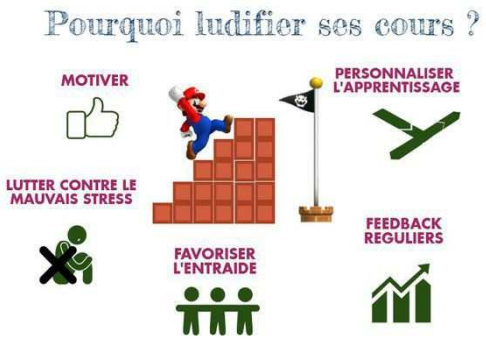
Université Libre de Bruxelles

Département d'Informatique

INFO-F530 Computer science seminar

Rapport du

Séminaire 6



Utilisation de plateformes ludiques de programmation pour motiver les apprenants

(par Sébastien Hoarau – 24 mai 2018)

Olivier Pirson – opi@opimedia.be
Master 2

4 juin 2018
(Petites corrections 5 juin 2018)

ULB

Résumé de la présentation par Sébastien Hoarau

Depuis une quinzaine d'années les étudiants de l'Université de la Réunion sont de moins en moins motivés par les énoncés classiques d'exercices de programmation notamment ceux très mathématiques. Ces cinq dernières années, de nombreuses plateformes ludiques d'apprentissage de la programmation ont vu le jour offrant la possibilité de proposer autre chose aux jeunes programmeurs. Que peut-on faire de ces outils dans le cadre d'un enseignement universitaire ? [Car+17]

(Illustration de la page de titre tirée de [Lor16].)

Table des matières

1 Provide an outline of the content of the seminar in your own words.

Face au constat du déclin de la motivation et des résultats des élèves de premières années universitaire abordant l'apprentissage de la programmation, une possible solution est de remplacer les exercices classiques (que ce soit sur ordinateur ou carrément sur papier) par l'usage de plateformes ludiques.

Cela peut participer de la notion de cours inversé. Les élèves suivent eux-mêmes des MOOCs par exemple, puis profitent de la présence des professeurs de façon plus interactive.

Une plus grande autonomie pourrait permettre aux élèves de s'adapter à leur niveau initial hétérogène.

Peut-être qu'il serait sage de parvenir à réorienter ceux qui n'ont pas le niveau (notamment logico-mathématique) dans le but de les aider à rattraper leur retard pour leur permettre ensuite d'apprendre la programmation. Cela pourrait aussi éviter que d'autres qui pourraient réussir ne soient tenter d'abandonner.

Un rapide aperçu des notions à connaître pour débuter la programmation devrait permettre de rassurer les élèves. Car elles ne sont pas si nombreuses, comparées à d'autre domaine, telle que la mathématique.

Le séminaire est un retour d'expérience sur l'utilisation des deux plateformes en ligne CodinGame [Cod] et CheckIO [Che], accompagné d'une démonstration de ce qu'elles permettent et leurs limitations.

CodinGame propose des résolutions de puzzles, selon plusieurs modes allant de la résolution en solo à la confrontation en groupe, pouvant être privé. C'est aussi une plateforme qui peut être utilisée pour évaluer les capacités de programmeurs dans un but de recrutement. C'est la plus "sexy" des deux, avec des problèmes proposant des animations. Des puzzles tirés au hasard permettent de s'initier aux concepts de base tels que les instructions conditionnelles, les boucles, types de données liste, dictionnaire, etc. Elle propose une grande variété de langage de programmation, dont Python. Une limitation est le manque de feedback de la plateforme lorsque l'étudiant coince sur un problème.

CheckIO est plus austère mais possède elle des possibilités utiles à l'enseignement, telle que le partage de solution au sein d'une classe. Elle fut utiliser comme base de travaux pratiques.

Le résultat de ces expériences n'a pas encore été formalisé.

Alors que les étudiants sont informés qu'en moyenne ceux qui sont présents aux cours obtiennent la moyenne, et que les autres ont en moyenne une moyenne catastrophique, cela ne semble pas inciter le grand nombre à participer. Prenons garde toutefois à ce que cette corrélation ne signifie pas forcément qu'il y ait une causalité.

Il ressort que les étudiants ne font pas suffisamment de petits exercices pour pouvoir ensuite parvenir à accomplir la gymnastique nécessaire à la modélisation et la création d'une solution tout en manipulant la syntaxe du langage utilisé. C'est d'un coup trop d'information à devoir gérer sans suffisamment d'entraînement préalable, malgré une compréhension correcte de problèmes simples.

Un dilemme est que procéder étape par étape sur chaque concept est ennuyeux alors que les énoncés attirants réclament d'un coup trop de notions.

Les étudiants ont des difficultés à progresser de l'application des concepts demandés explicitement à parvenir à y penser le moment venu et les appliquer pour résoudre un problème donné. Une difficulté à décomposer un problème en étapes élémentaires malgré l'insistance sur le principe d'analyse descendante. Une difficulté à passer du réel au modèle et inversement.

> « *[...] au lieu de ce grand nombre de préceptes dont la logique est composée, je crus que j'aurais assez des quatre suivants, pourvu que je prisse une ferme et constante résolution de ne manquer pas une seule fois à les observer.*
>
> *Le premier était de ne recevoir jamais aucune chose pour vraie que je ne la connusse évidemment être telle ; c'est-à-dire, d'éviter soigneusement la précipitation et la prévention ; et de ne comprendre rien de plus en mes jugements, que ce qui se présenterait si clairement et si distinctement à mon esprit, que je n'eusse aucune occasion de le mettre en doute.*
>
> *Le second, de diviser chacune des difficultés que j'examinerais, en autant de parcelles qu'il se pourrait, et qu'il serait requis pour les mieux résoudre.*
>
> *Le troisième, de conduire par ordre mes pensées, en commençant par les objets les plus simples et les plus aisés à connaître, pour monter peu à peu, comme par degrés jusques à la connaissance des plus composés ; et supposant même de l'ordre entre ceux qui ne se précèdent point naturellement les uns les autres.*
>
> *Et le dernier, de faire partout des dénombrements si entiers et des revues si générales, que je fusse assuré de ne rien omettre.*
>
> *Ces longues chaînes de raisons, toutes simples et faciles, dont les géomètres ont coutume de se servir, pour parvenir à leurs plus difficiles démonstrations, m'avaient donné occasion de m'imaginer que toutes les choses, qui peuvent tomber sous la connaissance des hommes, s'entre-suivent en même façon et que, pourvu seulement qu'on s'abstienne d'en recevoir aucune pour vraie qui ne le soit, et qu'on garde toujours l'ordre qu'il faut pour les déduire les unes des autres, il n'y en peut avoir de si éloignées auxquelles enfin on ne parvienne, ni de si cachées qu'on ne découvre.* »
>
> — René Descartes, *Discours de la méthode*, [Des37]

2 Which were the major points discussed by the speaker ? List and explain their importance.

— Manque de prérequis.

Sans une bonne acquisition des notions logico-mathématique élémentaires il est difficile de raisonner correctement sur les problèmes et de les résoudre. Il y a également des idées fausses de ce qu'est l'informatique, une confusion fréquente avec la bureautique. Il est devenu nécessaire d'instaurer un cours d'informatique en secondaire. Et des cours de manipulation d'ordinateur au préalable.

— Difficulté de modélisation et manque d'entraînement.

Même lorsque les élèves parviennent à manipuler les notions séparément, il leur est difficile de les pratiquer ensembles pour parvenir à modéliser un problème réel ou pour découper un problème en éléments plus élémentaires.

3 For which discipline in computer science is the topic relevant and why (explain) ?

Ce séminaire touche l'apprentissage de la programmation, à un niveau de base. Très peu d'algorithmique est impliquée.

Cet apprentissage de base est néanmoins un passage obligé pour une réelle formation informatique et pour presque toutes les autres disciplines informatiques.

4 Suggest two or three articles discussing similar work (not necessarily published by the speaker). Explain why they are related.

— *Ludification pour la motivation en apprentissage de la programmation* [Hoa17]

Un article sur l'expérience relatée dans ce séminaire, par l'intervenant lui-même.

— *A social gamification framework for a K-6 learning platform* [SRV13]

Un article sur une étude alors en cours analysant les résultats de la ludification, dans un contexte d'apprentissage plus général.

5 Propose two scientific questions relevant for the topic discussed in this seminar.

— Que disent les sciences de l'éducation sur la peur de la mathématique et les moyens d'en enseigner les bases relativement à divers profils d'étudiants ?
— Y a-t-il des études corrélant les connaissances logico-mathématiques avec les capacités de programmation ? Idéalement en distinguant divers types de projets à programmer.

6 Would you like to know more about this field ? Do you have particular critiques about the scientific content ? Provide a brief motivation for your answers.

Ce séminaire n'abordait pas directement de contenu scientifique.

J'avais déjà participer à quelques challenges, notamment sur HackerRank [Hac], une autre plateforme. J'aime bien ce genre de défi ; on en trouve de tout niveau.

Placer l'aspect ludique de l'autre côté peut aussi avoir son intérêt. Le MOOCs *An Introduction to Interactive Programming in Python* [Gre+] [War+] est une introduction à la programmation qui progressivement permet de réaliser des petits jeux de complexité croissante.

Références

[Car+17] Jean Cardinal, Martine Labbé, Tom Lenaerts et Olivier Markowitch. **Computer Science Seminar INFO-F-530**. 2017. URL : https://web.archive.org/web/20180601005154/https://www.ulb.ac.be/di/map/tlenaert/Home_Tom_Lenaerts/INFO-F-530.html (cf. p. 1).

[Che] CheckiO. **CheckiO**. URL : https://checkio.org/ (cf. p. 3).

[Cod] CodinGame. **CodinGame**. URL : https://www.codingame.com/ (cf. p. 3).

[Des37] René Descartes. **Discours de la méthode**. 1637. URL : https://fr.wikisource.org/wiki/Discours_de_la_m%C3%A9thode (cf. p. 4).

[Gre+] John Greiner, Stephen Wong, Scott Rixner et Joe Warren. **An Introduction to Interactive Programming in Python (Part 1)**. Coursera. URL : https://www.coursera.org/learn/interactive-python-1 (cf. p. 5).

[Hac] HackerRank. **HackerRank**. HackerRank. URL : https://www.hackerrank.com/ (cf. p. 5).

[Hoa17] Sébastien Hoarau. **Ludification pour la motivation en apprentissage de la programmation**. Juin 2017. URL : https://www.researchgate.net/publication/317688155_Ludification_pour_la_motivation_en_apprentissage_de_la_programmation (cf. p. 5).

[Lor16] Laetitia Lormier. **La ludification : « Jouer est une récompense en soi »**. Sciences de la vie et de la Terre. 17 juin 2016. URL : http://svt.ac-creteil.fr/?La-ludification-Jouer-est-une-recompense-en-soi (cf. p. 1).

[SRV13] Jorge Simões, Rebeca Díaz Redondo et Ana Fernández Vilas. **A social gamification framework for a K-6 learning platform**. In : *Computers in Human Behavior*. Advanced Human-Computer Interaction 29.2 (1er mar. 2013), p. 345–353. DOI : 10.1016/j.chb.2012.06.007 (cf. p. 5).

[W D72] Edsger W. Dijkstra. **Le programmeur modeste**. 1972. URL : https://web.archive.org/web/20160303171900/http://old.adrahon.org/le-programmeur-modeste.html (cf. p. 7).

[War+] Joe Warren, Scott Rixner, John Greiner et Stephen Wong. **An Introduction to Interactive Programming in Python (Part 2)**. Coursera. URL : https://www.coursera.org/learn/interactive-python-2 (cf. p. 5).

≪ Il a à voir avec l'influence des outils que nous essayons d'utiliser sur nos habitudes de penser. Je constate l'existence d'une tradition culturelle, dont les racines remontent sans doute à la Renaissance, d'ignorer cette influence, de considérer l'esprit humain comme le maître suprême et autonome de ses artefacts. Mais si je commence à analyser mon mode de penser et celui de compagnons humains, j'arrive, que cela me plaise ou non, à une conclusion totalement différente, à savoir que les outils que nous tentons d'utiliser et le langage ou la notation que nous utilisons pour exprimer ou enregistrer nos pensées sont les facteurs majeurs déterminant tout simplement ce que nous pouvons penser ou exprimer ! L'analyse de l'influence qu'ont les langages de programmation sur les modes de penser de leurs utilisateurs et la reconnaissance qu'aujourd'hui, la matière grise est de loin notre ressource la plus rare, ensemble, nous donne une nouvelle série de repères pour comparer les mérites relatifs de différents langages de programmation. Le programmeur compétent est pleinement conscient de la taille strictement limitée de son crâne ; il approche donc la tâche de la programmation en toute modestie, et, entre autres choses, il évite les astuces ingénieuses comme la peste.
[. . .]
Je n'ai pas connaissance d'une autre technologie couvrant un ratio de 10^{10} *ou plus : l'ordinateur, en raison de son extraordinaire rapidité, semble pour la première fois nous fournir un environnement où des artefacts hautement hiérarchisés sont à la fois possibles et nécessaires. Ce défi, à savoir la confrontation avec le travail de la programmation, est si unique que cette nouvelle expérience peut nous apprendre beaucoup sur nous-mêmes. Elle devrait approfondir notre compréhension du processus de conception et de création, elle devrait nous donner un meilleur contrôle sur l'organisation de nos pensées. Si ça n'est pas le cas, à mon avis nous ne méritons absolument pas l'ordinateur ! ≫*

— Edsger W. Dijkstra, *Le programmeur modeste*, [W D72]