# Atelier : Utiliser des stratégies d'instruction explicites dans l'enseignement de la programmation

Olivier Goletti[1,3][0000−0002−1610−4985] et Kim Mens[1][0000−0003−0303−1630]

[1] ICTEAM/INGI, UCLouvain, Louvain-la-Neuve, Belgique
firstname.lastname@uclouvain.be
[2] LIACS, Leiden University, Leiden, Pays-Bas

**Résumé** Trop souvent, on attend de nos élèves qu'ils soient capables de lire du code ou d'en écrire, sans prendre le temps de leur expliciter les étapes clés qui permettent d'y arriver. Dans cet atelier, on vous propose de découvrir quelques stratégies explicites supportées par la recherche qui vont aider les élèves à apprendre à lire et écrire du code via des stratégies simples à mettre en oeuvre de manière plus systématique. Ces stratégies, une fois intégrées dans vos pratiques, permettent aux élèves qui en ont besoin d'être plus guidés dans leur apprentissage.

**Mots clés :** enseignement de l'informatique · exemples résolus · objectifs étiquetés · stratégie d'instruction explicite

## 1 Introduction

Comme l'enseignement de l'informatique prend de l'ampleur, la recherche en didactique également. On enseigne maintenant l'informatique de plus en plus tôt et dans de plus en plus de pays [11].

Notre analyse d'un cours d'introduction à la programmation a mis en évidence l'intérêt d'utiliser des stratégies d'instruction explicite pour diminuer la charge cognitive des étudiants et promouvoir le transfert des apprentissages au sein du cours de programmation. Un besoin de techniques d'instruction explicites a été identifié [2,4] . En effet, les étudiants profitent de plus d'accompagnement [10] en particulier dans le cadre de méthodologies d'enseignement moins guidées [5].

Plusieurs stratégies d'instructions explicites ont déjà été testées récemment pour enseigner la programmation [12,8,1,6]. Nous en avons identifié quatre et les avons appliquées dans les travaux pratiques d'un cours d'informatique de première année d'un programme de bachelier universitaire en Belgique francophone [4,3]. Dans cet atelier on vous partagera nos expériences.

## 2 Contexte atelier

Dans cet atelier, on propose de faire découvrir aux participants quelques stratégies explicites tirées de la littérature scientifique qui vont aider et guider

les élèves dans leurs apprentissages. Les formateurs profitent de leur expérience d'utilisation de ces stratégies d'instruction dans un cours d'introduction à l'informatique de niveau universitaire.

Les stratégies présentées seront les suivantes ; les deux premières certainement et les deux suivantes en fonction de la durée de l'atelier.

1. **Explicit tracing** ou comment exécuter systématiquement un programme à la main avec une représentation écrite de l'évolution de la mémoire. Cette stratégie est basée principalement sur les recherches de Xie et al. [12] ;
2. **Subgoal-labeled worked examples** ou comment, via des exemples résolus avec objectifs étiquetés, mettre en évidence les étapes importantes de réflexion quand on présente et utilise certains concepts de programmation. L'utilisation d'exemples résolus avec objectifs étiquetés a fait l'objet de plusieurs études par Margulieux et al. [7,9,8] ;
3. **Parsons problems** ou comment, à partir de résolution de puzzles de code, travailler la résolution de problème de programmation sans faire écrire tout le code à l'apprenant. Les problèmes de Parsons sont créés à partir d'exemples résolus de programmes dont on a mélangé les lignes. Cette stratégie est inspirée notamment des recherches d'Ericson et al. [1] ;
4. **Explicit problem solving** ou comment guider les apprenants dans la résolution de problème de programmation en leur fournissant des guides métacognitifs de réflexion sur les étapes identifiées de résolution de tels problèmes. Cette stratégies est basée sur les recherches de Loksa et al. [6].

A titre d'illustration, la figure 1 montre un exemple résolu de lecture de fichier en Python annoté d'objectifs étiquetés. On y voit renseignés les étapes génériques de l'écriture dans un fichier ainsi que la résolution d'un exercice contextualisé.

## 3 Méthodologie

Pour chaque stratégie, quelques éléments didactiques seront donnés en référence à la littérature correspondante, leurs objectifs respectifs seront détaillés, des exemples et conseils de mise en oeuvre seront églament proposés.

Des documents seront distribués aux participants pour faciliter la mise en oeuvres des différentes stratégies. Une réflexion conjointe avec les participants est prévue pour voir comment ils peuvent intégrer ces techniques d'instructions dans leur enseignement de la programmation au jour le jour.

Ces stratégies sont relativement simples à mettre en oeuvre et sont systématiques. Ces stratégies, une fois intégrée dans les pratiques d'enseignements, permettent aux élèves qui en ont besoin d'être plus guidés dans leur apprentissage.

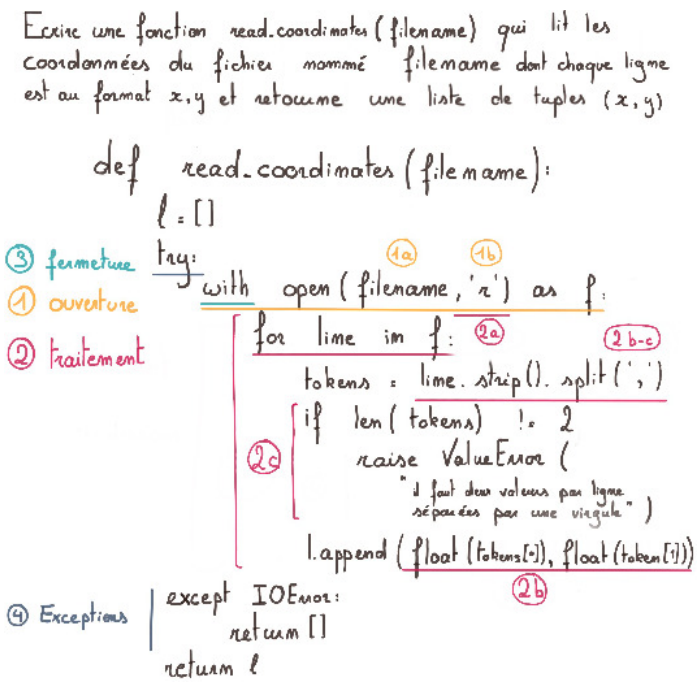


**Figure 1.** Exemple résolu de lecture de fichier en Python annoté d'objectifs étiquetés

## 4 Détails pratiques

**Matériel** Pour cet atelier, l'idéal serait d'avoir un projecteur pour présenter des slides.

**Participants** Cet atelier s'adresse à des enseignants qui travaillent avec de la programmation textuelle principalement. Même s'il peut s'adapter à de la programmation visuelle. Comme l'atelier propose des ressources imprimées, il est bon de connaître le nombre de participants à l'avance mais il n'y a pas vraiment de limite per se.

## Remerciements

Le projet Erasmus+ CAI[3] (Communauté d'apprentissage de l'informatique) dans lequel cette recherche se situe, a été financé avec le soutien de la Commission européenne. Cette proposition d'atelier n'engage que son auteur et la Commission n'est pas responsable de l'usage qui pourrait être fait des informations qui y sont contenues.

3. https://cai.community/

## Références


1. Ericson, B.J., Margulieux, L.E., Rick, J. : Solving Parsons Problems Versus Fixing and Writing Code. In : Proceedings of the 17th Koli Calling International Conference on Computing Education Research. pp. 20–29. Koli Calling '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/3141880.3141895
2. Goletti, O. : En quoi le dispositif mis en œuvre dans le cours d'introduction à l'informatique en BAC1 ingénieur civil basé sur l'apprentissage par problèmes soutient les processus du transfert des apprentissages : l'encodage et l'accessibilité aux connaissances ? Tech. rep., UCLouvain (2019), http://hdl.handle.net/2078.1/245579
3. Goletti, O. : Promoting Learning Transfer in Computer Science Education by Training Teachers to use Explicit Programming Strategies. In : ICER '17. pp. 411–412. ACM, Virtual Event USA (Aug 2021). https://doi.org/10.1145/3446871.3469776
4. Goletti, O., Mens, K., Hermans, F. : Tutors' Experiences in Using Explicit Strategies in a Problem-Based Learning Introductory Programming Course. In : ITiCSE '21. p. 7. ACM Press, Virtual Event, Germany (Jun 2021). https://doi.org/10.1145/3430665.3456348
5. Kirschner, P.A., Sweller, J., Clark, R.E. : Why Minimal Guidance During Instruction Does Not Work : An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching. Educational Psychologist **41**(2), 75–86 (Jun 2006). https://doi.org/10.1207/s15326985ep4102_1
6. Loksa, D., Ko, A.J., Jernigan, W., Oleson, A., Mendez, C.J., Burnett, M.M. : Programming, problem solving, and self-awareness : Effects of explicit guidance. In : Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems. pp. 1449–1461. ACM (2016)
7. Margulieux, L., Catrambone, R., Guzdial, M. : Subgoal Labeled Worked Examples Improve K-12 Teacher Performance in Computer Programming Training. In : Proceedings of the Annual Meeting of the Cognitive Science Society. vol. 35 (2013)
8. Margulieux, L.E., Morrison, B.B., Decker, A. : Design and Pilot Testing of Subgoal Labeled Worked Examples for Five Core Concepts in CS1. In : ITiCSE '19. pp. 548–554. ACM Press, Aberdeen, Scotland Uk (2019). https://doi.org/10.1145/3304221.3319756
9. Morrison, B.B., Margulieux, L.E., Guzdial, M. : Subgoals, Context, and Worked Examples in Learning Computing Problem Solving. In : ICER '11. pp. 21–29. ACM Press (2015). https://doi.org/10.1145/2787622.2787733
10. Sweller, J., van Merriënboer, J.J., Paas, F. : Cognitive architecture and instructional design : 20 years later. Educational Psychology Review pp. 1–32 (2019)
11. Vahrenhold, J., Caspersen, M., Berry, G., Gal-Ezer, J., Kölling, M., McGettrick, A., Nardelli, E., Pereira, C., Westermeier, M. : Informatics Education in Europe : Are We All In The Same Boat ? (2017)
12. Xie, B., Nelson, G.L., Ko, A.J. : An explicit strategy to scaffold novice program tracing. In : SIGCSE TS '49. pp. 344–349. ACM (2018)