La Démarche Mrv

— Méthode, Résultat, Vérification —

J. Tisseau, A. Nédélec, M. Parenthoën

Ecole Nationale d'Ingénieurs de Brest
Technopôle Brest-Iroise
CS 73862, 29238 Brest cedex 3, France

Table des matières

1 Introduction

L'objectif de ce document est de présenter la démarche suivie pour répondre aux exercices qui accompagnent le cours d'« Initiation à l'algorithmique » du semestre S1 à l'Ecole nationale d'ingénieurs de Brest (Enib). Cette démarche, dite Mrv (Méthode-Résultat-Vérification), est structurée en 3 étapes :

1. on commence par expliciter la méthode générique qui permet de résoudre des problèmes équivalents à celui qui est posé (étape M comme Méthode) ;
2. on applique ensuite cette méthode générique au cas particulier de l'énoncé pour obtenir le résultat attendu par l'exercice (étape R comme Résultat) ;
3. enfin, on réalise une vérification du résultat obtenu à l'aide de techniques alternatives ou complémentaires connues (étape V comme Vérification).

Pour fixer les idées, nous illustrons d'abord la démarche Mrv par deux exemples « simples » (section 2) : le passage des nœuds « marins » en kilomètres par heure « terrestres » (2.1) et le calcul d'une fonction composée de segments de droite (2.2). Puis nous commentons quelques retours d'expériences (section 3) qui nous ont guidés dans l'élaboration de la démarche Mrv : ne pas se tromper d'objectif (3.1), expliciter l'implicite (3.2) et encourager la rédaction (3.3). Forts de cette expérience, nous précisons ensuite les attendus à chaque étape de la démarche Mrv (section 4) : l'explicitation de la méthode (4.1), l'application de la méthode (4.2) et la vérification du résultat (4.3). Enfin, nous détaillons sa mise en œuvre (section 5) à l'Enib dans le cadre du cours « Initiation à l'algorithmique » (5.1) où elle conduit à une triple évaluation des exercices selon une notation adaptée (5.2) dans le cadre d'un contrôle continu systématique (5.3).

2 Exemples

Les exemples décrits dans cette section sont extraits des « Questionnements de cours » qui complètent le cours d'Informatique S1 à l'Enib. Chaque exemple est structuré ici en cinq paragraphes :

1. l'objectif thématique de l'exercice,
2. l'énoncé de l'exercice,
3. la méthode générique utilisée pour résoudre une famille de problèmes équivalents à celui de l'exercice proposé,
4. le résultat obtenu en appliquant la méthode générique au cas particulier de l'énoncé,
5. la vérification du résultat.

2.1 Des nœuds aux kilomètres par heure

Objectif Mettre en œuvre l'instruction d'affectation.

Enoncé On veut convertir une certaine quantité $n1$ de vitesse exprimée en nœuds (miles nautiques par heure) en la quantité équivalente n_2 exprimée en kilomètres par heure (km/h). Proposer une instruction de type « affectation » qui réalise cette conversion.

Méthode On cherche ici à convertir $n_1 \cdot u_1$ en $n_2 \cdot u_2$ où u_1 et u_2 sont des unités physiques compatibles qui dérivent de la même unité de base u_b du Système international d'unités.

$$\left| \begin{array}{l} u_1 = a_1 \cdot u_b \\ u_2 = a_2 \cdot u_b \end{array} \right. \Rightarrow \left| \begin{array}{l} n_1 \cdot u_1 = n_1 \cdot (a_1 \cdot u_b) = (n_1 \cdot a_1) \cdot u_b \\ n_2 \cdot u_2 = n_2 \cdot (a_2 \cdot u_b) = (n_2 \cdot a_2) \cdot u_b \end{array} \right. \Rightarrow \frac{n_1 \cdot u_1}{n_2 \cdot u_2} = \frac{n_1 \cdot a_1}{n_2 \cdot a_2}$$

Comme on cherche n_2 tel que $n_1 \cdot u_1 = n_2 \cdot u_2$, on a donc :

$$\frac{n_1 \cdot u_1}{n_2 \cdot u_2} = \frac{n_1 \cdot a_1}{n_2 \cdot a_2} = 1 \Rightarrow n_2 = n_1 \cdot \frac{a_1}{a_2}$$

où les coefficients a_i sont documentés dans le Système international d'unités par le Bureau international des poids et mesures.

Une fois connus les coefficients a_i, on détermine la quantité n_2 de l'unité u_2 par une affectation simple : `n2 = n1*a1/a2` .

Résultat On applique la méthode précédente à la conversion proposée dans l'énoncé où u_1 représente les nœuds (miles nautiques par heure), u_2 les kilomètres par heure (km/h) et u_b les mètres par seconde (m/s). Le Système international d'unités fournit par ailleurs les facteurs de conversion a_1 (nd $\rightarrow$ m/s) et a_2 (km/h $\rightarrow$ m/s) : $a_1 = 1852/3600$ et $a_2 = 1000/3600$.

Compte-tenu de ces valeurs, le code ci-contre permet de calculer le nombre n_2 de kilomètres par heure en fonction du nombre n_1 de nœuds.

```
1 a1 = 1852/3600
2 a2 = 1000/3600
3 n2 = n1*a1/a2
```

Remarque : on n'a pas cherché à effectuer « à la main » les calculs numériques : Python les fera mieux que nous ; et surtout, on n'a pas cherché non plus à particulariser la $3^{ème}$ ligne du code en `n2 = n1*1852/1000` : la forme plus abstraite `n2 = n1*a1/a2` restera identique pour convertir des parsecs en années-lumière (longueurs), des gallons en barils (volumes) ou encore des électron-volts en frigories (énergies), seules les valeurs des coefficients a_i changeront (lignes 1 et 2 du code).

Vérification Pour tester le résultat précédent, on peut comparer les valeurs obtenues par le calcul avec celles de quelques valeurs caractéristiques facilement évaluables « à la main » (exemples : $n_1 = 1$ nd $\Rightarrow$ n$_2$ = 1.852 km/h ou $n_1 = 1/1852$ nd $\Rightarrow$ n$_2$ = 1/1000 km/h).

```
>>> n1 = 1
>>> a1, a2 = 1852/3600, 1000/3600
>>> n2 = n1*a1/a2
>>> n2
1.852
```

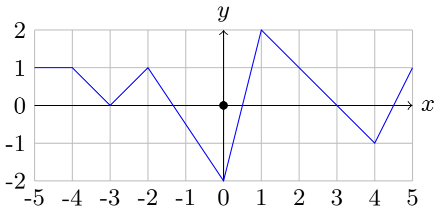
```
>>> n1 = 1/1852
>>> a1, a2 = 1852/3600, 1000/3600
>>> n2 = n1*a1/a2
>>> n2
0.0010000000000000002
```

On obtient bien par le calcul les résultats escomptés.

2.2 Graphe d'une fonction continue affine par morceaux

Objectif Mettre en œuvre l'instruction d'alternative multiple.

Enoncé On considère dans $\mathbb{R}$ la fonction continue f, affine par morceaux, définie sur $[-5;5]$ par le graphe ci-dessous et $\forall x < -5, f(x) = f(-5)$ et $\forall x > 5, f(x) = f(5)$.



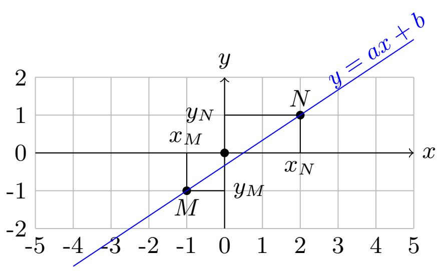
Proposer une instruction de type « alternative multiple » qui calcule la fonction $y = f(x)$ $\forall x \in \mathbb{R}$.

Méthode Il s'agit de déterminer la valeur $y = f(x)$ d'une fonction continue affine par morceaux sur $\mathbb{R}$. L'axe des réels $]-\infty, x_1, x_2, \ldots, x_n, +\infty[$ est donc vu comme une succession d'intervalles $]-\infty, x_1[$, $[x_1, x_2[$, ..., $[x_{n-1}, x_n[$ et $[x_n, +\infty[$ sur lesquels la fonction f est définie respectivement par les fonctions f_1, f_2, ..., f_n et f_{n+1} :

$$\begin{aligned} y = f(x) &= f_1(x) \quad \forall x \in\,]-\infty, x_1[\\ &= f_2(x) \quad \forall x \in [x_1, x_2[\\ &= f_3(x) \quad \forall x \in [x_2, x_3[\\ &= \quad ... \\ &= f_n(x) \quad \forall x \in [x_{n-1}, x_n[\\ &= f_{n+1}(x)\ \forall x \in [x_n, +\infty[\end{aligned}$$

Chacune des fonctions f_i correspond à une droite d'équation $y = a_i x + b_i$ où a_i représente la pente de la droite et b_i son ordonnée à l'origine. Lorsqu'on connaît 2 points $M(x_M, y_M)$ et $N(x_N, y_N)$ d'une droite d'équation $y = ax+b$, les coefficients a (pente de la droite) et b (ordonnée à l'origine) de la droite sont obtenus par résolution du système de 2 équations : $y_M = ax_M + b$ et $y_N = ax_N + b$. On obtient alors a et b :

$$a = \frac{y_N - y_M}{x_N - x_M} \text{ et } b = \frac{y_M x_N - y_N x_M}{x_N - x_M}$$



Pour la droite ci-contre :

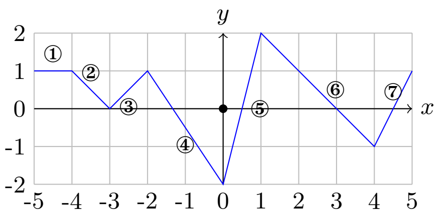
$$a = \frac{1-(-1)}{2-(-1)} = \frac{2}{3} \text{ et } b = \frac{(-1)\cdot 2 - 1\cdot(-1)}{2-(-1)} = -\frac{1}{3}$$

On vérifie graphiquement ces résultats : pour passer de M à N, on se déplace de $\Delta x = 3$ horizontalement puis de $\Delta y = 2$ verticalement (d'où la pente $a = \Delta y/\Delta x = 2/3$), et la droite coupe bien l'axe des ordonnées en $y = -1/3$.

Une fois déterminés les coefficients a_i et b_i de chaque droite, on détermine la valeur de la fonction $y = f(x)$ par une alternative multiple du genre :

```
if x < x1 : y = a1*x + b1
elif x < x2 : y = a2*x + b2
elif x < x3 : y = a3*x + b3
...
elif x < xn : y = an*x + bn
else : y = an+1*x + bn+1
```

Résultat On applique la méthode précédente à la fonction f de l'énoncé. Il faut donc déterminer les équations de droite correspondant aux différents segments du graphe de la fonction, à savoir :



① $y = 1$
② $y = -x - 3$
③ $y = x + 3$
④ $y = -3x/2 - 2$
⑤ $y = 4x - 2$
⑥ $y = -x + 3$
⑦ $y = 2x - 9$

Compte-tenu de ces équations, le code ci-contre permet de calculer $y = f(x)$, y compris pour $x < -5$ ($y = f(-5) = 1$) et $x > 5$ ($y = f(5) = 1$).
Remarque : on aurait pu simplifier les deux premières lignes de ce code en

```
if x < -4 : y = 1
```

car les instructions associées sont identiques (ie. les fonctions affines sont identiques sur $]-\infty, -5[$ et $[-5, -4[$).

```
if   x < -5 : y = 1
elif x < -4 : y = 1
elif x < -3 : y = -x - 3
elif x < -2 : y = x + 3
elif x <  0 : y = -3*x/2 - 2
elif x <  1 : y = 4*x - 2
elif x <  4 : y = -x + 3
elif x <  5 : y = 2*x - 9
else        : y = 1
```

Vérification Pour tester le résultat précédent, on peut comparer les valeurs obtenues par le calcul avec celles lues directement sur le graphe pour quelques points caractéristiques. Ces points de mesure sont choisis judicieusement : ils ne correspondent pas aux bornes des intervalles déjà prises en compte dans la méthode mais plutôt à des points où la fonction s'annule (exemples : $x = -4/3$, $1/2$, 3 ou $9/2$) ou à des points d'abscisses aux nœuds de la grille de lecture (exemples : $x = -1$ ou $x = 2$). On peut vérifier par exemple pour $x = -1$ ($y = f(-1) = -1/2$) et $x = 3$ ($y = f(3) = 0$).

```
>>> x = -1
>>> if x < -4 : y = 1
elif   x < -3 : y = -x - 3
elif   x < -2 : y = x + 3
elif   x <  0 : y = -3*x/2 - 2
elif   x <  1 : y = 4*x - 2
elif   x <  4 : y = -x + 3
elif   x <  5 : y = 2*x - 9
else          : y = 1

>>> y
-0.5
```

```
>>> x = 3
>>> if x < -4 : y = 1
elif   x < -3 : y = -x - 3
elif   x < -2 : y = x + 3
elif   x <  0 : y = -3*x/2 - 2
elif   x <  1 : y = 4*x - 2
elif   x <  4 : y = -x + 3
elif   x <  5 : y = 2*x - 9
else          : y = 1

>>> y
0
```

On obtient bien par le calcul les résultats lus sur la grille.

3 Retours d'expériences

Les exercices présentés précédemment (section 2) ont été proposés à de nombreuses générations d'étudiants de l'Enib bien avant d'utiliser la démarche Mrv. Les retours d'expériences associés ont permis de mettre en évidence au moins trois réflexions méthodologiques : ne pas se tromper d'objectif (3.1), expliciter l'implicite (3.2) et encourager la rédaction (3.3).

3.1 Ne pas se tromper d'objectif

Les étudiants de l'ENIB sont issus des voies scientifique (BAC S) et technique (BAC STI) de l'enseignement secondaire. C'est pourquoi, pour donner du sens aux exercices d'algorithmique, de nombreux exemples sont empruntés aux mathématiques, à la physique et plus généralement aux sciences de l'ingénieur. C'est ainsi le cas des deux exemples précédents : la physique pour les conversions d'unités (exercice 2.1) et les mathématiques pour les fonctions continues affines par morceaux (exercice 2.2).

Ces emprunts interdisciplinaires sont absolument nécessaires pour participer au décloisonnement des disciplines… que les étudiants ont progressivement appris à cloisonner et à isoler au cours de leur scolarité. Mais ils posent le problème de l'objectif thématique de l'exercice. En effet, les étudiants peuvent être si perturbés par la thématique secondaire (ici les mathématiques ou la physique) qu'ils en oublient la thématique principale (ici l'algorithmique), ce qui n'est évidemment pas le but recherché par l'exercice d'algorithmique.

Dans l'exemple de la fonction continue affine par morceaux, un point de blocage souvent rencontré porte sur la détermination de l'équation d'une droite. A ce niveau (premier semestre de l'enseignement supérieur), très peu d'étudiants en difficulté avec l'équation d'une droite, « oseront » écrire quelque chose comme : « supposons que l'on connaisse l'équation de la droite $y = f_i(x)$ dans l'intervalle i considéré, alors l'alternative multiple recherchée s'écrira sous la forme… ». Or, c'est pourtant fondamentalement ce qu'attend l'informaticien, quelle que soit sa « déception » face à la non-maîtrise d'une notion mathématique aussi élémentaire que l'équation d'une droite. Dans une initiation à l'algorithmique, on s'attachera à vérifier la cohérence logique des différentes conditions de l'alternative multiple plutôt que la précision des instructions associées à ces conditions. Ainsi, une réponse cohérente du point de vue des conditions de l'alternative multiple sera beaucoup plus proche de l'objectif recherché qu'une réponse incohérente sur les conditions même avec des équations de droite correctes.

On s'attachera alors à expliciter l'objectif thématique de l'exercice et à renseigner au mieux les éléments nécessaires aux thématiques secondaires. On rappellera aux étudiants que l'évaluation porte sur l'objectif thématique principal et non sur les thématiques secondaires.

3.2 Expliciter l'implicite

Si on retient que la qualité d'une réponse est son aptitude à satisfaire strictement aux besoins exprimés dans la question, alors les étudiants de l'ENIB sont devenus des professionnels de la qualité. Dans l'exemple de la conversion d'unités, il est fréquent que la réponse des étudiants tienne en une seule ligne : `n2 = 1.852*n1`, ce qui est effectivement la bonne réponse qui mérite donc, dans leur esprit, la meilleure appréciation. Mais si l'on peut se contenter de cette réponse dans un contexte de physique élémentaire, il n'en va pas de même dans un contexte de formation d'ingénieur.

En algorithmique, on cherche à caractériser différentes propriétés d'un algorithme telles que sa validité, sa réutilisabilité, sa robustesse, sa complexité ou encore son efficacité. Il faut donc progressivement développer chez l'informaticien débutant des réflexes de validation, de réutilisation, de protection, d'évaluation ou encore d'adaptation au support matériel. L'acquisition de ces réflexes méthodologiques doit être évaluée au même titre que l'acquisition des connaissances proprement dites comme l'affectation ou l'alternative multiple. Il faut donc expliciter auprès des étudiants ces objectifs méthodologiques et ne pas les cantonner au niveau d'objectifs implicites plus ou moins pris en compte dans l'évaluation d'une réponse à un exercice donné.

En premier lieu, il faut s'assurer que l'algorithme est valide : réalise-t-il exactement la tâche pour laquelle il a été conçu ? On demandera alors explicitement aux étudiants de proposer une démarche de vérification de leurs réponses. Dans l'exemple de la fonction continue affine par morceaux, on compare le résultat du calcul par algorithme à une lecture directe sur le graphe pour quelques points caractéristiques bien choisis (points où la fonction s'annule, points aux nœuds de la grille de lecture). Cette méthode ne permet évidemment pas de valider l'algorithme $\forall x \in \mathbb{R}$, mais permet d'invalider l'algorithme proposé s'il existe une incohérence entre le calcul et la lecture directe sur le graphe pour un des points caractéristiques considérés. Cette « vérification » n'en demeure pas moins satisfaisante, et essentielle, dans un cours d'initiation à l'algorithmique.

Dans un deuxième temps, on s'intéresse à sa généricité : l'algorithme est-il réutilisable pour résoudre des tâches équivalentes à celle pour laquelle il a été conçu ? Dans l'exemple de la conversion d'unités, on préfère l'affectation générique (`n2 = n1*a1/a2`) à l'affectation particulière (`n2 = n1*1852/1000`) et encore plus à l'affectation pré-calculée (`n2 = 1.852*n1`). L'affectation générique s'applique en effet à toute conversion d'unités linéairement dépendantes l'une de l'autre. L'affectation particulière répond correctement mais de manière *ad hoc* au problème posé, ce qui ne permet pas sa réutilisabilité à d'autres types d'unités, y compris même à d'autres unités de vitesse telle que la conversion de miles terrestres par heure en kilomètres par heure ($1 \text{ mi} = 1.609344 \cdot 10^3 \text{ m}$). Quant à l'affectation pré-calculée, outre le fait qu'on n'est jamais à l'abri d'une erreur de calcul, elle ne permettra pas de « remonter » aussi facilement que les deux autres à la signification du coefficient numérique calculé et donc, ne facilitera pas la maintenance du code proposé.

Dans un troisième temps, on s'attachera à le rendre robuste : l'algorithme est-il protégé de conditions anormales d'utilisation ? Dans l'exemple de la conversion d'unités, l'affectation générique `n2 = n1*a1/a2` ne s'applique qu'à des unités linéairement dépendantes l'une de l'autre ($u_2 = \alpha u_1$). Il existe cependant des grandeurs qui sont en relation affine l'une de l'autre ($u_2 = \alpha u_1 + \beta$). C'est le cas des températures Farenheit t_F et Celsius t_C qui sont reliées à la température thermodynamique Kelvin T_K (unité de base des températures) par les relations $t_C = T_K - 273.15$ et $t_F = 9T_K/5 - 459.67$, soit $t_C = 5/9 \cdot (t_F + 459.67) - 273.15$ entre elles. En algorithmique, l'étude des fonctions et de leurs préconditions (conditions d'application) permettra d'aborder systématiquement cette propriété de robustesse.

En ce qui concerne les propriétés d'un algorithme telles que la complexité (combien d'instructions élémentaires seront exécutées pour réaliser la tâche pour laquelle l'algorithme a été conçu ?) et l'efficacité (l'algorithme utilise-t-il de manière optimale les ressources du matériel qui l'exécute ?), elles seront abordées au travers d'exemples précis, leur étude systématique ne relevant pas du cours d'initiation à l'algorithmique de l'Enib.

Dans tous les cas, on s'attachera à préciser le ou les objectifs méthodologiques de l'exercice qui seront alors évalués explicitement au même titre que l'objectif thématique.

3.3 Encourager la rédaction

La rédaction « en bon français » de la réponse à un exercice ne constitue pas le point fort des jeunes étudiants de l'Enib. Les réponses sont (trop) souvent libellées « simplement » sous forme de valeurs, de formules, de diagrammes ou de codes informatiques : aucune explication « en bon français » ne vient compléter ni expliciter la réponse, ni la démarche qui a conduit à cette réponse, encore moins la critique de la solution proposée. Et pourtant, si la réponse à la question est attendue, les éléments de discours qui l'accompagnent le sont tout autant, d'autant plus dans une formation d'ingénieurs qui vise à former des professionnels qui devront rédiger des cahiers des charges, des spécifications et des conceptions détaillées, des recettes de tests, des notes de synthèse, écrites comme orales, ou encore des réponses à des appels d'offres. Le métier d'ingénieur ne peut se contenter d'une valeur, d'une formule, d'un diagramme ou d'un code : s'il faut être capable de trouver une solution à un problème donné, il faut aussi savoir « défendre » rationnellement la solution proposée. L'argumentaire qui accompagne la solution proposée doit permettre de mieux comprendre cette solution et augmenter ainsi la confiance du « lecteur » dans les compétences du « rédacteur » à résoudre le problème posé.

On s'attachera alors à prendre en compte explicitement la rédaction dans l'évaluation de la réponse. Il faudra sans doute pour cela, soit augmenter le temps accordé à l'exercice, soit diminuer le nombre d'exercices à résoudre dans un temps imparti, pour permettre à l'étudiant « rédacteur » de soigner cet aspect important de sa réponse.

4 Généralisation

A l'aune des exemples précédents (section 2) et des retours d'expériences associés (section 3), cette section reconsidère les trois étapes de la démarche Mrv : explicitation de la méthode (4.1), application de la méthode (4.2) et vérification du résultat (4.3).

4.1 Explicitation de la méthode

Etant donné un énoncé qui propose à l'étudiant de résoudre un exercice portant sur un cas particulier donné, la première étape de la démarche Mrv consiste à décrire une méthode générique qui, lorsqu'on l'appliquera, permettra de résoudre le cas particulier considéré ainsi que tout problème équivalent à celui qui est posé.

Pour un débutant, cette étape d'explicitation d'une méthode générique est une étape difficile. Elle nécessite de développer des capacités d'abstraction qui mettent en œuvre des mécanismes d'induction pour favoriser le passage de données particulières à des propositions plus générales. L'induction est en effet un type de raisonnement qui permet de « remonter » de cas particuliers à la loi qui les régit, des effets à la cause ou encore des conséquences au principe.

C'est également une étape difficile parce que la description d'une méthode peut difficilement se résumer à une valeur, une formule, un diagramme ou un code informatique. Elle nécessite une phase rédactionnelle rigoureuse et suffisamment détaillée pour qu'un lecteur averti puisse appliquer sans hésiter la méthode décrite.

Cette étape permet ainsi de développer des capacités d'abstraction et des capacités rédactionnelles absolument nécessaires aux futurs ingénieurs.

4.2 Application de la méthode

Etant donné une méthode générique pour résoudre un ensemble de problèmes équivalents, le deuxième étape de la démarche Mrv consiste à appliquer cette méthode à un problème particulier.

Pour un débutant, cette étape d'application d'une méthode générique est une étape assez facile. Elle nécessite de développer des capacités d'exécution qui mettent en œuvre des mécanismes de déduction pour réaliser le passage de propositions générales à un cas particulier. A l'inverse de l'induction, la déduction est en effet un type de raisonnement qui « va » du général au particulier, de la cause aux effets ou encore du principe aux conséquences.

Cette étape permet ainsi de développer des capacité d'exécution en respectant rigoureusement des consignes imposées. Elle met ainsi en évidence des capacités « techniciennes » qui seront très utiles aux futurs ingénieurs dans leur mission d'encadrement d'équipes d'ouvriers et de techniciens.

4.3 Vérification du résultat

Etant donné un résultat obtenu par application d'une méthode générique pour résoudre un problème particulier, la troisième étape de la démarche MRV consiste à vérifier ce résultat par des méthodes alternatives ou complémentaires de celle déjà utilisée.

Pour un débutant, cette étape de vérification du résultat obtenu est assez difficile car il s'agit avant tout de remettre en cause son propre travail. Dans le meilleur des cas, le débutant estime que la vérification consiste simplement à refaire les mêmes calculs, le même raisonnement ou la même démarche : c'est effectivement la moindre des choses que de ré-appliquer la méthode pour vérifier qu'on ne s'est pas trompé en l'appliquant la première fois. Mais la vérification consiste plutôt à changer de point de vue sur le problème et à utiliser d'autres méthodes pour « estimer » la validité du résultat.

Il peut effectivement exister plusieurs méthodes alternatives pour résoudre un même problème. Résoudre alors le problème par deux méthodes différentes et obtenir le même résultat renforce bien entendu la « confiance » en ce résultat. Mais dans bien des cas, il s'agit plutôt de méthodes complémentaires, le plus souvent sous forme d'heuristiques, qui permettent de détecter que le résultat est certainement faux, comme par exemple la preuve par 9 en calcul élémentaire ou l'analyse dimensionnelle en physique. Et si de telles méthodes complémentaires ne détectent pas que le résultat est faux, alors ça renforce ici encore la « confiance » que l'on peut avoir dans le résultat obtenu.

Cette étape permet ainsi de développer l'esprit critique des futurs ingénieurs en insistant sur le souci de vérification systématique de ses propres résultats comme de ceux provenant d'autres sources (collègues, articles, internet…).

5 Mise en œuvre

La démarche MRV est mise en œuvre dans le cours d'« Initiation à l'algorithmique » du semestre S1 à l'ENIB (5.1). Lors des contrôles, elle conduit à une triple évaluation des exercices selon une notation adaptée (5.2) dans le cadre d'un contrôle continu systématique (5.3).

5.1 Contexte

Les « Questionnements de cours » qui accompagnent le cours d'« Initiation à l'algorithmique » du semestre S1 à l'ENIB sont conçus de façon plutôt « ascendante » (de l'exemple au concept) et se veulent complémentaires des notes de cours qui, elles, sont conçues plutôt classiquement de façon « descendante » (du concept à l'exemple).

Chaque questionnement concerne un point particulier du cours ; il est structuré en 5 parties de la manière suivante :

1. Exemple : dans cette partie, des questions « simples » sont posées sur un problème « connu » de la « vie courante » afin d'introduire le concept informatique sous-jacent. On y trouvera des exemples tels qu'aller au restaurant, ranger un meuble à tiroirs, analyser les sorties d'un circuit logique, compter avec BOBBY LAPOINTE en base « bibi », déterminer sa mention au bac, planter un clou, ranger des rondins de bois, cuisiner un quatre-quarts aux pépites de chocolat, jouer aux tours de Hanoï ou encore trier un jeu de cartes.

   Cette partie est principalement traitée de manière informelle par les étudiants eux-mêmes, individuellement ou en groupe.

2. Généralisation : dans cette partie, les concepts informatiques sous-jacents sont présentés et introduits à l'aide de questions plus « informatiques ». On y aborde les concepts d'algorithmique, d'affectation, de calculs booléens, de codage des nombres, de tests et d'alternatives, de boucles, de spécification de fonction , de récursivité ou encore de manipulation de séquences.

   En général, cette partie est traitée par l'enseignant.
3. Applications : des exemples « simples » d'application sont ensuite proposés.

   Le premier exemple est en général traité *in extenso* par l'enseignant en suivant la démarche MRV, les autres par les étudiants, en groupe ou individuellement.
4. Entraînement : cette partie est une préparation à l'évaluation qui a lieu en début de séance suivante. Les étudiants y travaillent chez eux entre les deux séances, individuellement ou en groupe.

   (a) Enoncé : on présente ici le problème que l'on souhaite traiter tel que le calcul en base « Shadok », le calcul de facteurs de conversion entre unités physiques, l'établissement de la table de vérité d'une expression logique, l'écriture d'un nombre réel selon la norme IEEE 754, la détermination de la valeur d'une fonction continue et linéaire par morceaux, le calcul d'un développement limité selon une certaine précision, le dessin d'un motif géométrique composé de polygones réguliers, la spécification d'une fonction connue (un « grand classique » de la programmation), le parcours d'un arbre binaire ou encore le tri d'un annuaire selon différents critères.

   (b) Exemple : un exemple est traité en détail dans cette partie en suivant la démarche MRV. Ce sont de tels exemples qui ont été présentés dans la section 2 du présent document.

   (c) Questions : 24 questions de même difficulté sont proposées ici pour permettre à chaque étudiant de s'entraîner sur le problème à traiter. La résolution de 2 ou 3 de ces exemples l'aide ainsi à induire (à faire émerger) la méthode générique qui est attendue ainsi qu'à mener explicitement les vérifications souhaitées.

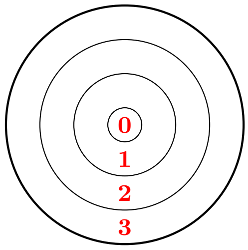
   Le jour de l'évaluation, chaque étudiant traite individuellement une des 24 questions tirée au sort le jour du contrôle (une question différente par élève). Lors de cette évaluation, tous documents, calculettes, téléphones et ordinateurs sont interdits. A la fin de chaque contrôle, il est demandé à chaque étudiant de s'auto-évaluer pour chacune des étapes de la méthode MRV selon une grille de notation à 4 niveaux (voir section 5.2 suivante). Enfin, une correction est proposée par l'enseignant juste après le contrôle, « à chaud ».
5. Révisions : Cette partie fait le lien entre les questionnements de cours et les notes de cours.

5.2 Triple évaluation

Chaque contrôle donne lieu a une triple évaluation de la part de l'enseignant : une évaluation concerne la qualité de l'explicitation de la méthode générique (M), une autre la qualité du résultat obtenu (R) et la troisième la pertinence de la vérification du résultat (V). Ainsi, le résultat à la question posée, dont se contentent le plus souvent les étudiants, n'est plus le seul élément de réponse attendu : il est également demandé aux étudiants d'expliciter la méthode utilisée ainsi que les vérifications menées.

La grille de notation adoptée doit permettre de « soulager » l'enseignant dans sa tâche de correction et d'aider les étudiants à mener leurs propres évaluations. Un exercice cherchant à évaluer un objectif particulier, la notation exprime alors « simplement » la distance qui reste à parcourir pour atteindre cet objectif. Quatre « distances » sont ainsi pré-définies selon une métaphore de la cible :

0 : « en plein dans le mille ! » → l'objectif est atteint
1 : « pas mal ! » → on est proche de l'objectif
2 : « juste au bord de la cible ! » → on est encore loin de l'objectif
3 : « la cible n'est pas touchée ! » → l'objectif n'est pas atteint



Ayant choisi de ne garder qu'un petit nombre de niveaux pour « faciliter » l'évaluation, le choix de 4 niveaux a finalement été préféré à 2, 3 ou 5 niveaux :
- une notation sur 2 niveaux (*tout ou rien*) est un peu trop caricaturale ;
- avec un (petit) nombre impair de niveaux (3 ou 5), l'expérience montre que, dans le doute, le correcteur a tendance à choisir plus facilement le niveau du milieu (1 ou 3) alors qu'avec un nombre pair de niveau (ici 4), il doit « choisir son camp » : objectif plutôt atteint (0 ou 1) ou plutôt raté (2 ou 3).

En fait, il existe un cinquième niveau qui correspond à une absence au contrôle, sanctionnée par la note 4 (l'objectif n'a pas été visé), dite note minimale.

Ainsi, et pour changer de point de vue sur la notation, le contrôle est réussi lorsqu'on a 0 ! Il n'y a pas non plus de 1/2 point ou de 1/4 de point : le seul barème possible ne comporte que 4 niveaux : 0, 1, 2 et 3. On ne cherche donc pas à « grappiller » des points :
- on peut avoir 0 (objectif atteint) et avoir fait une ou deux erreurs bénignes en regard de l'objectif recherché ;
- on peut avoir 3 (objectif non atteint) et avoir quelques éléments de réponse corrects mais sans grand rapport avec l'objectif.

Pour obtenir une note plus « classique » (ie. une note sur 20 : $n_{/20}$), il suffit de prendre le complément à 4 de la note sur 0 ($n_{/0}$) et de le multiplier par 5 :

$n_{/20} = (4 - n_{/0}) \times 5$ soient les équivalences :

$n_{/0}$	$n_{/20}$	signification
0	20	l'objectif est atteint
1	15	on est proche de l'objectif
2	10	on est encore loin de l'objectif
3	5	l'objectif n'est pas atteint
4	0	l'objectif n'a pas été visé

Dans ce contexte, avoir 20/20 ne signifie pas qu'on est génial ou que c'est parfait, cela signifie « juste » qu'on a atteint un objectif fixé, et c'est déjà beaucoup !

5.3 Contrôle continu

A l'Enib, le cours d'informatique du semestre S1 est un enseignement de 42h réparties régulièrement sur 14 semaines : 1h30 de cours-td en salle banalisée toutes les semaines (par groupes de 36 étudiants maximum) et 3h de laboratoire en salle informatique toutes les deux semaines (par groupes de 24 étudiants maximum). Chaque séance donne lieu a priori à une évaluation :
- 1 Qcm de questions de cours une séance de cours-td sur deux, soient 7 Qcm de 5' chacun, en fin de séance ;

– 1 contrôle MRV une séance de cours-td sur deux, soient 7 MRV de 30' chacun, en début de séance ;
– 1 contrôle sur machine à chaque séance de laboratoire, soient 7 LABO de 30' chacun, en début de séance.

Chaque exercice est évalué selon la grille de notation à 4 niveaux décrite à la section 5.2 précédente.

L'accumulation et la fréquence des contrôles, notés d'une séance à l'autre, permettent un suivi plus régulier et plus fin des apprentissages des étudiants. Ceux-ci travaillent plus et plus régulièrement en développant au fur et à mesure leurs capacités d'abstraction et leur esprit critique. Et de leur avis même, ils ont l'impression au bout du compte de mieux maîtriser leurs apprentissages.

6 Conclusion

La démarche MRV (Méthode–Résultat–Vérification) mise en place dans le cadre du cours d'« Initiation à l'algorithmique » du semestre S1 à l'ENIB, cherche à développer, à travers l'acquisition de compétences thématiques en informatique, des compétences plus transversales, nécessaires aux futurs ingénieurs : la capacité d'abstraction, la rigueur applicative et l'esprit critique.

C'est pourquoi la démarche MRV repose sur trois étapes bien distinctes :

1. l'explicitation d'une méthode générique de résolution d'une famille de problèmes équivalents pour développer la capacité d'abstraction de l'étudiant,
2. l'application de la méthode proposée à un cas particulier pour développer sa rigueur applicative et
3. la vérification du résultat ainsi obtenu pour développer son esprit critique.

Sa mise en œuvre à travers un contrôle continu systématique, quoique récente, permet d'entrevoir quelques évolutions encourageantes. Les étudiants ne se contentent plus d'un simple résultat répondant strictement à la question posée mais s'engagent avec plus d'intérêt dans la généralisation de leur méthode et dans la remise en cause de leur propre résultat. Ils travaillent plus et plus régulièrement et enfin, ils ont l'impression de mieux maîtriser leurs apprentissages.

Il reste que cette démarche est lourde à mettre en place pour un enseignant isolé et seules des équipes pédagogiques constituées pourront s'engager sereinement dans cette voie exigeante.

Références

[1] Tisseau J., Initiation à l'algorithmique, *Notes de cours*, ENIB 2009

[2] Tisseau J., Initiation à l'algorithmique, *Travaux dirigés*, ENIB 2009

[3] Tisseau J., Initiation à l'algorithmique, *Questionnements de cours*, ENIB 2012