Mise à jour: 30 mai 2012

Quelques idées pour monter un cours d'introduction à l'informatique en Terminale S

[2ème partie du cours n°5 d'Architecture/Réseaux pour Professeurs de Lycée, 2011-2012]

Nicolas Bredèche

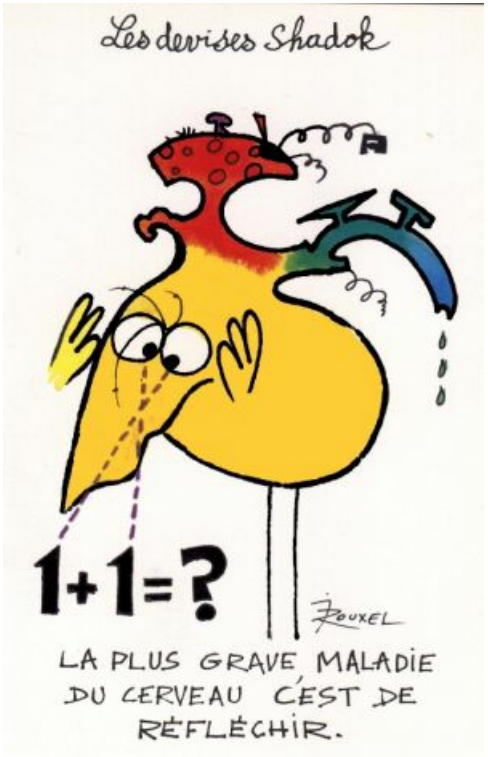
Maître de Conférences HDR

Université Paris-Sud

Equipe **TAO** (LRI, INRIA, CNRS)
Machine Learning & Optimization

Laboratoire de Recherche en Informatique
LRI, UMR 8623

bredeche@lri.fr
http://www.lri.fr/~bredeche



Avertissement: le contenu de cette présentation n'a pour but que de donner des pistes, et ne constitue en aucun cas un guide pratique à prendre au pied de la lettre. En particulier, je n'ai pas forcément pris en compte toutes les contraintes imposées.

- L’informatique
  - une science
  - une technologie

Mon retour d'expérience

- Sources
  - Public non-informaticien
  - «Jeunes» étudiants (L2)
- Profils
  - Diversité des profils:
    - 90% des interventions viennent de 10% qui sont bons
    - les 10% qui sont bloqués... sans raison
- (Mes) conclusions
  - Application dirigée, via des TD
  - Autonomie, via des projet

- Message : de la mise en pratique

Sur le web

google: *teaching computer science for highschool*

remarque: l'enseignement de l'informatique est très développé outre-atlantique (de la maternelle à la terminale)

- http://imej.wfu.edu/articles/2001/2/04/
- ACM «**computer science for highschool**»: http://csta.acm.org/
- http://www.cs4hs.com
- http://www.ehow.com/how_6602115_teach-computer-science-high-school.html (une interview intéressante)
- http://www.techflash.com/seattle/2010/08/qa_lessons_in_computer_science_and_the_future_of_the_industry.html
- http://blogs.technet.com/b/microsoft_in_education/archive/2012/05/17/what-should-the-high-school-cs-curriculum-look-like.aspx
- http://teacher.tchcvs.tc.edu.tw/mhtsai/essay/high-school-program.pdf
- etc.

Message : de la mise en pratique

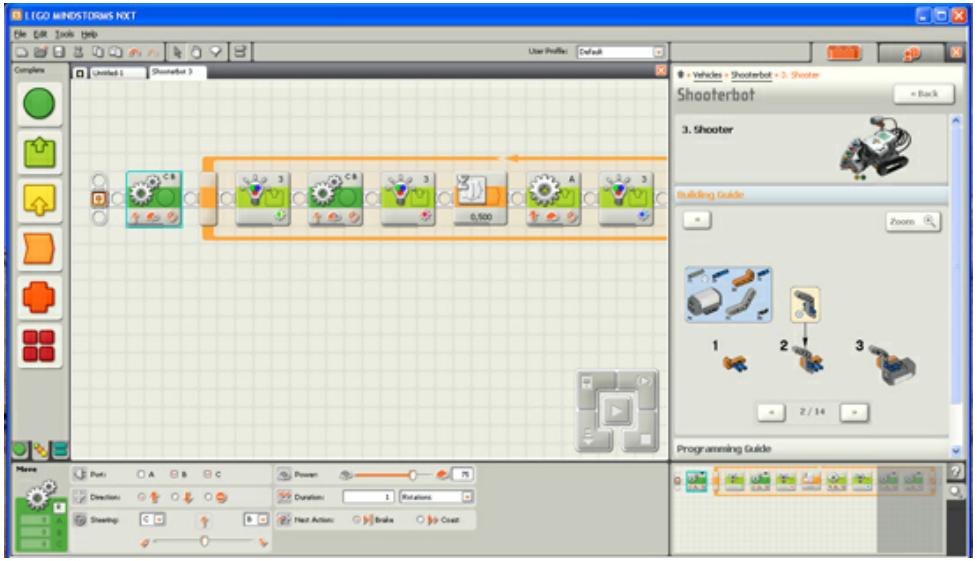
Quelques outils

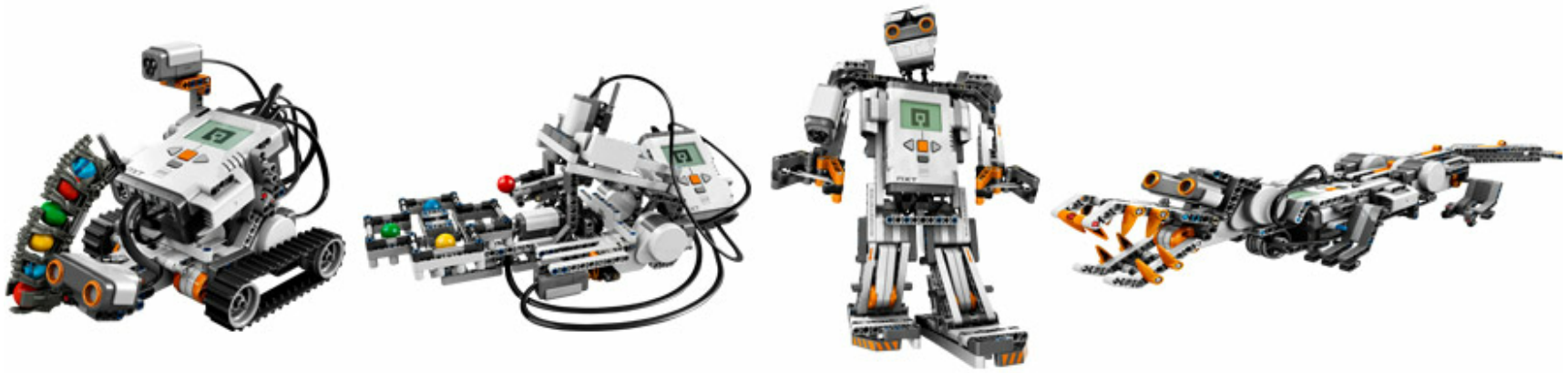
Outils : programmation

- Python : pédagogique, facile d'accès et puissant
- Processing : pédagogique, graphique, mais spécialisé
- Puredata : idéal pour contrôler un périphérique (ex.: Arduino)
- Langages avancés: puissants, mais prise en main difficile
  - Java, C++, C#, Visual Basic, etc.
- Langages graphiques: limités mais pédagogiques (et fait pour vous?)
  - Scratch (MIT), Javaschool (INRIA)

(je choisirai probablement Python, mais ça se défend... et ça dépend du programme que vous choisirez)

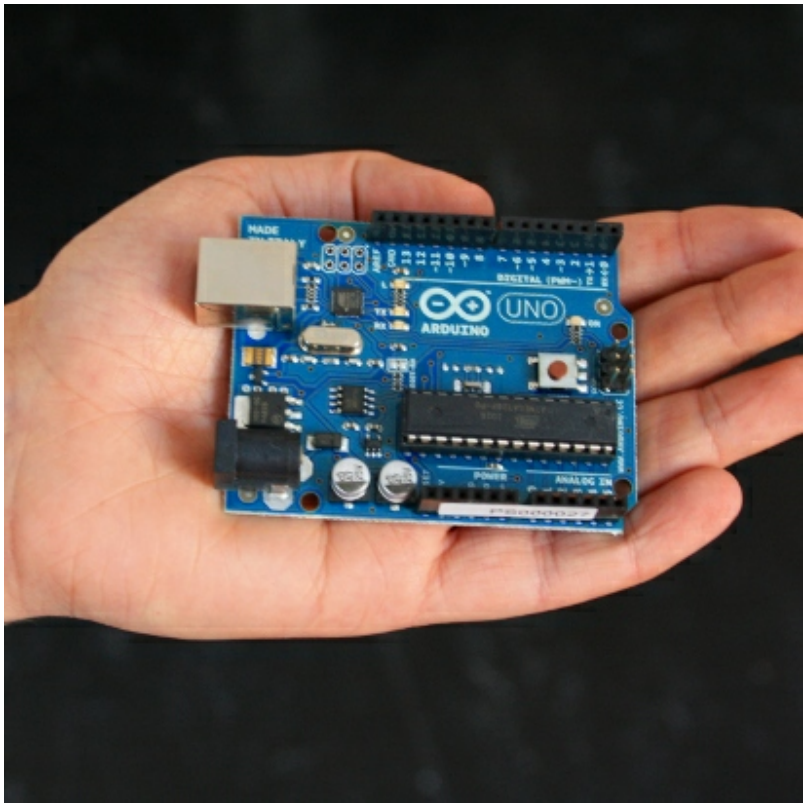
Outils: robotique





Lego Mindstorms
300e

Outils : électronique



Arduino (ou e-block)
à partir de 20e

Organisation du cours (proposition)

AVERTISSEMENT

Lycéens (ou vague souvenir que j'en ai)

je ne connais pas le public de Lycée, les contraintes, etc.

mais... j'enseigne l'informatique depuis 10+ ans à l'Université

...ce qui suit n'est donc qu'une *tentative* pour me projeter à votre place...

Approche pour faire un cours

- Question à laquelle je tente de répondre: faire un cours pour...
  - ...25-30 séances
  - ...divisé en 5 blocs de 5-6 semaines (entre les vacances)
  - ...à raison de 2h par semaine
  - ...devant un public de lycéens en terminal S
- Sachant que je dois...
  - ...donner (au moins) une note individuel
  - ...(et des notes collectives)

Les contraintes du programme

- Contenu obligatoire
  - Représentation de l'information
  - Algorithmes
  - Langage de programmation
  - Architecture des ordinateurs
  - Informatique et libertés
  - Réseaux (en option, sauf l'essentiel)
  - Robotique (en option)
- Constat
  - on ne pourra pas tout faire en détail...
  - ...mais on peut choisir un fil directeur (qui va constituer l'essentiel des cours/TD)
    - ... et traiter le reste lors d'interventions ciblées (exposés, travail à la maison)

Questions à (se) poser

- Deux approches
  - en profondeur: enseigner un point précis sur 50h
    - programme proposé : la programmation
    - pour: fondamental, incrémental, aspect pratique
    - contre: un aperçu réducteur (mais on peut compenser par des exposés courts et ciblés)
  - en largeur: 3 ou 4 thèmes
    - programme proposé : web, programmation, électronique
    - pour: bon aperçu de plusieurs aspects
    - contre: nombre d'heures insuffisants

(dans la suite, je choisis l'option 1)

Mes choix (1/2)

- Organisation générale
  - découpage en «blocs» de 6 semaines chacun (entre les vacances)
  - sur machine, systématiquement
  - un seul langage de programmation pour toute l'année (c'est important)
  - changement des groupes à chaque nouveau «bloc»
- Pédagogie
  - La partie cours/td/projet: apprendre à programmer
  - La partie exposé/rapport: ouverture sur l'informatique
- Evaluation
  - sur projet (par groupe de 3?)
  - sur un exposé ou rapport (individuel?)

Mes choix (2/2)

- Découpage d'un bloc de 6 semaines
  - 1. cours/TD (2h) -- orienté technique + mise en pratique
  - 2. cours/TD (éventuellement sur un sujet différent) + présentation du projet
  - 3, 4, 5. projet
  - 6. soutenance (par groupe, en entretien plutôt que grand oral)

- Notations
  - Contrôle continu:
    - Chaque bloc donne lieu à une note de projet par soutenance sur machine (sauf un, réservé pour le Bac, avec les modalités de notation imposés)
  - Note du bac:
    - Un bloc fait l'objet d'une soutenance particulière, lors du bac
    - Plutôt en fin d'année scolaire (avant pâques, pour éviter le conflit avec les révisions des autres matières?)

Sur les cours/TD

- Contenu essentiel
  - Représentation simple des données
  - Variables et types de données
  - Assignation et calculs simples
  - Boucles (for, while)
  - Fonctions, récursivités
  - Tableaux
  - Structures de décision (if-then-else, switch-case)
  - *Objets* *(statiques, sans création) -- optionnel, probablement pas fait.*
- Contenu additionnel (qui peuvent faire l'objet de mini-interventions, de sujets de recherche, d'exposés d'élèves, etc.)
  - cours sur des sujets précis dépendant du projet, ou non
  - robotique autonome, automates cellulaires, ..., CNIL, ...

fil directeur: apprendre à programmer

parce que:
(1) cela touche l'algorithmique, la programmation, la représentation de l'information (dans le programme)
(2) c'est la meilleure initiation au mode de pensée informatique (à mon avis en tout cas), et à l'apprentissage d'un raisonnement séquentiel rigoureux.

Sur les projets

- Fil directeur:
  - Apprendre à programmer
  - Visualiser le résultat
  - Résultat «dynamique»
- Sujets choisis (ordre chronologique, par bloc)

HTML?
cas particulier: il s'agit d'une entrée en matière facile à appréhender, qu'on préférera mettre en tout début d'année. (un conseil: évitez PHP, redondant avec la suite en terme d'apprentissage de la programmation).

rentrée
  - HTML

toussaint
  - Programmer des automates cellulaires (un jeu de la vie)

Noël
  - Programmation graphique et récursivité (p.ex. tracer des fractales IFS)

hiver
  - Programmer un monde avec des proies-prédateurs (+ feu de forêt?)
    - cf. TP ... OU ... monde de cases (2D) ... OU ... arduino/mindstorms
    - un sujet synthétique qui fera office de note au Bac (soutenance BAC)

pâques
  - Programmer un robot en simulation (cf. véhicules de braitenberg)
    - à partir de pâques, et en // du projet

fin de l'année

Sur les rapports

- Quelques idées de thèmes
  - Loi informatique et libertés (cf. site le CNIL, wikipedia)
  - Internet: historique, structure
  - Sécurité et cryptographie
  - Historique des machines a calculer / ordinateur
  - Calcul haute-performance
  - Intelligence Artificielle
  - Intelligence Ambiante
  - ...
- Méthodologie
  - objectif: **apprendre à chercher et comprendre l'information**
  - apprendre à **citer ses sources** et à **vérifier les informations**
  - présentation orale si le temps dans le calendrier.

Bon courage!