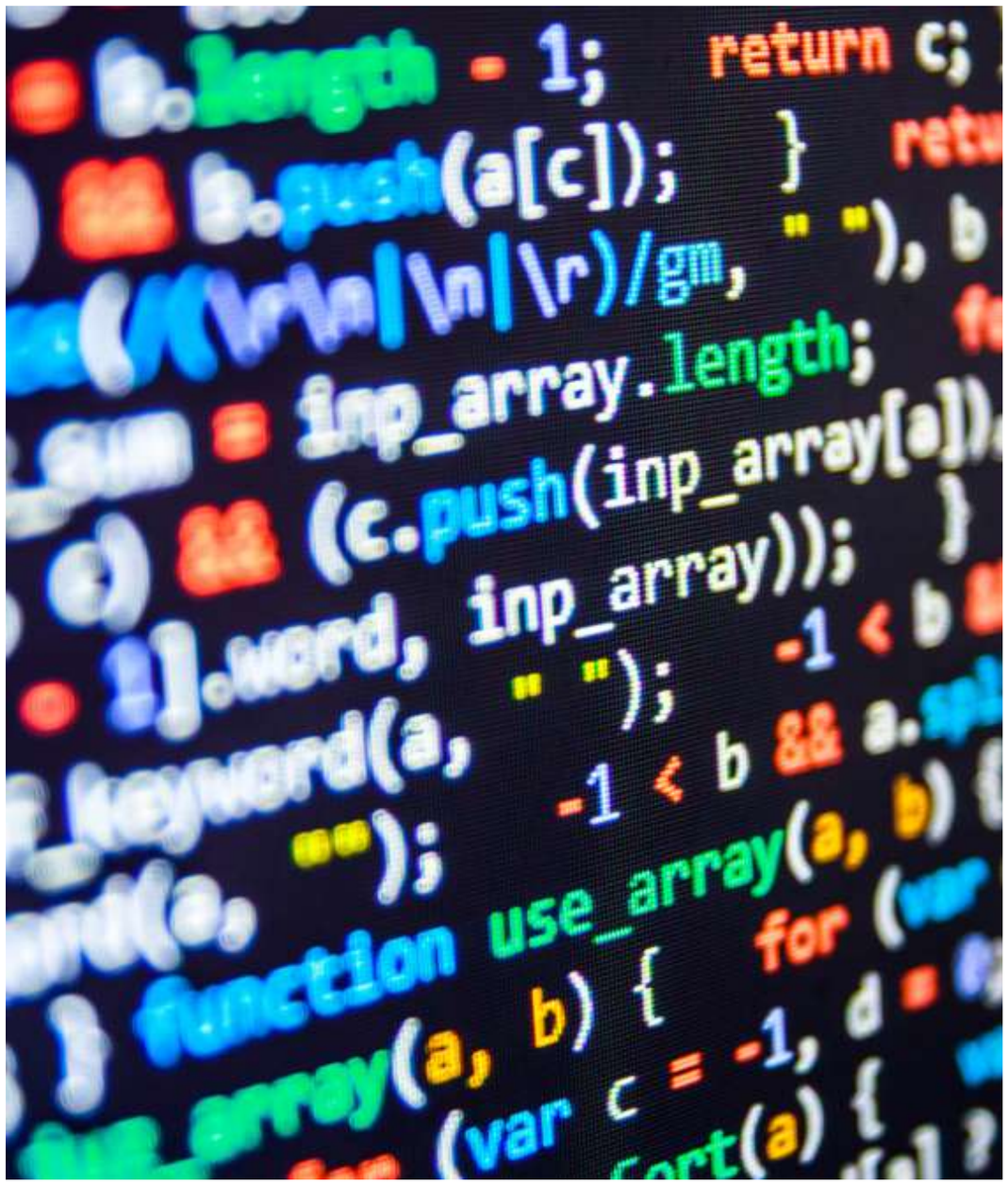
Thème 4: langages formels (ou non)

- **Paul Gastin, David Baelde**
  ~10 leçons

- **907**: algorithmique du texte
  <u>exemples</u> et <u>applications</u>
  909: langages rationnels
  et automates finis
  <u>exemples</u> et <u>applications</u>
  923: analyses lexicale
  et syntaxique
  <u>applications</u>
  930: sémantique des langages
  de programmation
  <u>exemples</u>

Modélisation et programmation

- Thomas Chatain
  vous en parle à partir de lundi
  5 octobre.

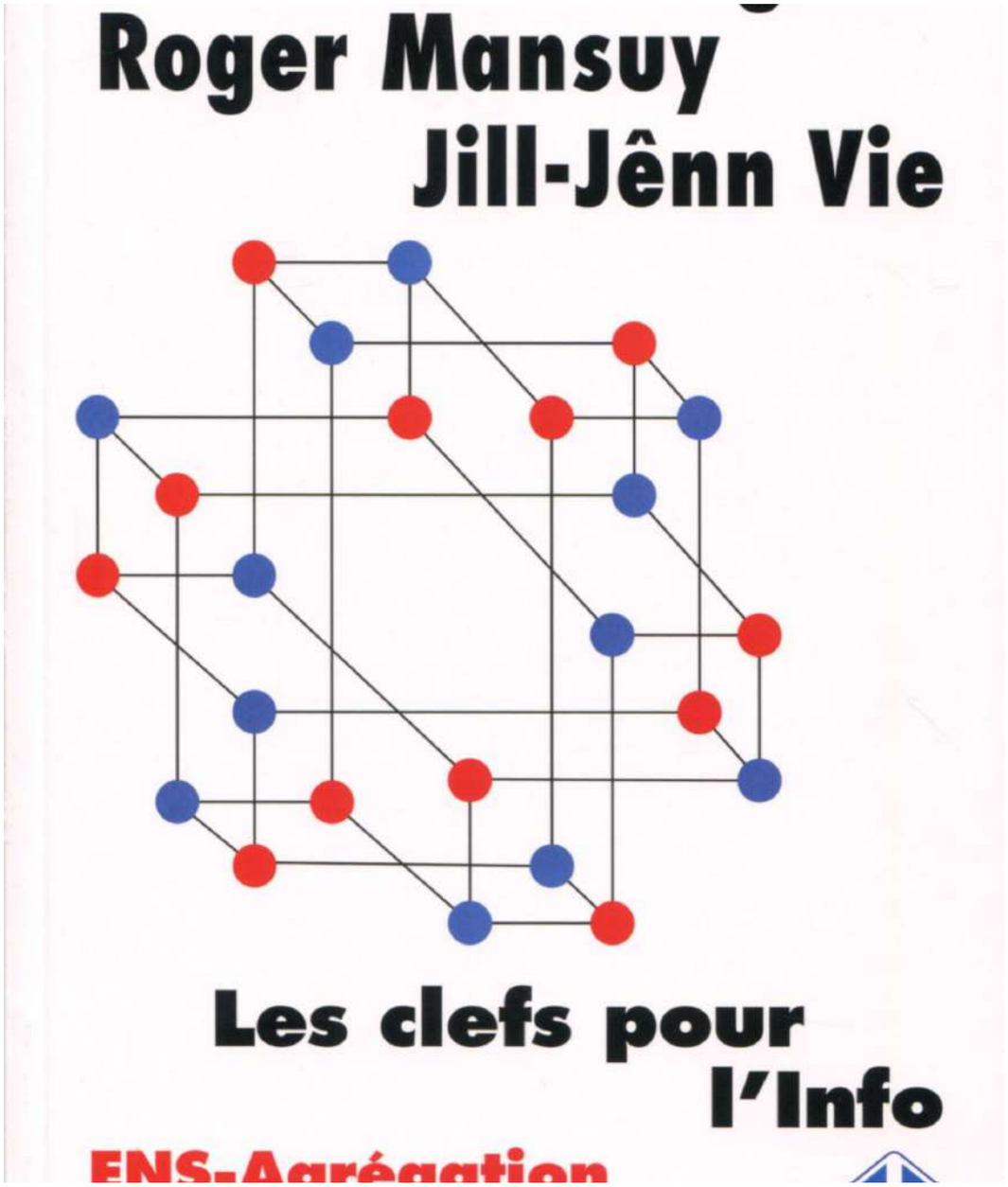


Support

- Plusieurs sites pour agrégatifs, et des livres!

- Avec des exemples de plans, de développements, etc.

- **Attention**:
  ne pas les suivre aveuglément
  votre travail doit être **personnel**.



Quelques commentaires

- Vous serez tentés de faire du **recyclage**: par ex., faire la NP-complétude de SAT dans les leçons 913, 915, 916, 928.
  Pourquoi pas? Le jury s'est adapté. Mais l'**angle** sous lequel vous aborderez le développement devra être différent selon la leçon.

- Faites au **bon niveau**: inutile de présenter un algo. super fin de minimisation d'automate fini si vous ne savez pas le faire tourner sur un petit exemple.

- Bétonnez la **base**. Connaissez les définitions, les théorèmes de base, les algos fondamentaux sur le bout des ongles.

- **Rigueur** mathématique. Absolue! Pas d'approximation.

- **Pédagogie**. Vous êtes les profs, maintenant!

Conclusion

- Beaucoup de travail… surtout en option info (peu de partage avec le reste des maths!)

- Risque de faire tout ça pour rien, comme en 2020 :–(
  ⇒ assurez les bases à l'écrit!

- Mais un beau diplôme à la fin :–)

- Questions?