Pour une pédagogie de la réussite dans un premier cours de programmation

Pierre Bard
CEGEP de Rimouski

Introduction

Dans cet exposé, je tenterai de vous livrer le plus simplement possible les réflexions qu'une longue pratique de l'enseignement de l'informatique suscite chez moi. Ce sujet, il va sans dire, me tient particulièrement à coeur, ayant investi des efforts constants, depuis mes débuts comme prof d'informatique en 1969, dans la recherche de moyens pour faciliter l'apprentissage de la programmation chez nos élèves du collégial.

Où en sommes-nous en 1994 ?

Si je devais identifier le problème le plus crucial auquel nous devons faire face comme profs d'informatique en 1994, je dirais sans trop hésiter que c'est le **phénomène du décrochage** : une proportion importante des élèves qui commencent un DEC en informatique ne termine même pas une première année. Les causes sont multiples.

Pour plusieurs, l'aspect ludique de l'informatique, le seul qu'ils aient connu jusque là, se trouve anéanti dès les premières semaines de cours. C'est la grande désillusion : à la liberté du jeu on substitue les contraintes qu'exige l'acquisition d'une discipline intellectuelle.

Plusieurs arrivent également au cégep sans posséder l'outillage intellectuel et affectif nécessaire pour réussir des cours le moindrement exigeants. Bien sûr, nous n'aurions pas conscience de ce problème si nous avions des concours d'entrée aussi sévères que ceux des conservatoires de musique ! Nous ne travaillerions qu'avec la crème. Mais on sait très bien que cela va à l'encontre des objectifs ministériels qui sont d'amener une proportion plus grande de nos jeunes à acquérir un DEC. Et, il ne faut pas se le cacher, nous sommes complices, nous les profs, de l'acceptation de tous ces élèves qui n'ont que peu de chances de réussir : nos plus jeunes profs, notre sang neuf pour ainsi dire, perdraient du coup leur emploi. Plus noblement, d'une façon moins intéressée, nous considérons notre travail comme une entreprise de sauvetage : nous voulons donner une chance à tous et à toutes et, par nos efforts, en remettre le plus grand nombre sur la voie de la réussite.

Je ne peux passer sous silence l'un des plus graves causes de décrochage qui affecte cette fois-ci même de bons élèves. C'est que **la société n'envoie pas à nos jeunes un message clair à l'effet qu'elle veut d'eux**. Je me souviens du temps où nos élèves recevaient avant la fin de leur DEC la visite d'employeurs prêts à leur offrir des emplois à temps plein. Nous en

sommes plutôt aujourd'hui, en informatique tout au moins, à l'ère des « jobines ». Dans ces conditions, comment nos jeunes peuvent-ils poursuivre leurs études avec enthousiasme ? Il flotte un climat d'incertitude, face à l'avenir, qui n'a rien de bon.

Ceci étant dit, que pouvons-nous faire ? Miser encore plus fortement qu'avant sur l'acquisition des compétences qui permettront à nos jeunes de tirer leur épingle du jeu.

Et la place de la programmation ?

Réglons cette question tout de suite. Dans un monde où de plus en plus en plus de personnes utilisent, dans le cadre de leur travail et sans être diplômées en informatique, des logiciels conviviaux et performants, on peut se demander si l'apprentissage de la programmation n'est pas devenu désuet, d'autant plus que, même dans le domaine de la programmation comme telle, nous avons des logiciels d'aide, comme des générateurs de programmes, qui font automatiquement une partie du boulot. Mon point de vue sur le sujet est le suivant : l'apprentissage de la programmation est l'un des premiers pas que l'apprenti-informaticien doit franchir pour réussir plus tard des opérations plus complexes comme l'analyse de système. Il y a là un continuum qui ne peut être brisé. La programmation est à l'informaticien ce que la maîtrise d'un instrument est au compositeur de musique. Peut-on imaginer un instant que Mozart aurait pu composer ses sonates pour piano s'il n'avait pas maîtrisé un clavier ? Dans la mesure où l'informaticien est plus qu'un utilisateur de logiciel, qu'il est aussi un concepteur, il doit posséder des bases logiques solides. Et dans ce sens, l'apprentissage d'une démarche de résolution de problèmes fondée sur la capacité d'aller du général au particulier est tout à fait capital.

Facteurs aidants chez le prof

Je vais maintenant énumérer et commenter des actions et des attitudes du prof qui peuvent concourir à la réussite de l'élève. Qu'il soit bien entendu qu'en rabaissant nos exigences on peut amener n'importe qui à passer, mais que ce n'est certainement pas ce qui doit être fait ! Un premier cours de programmation ne doit pas être une blague : l'élève doit le terminer avec l'impression d'avoir vraiment pu évaluer ses propres capacités. Comme une initiation dans son sens le plus direct, un premier cours de programmation doit être une mise à l'épreuve honnête, « réussissable » et déterminante.

Accueillir positivement l'élève malgré ses faiblesses de départ. Jacques Drillon, dans son *Traité de la ponctuation française*, dit quelque part : « Il faut être impitoyable pour la faute, mais plein de compassion pour le coupable. » Voilà joliment résumée une attitude qui reconnaît à l'élève le droit à l'erreur et au prof le devoir de la corriger.

Aller du facile au difficile. Notre pédagogie doit être basée sur une démarche progressive de l'élève. Inutile de l'assommer dès le départ avec des problèmes hors de sa portée ! Il est intéressant à cet égard de commencer l'apprentissage de la programmation par une identification des mécanismes communs au monde des ordinateurs et à celui de notre vie de

tous les jours, comme, par exemple, la séquence, la répétition et la sélection d'opérations à exécuter.

S'assurer que les notions de base sont maîtrisées. J'aime comparer la programmation au jeu de blocs LEGO. L'enfant ne réussira un assemblage que s'il a joué avec chacune des pièces pour en connaître les propriétés. Nos élèves ne peuvent espérer *imaginer et élaborer* un programme opérationnel s'ils ne jouent pas d'abord avec les mécanismes de base que sont en programmation la séquence, la répétitive et l'alternative. Ils doivent manipuler de petits programmes déjà faits, les modifier, les faire tourner avec un ordinateur mais aussi les exécuter à la main sur papier. C'est aussi l'occasion rêvée, avec de petits programmes très simples, de se familiariser avec un environnement de programmation.

Montrer à aller du général au particulier. Le prof doit croire mordicus à la valeur d'une démarche structurée qui va du général au particulier, car les élèves, eux, n'y croient pas. En tous cas, pas au départ. Au mieux, ils sont prêts à nous faire confiance. Mais le plus souvent, ils nous prennent pour des empêcheurs de tourner en rond : ils se sentent brimés dans leur liberté de faire à leur guise. Cette attitude tient à un certains nombres de facteurs. D'abord, c'est nouveau : jamais on ne leur a fait utiliser une telle approche, pas de façon systématique tout au moins. Et puis cette démarche suppose une habileté intellectuelle peu maîtrisée à leur âge : *savoir décrire dans des termes généraux ce qui n'a pas encore été développé dans le détail*. Ils voudraient plutôt rester dans le particulier et n'avoir jamais à penser en termes généraux. Heureusement, après quelques mois ou après un an, ils se rendent compte que plus les projets grossissent, moins leur conception des choses est viable.

Être clair. Il n'est pas suffisant pour un prof de maîtriser sa matière, ce qui est relativement simple à ce niveau. Encore faut-il être capable de donner des explications claires et surtout de connaître les points où ça accroche. La qualité de la communication joue beaucoup ici. Le prof ne doit pas hésiter à créer des occasions de parler personnellement à chaque élève, ce que l'évaluation formative, abordée au point suivant, encourage fortement.

Croire à la valeur des activités formatives. Le prof doit consacrer du temps et de l'énergie pour suivre de très près la réalisation d'un bon nombre de petits travaux (voir la liste des activités formatives ci-jointe). Corriger sans avoir à mettre de points n'est pas si désagréable après tout ! Personnellement, ça me réconcilie un peu avec la correction. Et les élèves aussi aiment qu'on leur reconnaisse enfin le droit à l'erreur, sans pénalité. En tous cas, la remise individuelle des travaux corrigés est une bonne occasion d'échanger quelques mots avec des élèves par ailleurs souvent timides.

Savoir se faire aimer et respecter. D'une manière générale, les élèves aiment que le prof fassent régner en classe un climat à la fois serein et respectueux de tous. Il doit donc faire preuve de tact pour garder le contrôle. Quant au fait de se faire aimer, ce n'est souvent que la contrepartie du fait que le prof aime aussi ses élèves et veut aider chacun à réussir.

Être vigileant. S'il est une matière où le copiage est facile, c'est bien l'informatique. Copier un programme sur disquette, changer des noms de variables, c'est si simple ! Ce n'est pas favoriser le développement de l'élève que de fermer les yeux sur une pratique qui est

devenue une véritable plaie non seulement dans nos collèges mais tout autant, sinon plus, dans nos universités. À cet égard, notre rôle n'est pas surtout de jouer les cerbères. Il est de créer chez l'élève, dans le cadre de l'élaboration d'une solution à un problème, un contexte d'appropriation des premières étapes de la solution, celles qui concernent la logique. Quand un élève a compris le problème à résoudre et qu'il a réussi, souvent avec quelques conseils de notre part, à échafauder une solution qui se tient, il est beaucoup moins tenté de copier. À mon point de vue, qui est en même temps celui de mes collègues, le mieux à faire lorsqu'on soumet un travail pratique (qui compte dans l'évaluation sommative), c'est de consacrer 2 ou 3 heures en classe à la *compréhension de la donnée* (les élèves ont à lire attentivement le texte et à fournir par écrit des questions) et à l'*élaboration par l'élève*, à l'aide d'outils appropriés (diagramme hiérarchique, pseudo-code, etc.), *d'une solution* qu'il s'approprie vraiment. Il s'agit pour le prof de superviser et de guider chacune des étapes de cette élaboration : les élèves à tour de rôle viennent soumettre ce qu'ils ont de fait. À la fin, le prof appose sa signature sur le brouillon qui devra être présenté comme un des éléments du rapport final. On peut contester le caractère « contrôleur » de cette pratique qui a par ailleurs un avantage considérable : elle permet au prof de vraiment comprendre le mode de fonctionnement de chaque élève et les difficultés qu'il ou elle rencontre.

Faire de l'ordinateur son assistant, non son remplaçant. L'ordinateur est un outil merveilleux losque vient le temps de rendre tangible un concept abstrait. En voyant un programme fonctionner, l'« imaginé » devient réalité. Ce qui est visualisé à ce moment-là crée un renforcement du concept qui est dès lors utilisable dans l'élaboration d'une solution à de nouveaux problèmes. Dans notre enseignement, nous devons par contre éviter de mettre l'élève en contact avec l'ordinateur alors que, dans l'élaboration d'une solution, il est encore dans le brouillard. L'ordinateur n'a aucun talent pour guider l'élève et peut même encourager un certain « mouvement brownien » dans l'esprit de celui-ci, une agitation qui ne mène nule part.

Voir un langage de programmation comme un outil, non comme une fin en soi. Il peut être tentant pour des profs de se lancer dans de grands débats sur la supériorité d'un langage par rapport à un autre. En fait, ce qu'il importe de développer chez nos élèves, c'est un *outillage logique pour résoudre avec assurance des problèmes*. On peut à coup sûr affirmer qu'une fois l'algorithme développé, la transposition dans un langage de programmation est un problème relativement secondaire que nos élèves surmonteront de plus en plus facilement. Il est bien certain cependant que si nous favorisons une conception structurée, nous avons avantage à utiliser également, au moment de la programmation, un langage structuré comme la Pascal ou le C. Dans l'enseignement d'un premier cours de programmation, nous devrions aussi nous en tenir au minimum d'éléments du langage nécessaires pour mener à bien les exercices de programmation. Il est parfaitement inutile de vouloir faire le tour des instructions.

Éviter le dogmatisme. Dans une programmation parfaitement structurée par exemple, chaque module ne possède qu'un seul point de sortie. Certaines personnes sont prêtes à jeter l'anathème sur ceux qui transgressent parfois cette règle. Il s'agit souvent de personnes qui enseignent la programmation sans jamais en faire. Pour un professeur de programmation, avoir à développer soi-même de temps en temps des systèmes de taille respectable remet les

deux pieds sur terre et fait comprendre qu'un langage de programmation n'est qu'un outil, rien d'autre. Cela ne veut pas dire que nous ne devons pas exiger de nos élèves qu'ils respectent des normes. Au contraire. Mais celles-ci doivent toujours se justifier par le fait qu'elles permettent au produit fini de maintenir une clarté logique et une limpidité qui le rendent facilement compréhensible pour d'autres.

Facteurs aidants chez l'élève

L'énumération qui précède n'est pas exhaustive. On y retrouve des actions et des attitudes qui devraient aider le prof à améliorer son enseignement. L'élève doit aussi faire sa part. Après tout, c'est lui qui est au coeur de l'activité d'apprentissage. À cet égard, on peut à juste titre se désoler du fait que, pour plusieurs de nos élèves, la « soif de vivre » ne passe pas nécessairement par la « soif d'apprendre ». Enfin, une partie de notre tâche est d'essayer de les stimuler et dans ce sens nous avons un travail de « séduction » à faire. Pour ce qui est de l'élève, voici également quelques attitudes et comportements qui peut contribuer à son succès.

Être conscient de l'adéquation entre réussite et travail. C'est probablement un euphémisme de dire que plusieurs élèves n'ont pas un sens aigu de l'effort. Ils arrivent au cégep en affirmant n'avoir jamais eu, pour réussir, à déployer des énergies importantes comme par exemple d'avoir à effectuer de nombreux travaux scolaires à domicile. En ce sens, la réussite au cégep peut être sérieusement compromise chez l'élève qui espère continuer au même rythme. Pour le prof, l'évaluation formative est un excellent outil pour déceler de tels cas et pour discuter avec les élèves concernés des correctifs à apporter.

Développer le goût du travail bien fait. L'ordinateur est un outil précis. Il faut aussi être soi-même précis lorsqu'on l'utilise. Esprits trop brouillons, prière de s'abstenir, ou de se réformer ! Il est étonnant de constater que plusieurs de nos élèves n'ont au départ aucune méthode de travail et sont même inaptes à effectuer des activités de base comme de classer correctement de l'information. Pour ce qui est d'être minutieux, il faudra repasser ! On peut ici ouvrir une parenthèse pour se demander si, sur cette question du travail bien fait, certains garçons ne sont pas, de par leur façon différente de voir les choses (certains parlent d'une « culture masculine contraire à l'acquisition des valeurs nécessaires pour réussir à l'école », Le Devoir 94-03-05), désavantagés par rapport aux filles : ils se plient difficilement au respect de normes et se complaisent dans une sorte de «délinquance » intellectuelle. Bien que peu nombreux, ils peuvent avoir une influence négative sur le reste de la classe.

Se faire un devoir de ne rater aucun cours. Dans une matière comme la programmation, l'absence au cours est presque toujours désastreuse. La raison est simple : la matière forme une chaîne continue dont on ne peut enlever aucun maillon. L'ivresse de la liberté qu'éprouvent plusieurs de nos élèves en entrant au cégep, particulièrement ceux qui viennent de l'extérieur après avoir quitté pour la première fois le milieu familial, cette ivresse en perd plus d'un. Le prof doit envoyer des signaux clairs à ce sujet et ne pas hésiter à prendre les présences à chaque rencontre. L'élève doit être présent et exploiter au mieux ses heures de

cours, mais il doit aussi être assidu dans son travail et suivre le rythme du groupe.

Avoir le sens de l'autonomie. L'apprentissage de la programmation est une entreprise profondément intérieure. Elle vise au développement d'une pensée logique féconde, d'une assurance en soi qui est tout à fait incompatible avec le travail d'équipe. Un travail de programmation ne se fait pas à deux. Même sur le marché du travail, dans le développement en équipe de vastes systèmes, deux personnes ne font pas la même chose. Pour quelqu'un qui n'a pas confiance en soi, ce peut être tentant de toujours se fier au voisin. Être autonome, ça veut dire aussi être capable de prendre des décisions non suggérées par le prof, comme par exemple de réviser des blocs complets de matière déjà vus mais mal assimilés.

Savoir communiquer avec les autres élèves. L'autonomie dont il vient d'être fait mention n'est pas incompatible avec le fait d'avoir recours à un autre élève pour obtenir des explications. L'élève plus doué devra cependant avoir la sagesse de donner des indices, et non la solution toute faite. Dans ce sens il devra mettre en application le dicton populaire à l'effet qu'il vaut mieux, quand quelqu'un a faim, lui montrer à pêcher que de lui donner un poisson.

Savoir communiquer avec le prof. Selon l'image populaire, le programmeur est un petit génie qui vit seul dans son coin, presque misanthrope. Rien n'est plus faux ! Pour réussir en informatique, il faut cultiver la communication. On oublie trop souvent de dire qu'en programmation, on ne travaille pas pour soi, mais plutôt pour les autres, pour des clients en chair et en os. Il faut discuter avec eux pour connaître leurs besoins, pour développer ou adapter des systèmes en fonction de leurs exigences. Dans une situation d'apprentis-sage, le prof joue souvent ce rôle de client, en plus de jouer celui de guide.

Conclusion

Dans les propos qui précèdent, j'ai voulu énoncer quelques principes élémentaires mais qui sont à la base de notre mode de fonctionnement, dans nos cours de programmation au cégep de Rimouski. Ces principes font l'objet d'un consensus dans notre département. En les appliquant, nous voulons aider nos élèves à faire des progrès, tout en maintenant un cadre exigeant.