Quelles traces écrites pour la programmation ?

Emmanuel Beffara

Laboratoire d'Informatique de Grenoble

IREM de Grenoble

Commission inter-IREM Informatique (C3I)

7ème journée académique sur l'enseignement de l'informatique,
Marseille, 17 mai 2023

Introduction

Introduction

Objectif

Réfléchir aux rôles des **traces écrites** et des **productions écrites** dans l'apprentissage de la programmation.

- partager des pratiques existantes
- préciser le rôle attendu des traces écrites
- identifier des objectifs pédagogiques à poser par écrit

Organisation

1. Tour des pratiques
2. Point sur les compétences
3. Étude de cas

Précisons le sujet

On parle de la **programmation** :

- le savoir-faire que l'on développe par les travaux pratiques et dans les projets de programmation,
- les connaissances associées, générales (techniques de programmation) ou spécifiques au langage utilisé (particularités du langage, bibliothèques),
- les « bonnes pratiques » établies.

Le temps de l'atelier, on met de côté les autres thèmes d'informatique (algorithmique, représentation de l'information, architecture, réseaux…) même si tout est lié.

Tour des pratiques

Trace écrite

Sur la notion de « trace écrite » **en général** :

- Qu'est-ce que c'est ?
- Quelle forme prend-elle ?
- Quel est son rôle ?

15 minutes d'élaboration en petits groupes,
mise en commun et discussion

Trace écrite – contributions de la salle pendant l'atelier

Qu'est-ce que c'est ?

- Ce qu'on fait écrire aux élèves pour qu'ils retiennent
- Tout ce qui s'écrit: en maths, définitions, exemples, propriétés etc exercices
- Enregistrement d'un programme ?
- Support papier ou numérique ?
- Forcément construite par l'élève ou pas ?
- Éléments à retenir: synthèse par le prof ou par l'élève
- Résumé, carte mentale
- Travail sur les erreurs, cheminement pour arriver au résultat
- Rituel, ex en info pour écrire une fonction, on fait comme ci
- Poser une méthode de référence

Quelle forme prend-elle ?

- Trois types: prise de notes, synthèse, à la maison
- Corrections d'activités ou d'exercices
- En techno, travail de recherche, réponse à des questions sur une documentation
- Logigrammes comme intermédiaire entre description informelle et programme
- En maths: synthèse de ce qui a été fait
- Certaines choses restent expliquées oralement sur la base d'exemples
- Document préparé pour la prise de notes, s'il y a un support vidéo ou diaporama
- Certains élèves se font une fiche pour se souvenir de ce qu'ils ont fait à l'occasion d'un TP (fiche réalisée sur ordinateur)
- Carte mentale
- Journal de bord pendant un projet
- Utilisation du notebook style Jupyter

Quel est son rôle ?

- En projet, se souvenir des tâches attribuées, de l'évolution du projet
- Compte-rendu des difficultés rencontrées et de leur résolution
- Expliciter du vocabulaire et des méthodes
- Pouvoir retrouver l'information qu'on n'a plus en tête
- Noter les spécificités de l'outil utilisé (ex en Scratch)
- Outil de révision
- Démarche spiralaire
- Gain de temps (éviter de rechercher une ressource ou un exemple)

Rôles de la trace écrite

Fonctions habituelles de la trace écrite :

- **Institutionnalisation** : mettre des mots (et les mêmes pour tous) sur les notions apprises, conceptualiser
- **Mémorisation** : aider à retenir, identifier un lieu où retrouver une information
- **Réinvestissement** : référence explicite à des savoirs précédents, enrichissement progressif
- **Rédaction** : travail de l'écrit, synthèse, mise en forme
- **Métacognition** : prendre conscience de ses apprentissages

En programmation

Quelles sont vos pratiques pour remplir ces différentes fonctions, dans le cas des savoirs et savoir-faire de programmation ?

- institutionnaliser
- mémoriser
- réinvestir
- rédiger

Les compétences en programmation

Compétences selon les programmes officiels

- analyser et modéliser un problème en termes de flux et de traitement d'informations ;
- décomposer un problème en sous-problèmes, reconnaître des situations déjà analysées et réutiliser des solutions ;
- concevoir des solutions algorithmiques ;
- traduire un algorithme dans un langage de programmation, en spécifier les interfaces et les interactions, comprendre et réutiliser des codes sources existants, développer des processus de mise au point et de validation de programmes ;
- mobiliser les concepts et les technologies utiles pour assurer les fonctions d'acquisition, de mémorisation, de traitement et de diffusion des informations ;
- développer des capacités d'abstraction et de généralisation.

Les compétences en programmation

Une façon de structurer les différentes compétences en jeu :

- **Abstraire** : ignorer les détails non pertinents, définir et utiliser des interfaces de programmation.
- **Décomposer** : structurer un problème complexe en un ensemble de problèmes plus simples équivalent au problème initial.
- **Anticiper** : se mettre en posture de programmeur pour décrire l'enchaînement des opérations, avant le début de l'exécution.
- **Évaluer** : attribuer mentalement une valeur (au sens large) à un programme donné.
- **Généraliser** : repérer et exploiter une régularité dans les données ou les traitements, inférer un problème à partir d'un cas particulier.

Les bonnes pratiques

Des principes qui s'appuient sur ces compétences :

- **Documenter** : bien nommer ses variables et ses fonctions, spécifier précisément les interfaces
- **Tester** : donner des exemples d'attendus pour chaque fonction, écrire des tests systématiques (avant d'implémenter)
- **Être clair** : écrire pour que la structure soit compréhensible, commenter les difficultés seulement quand elles sont inévitables
- **Réutiliser** : reconnaître que la plupart des problèmes ont déjà été résolus par d'autres que nous (et mieux)

Les raisons sont connnues :

- on lit du code bien plus souvent qu'on n'en écrit,
- déboguer est plus difficile que programmer.

Ces principes sont valables en enseignement s'ils sont bien transposés.

Étude de cas

Étude de cas

- Choisir une tâche de programmation classique (sujet de TP)
- Analyser les objectifs d'apprentissage en termes de compétences *de programmation* en jeu
- Donner des critères d'évaluation pour les productions d'élèves
- Écrire un modèle de trace écrite associée (synthèse en fin de TP)

15 minutes d'élaboration en petits groupes,
mise en commun et discussion

Discussion

Quelques pistes à explorer

La programmation est une *production d'écrit* :

- On s'intéresse autant au procédé qu'au résultat
- On écrit pour être lu, le *style* a son importance
- Mais on écrit dans un langage formel

La programmation est une activité *expérimentale*

- Il y a des outils à maîtriser
- Il y a des protocoles à suivre pour de bonnes raisons
- On veut partager des connaissances et du savoir-faire