Didactique de l'informatique

M1 MEEF NSI
Planification

Benjamin Wack

Institut national supérieur du professorat et de l'éducation
Académie de Grenoble

2024

Plan

La « recette » de Kaprekar (ici proposée en Terminale S)

Essayez de suivre la procédure ci-dessous avec plusieurs nombres de départ distincts :

1. Choisir un nombre à 3 chiffres
2. Écrire le plus grand nombre possible et le plus petit nombre possible avec ces 3 chiffres
3. Calculer la différence
4. Recommencer en 2. avec le résultat obtenu

La « recette » de Kaprekar (ici proposée en Terminale S)

Essayez de suivre la procédure ci-dessous avec plusieurs nombres de départ distincts :

1. Choisir un nombre à 3 chiffres
2. Écrire le plus grand nombre possible et le plus petit nombre possible avec ces 3 chiffres
3. Calculer la différence
4. Recommencer en 2. avec le résultat obtenu

Quelques questions

- Que constate-t-on ?

La « recette » de Kaprekar (ici proposée en Terminale S)

Essayez de suivre la procédure ci-dessous avec plusieurs nombres de départ distincts :

1. Choisir un nombre à 3 chiffres
2. Écrire le plus grand nombre possible et le plus petit nombre possible avec ces 3 chiffres
3. Calculer la différence
4. Recommencer en 2. avec le résultat obtenu

Quelques questions

- Que constate-t-on ?
- Pouvez-vous vérifier votre conjecture ?

La « recette » de Kaprekar (ici proposée en Terminale S)

Essayez de suivre la procédure ci-dessous avec plusieurs nombres de départ distincts :

1. Choisir un nombre à 3 chiffres
2. Écrire le plus grand nombre possible et le plus petit nombre possible avec ces 3 chiffres
3. Calculer la différence
4. Recommencer en 2. avec le résultat obtenu

Quelques questions

- Que constate-t-on ?
- Pouvez-vous vérifier votre conjecture ?
  On privilégiera ici une vérification assistée par ordinateur (pas de preuve algébrique)

Mise en place d'une procédure de vérification exhaustive

Pour chacune des phases suivantes :

- Faire le travail demandé
- Prévoir les productions et réactions d'élèves
- Expliquer le rôle didactique de cette phase

Phase 1 : De quelles variables aurons-nous besoin ?

Phase 1 : De quelles variables aurons-nous besoin ?

E1 : On va prendre A en nombre initial, on va prendre B arrangé en ordre décroissant, C en ordre croissant, on va faire la différence de C moins B. On va le remettre dans A, et on va refaire avec B et C.

E2 : Ben si A c'est une variable où c'est un nombre, pour qu'il puisse le ranger dans l'ordre croissant et décroissant, faut d'abord séparer les chiffres des dizaines, des unités et des centaines. Donc il faut trois variables, un pour le chiffre des unités, un pour le chiffre des dizaines et un pour le chiffre des centaines.

Phase 1 : De quelles variables aurons-nous besoin ?

E1 : On va prendre A en nombre initial, on va prendre B arrangé en ordre décroissant, C en ordre croissant, on va faire la différence de C moins B. On va le remettre dans A, et on va refaire avec B et C.
E2 : Ben si A c'est une variable où c'est un nombre, pour qu'il puisse le ranger dans l'ordre croissant et décroissant, faut d'abord séparer les chiffres des dizaines, des unités et des centaines. Donc il faut trois variables, un pour le chiffre des unités, un pour le chiffre des dizaines et un pour le chiffre des centaines.
E3 : On pourrait faire lire directement en entrée les 3 chiffres ?

Phase 1 : De quelles variables aurons-nous besoin ?

E1 : On va prendre A en nombre initial, on va prendre B arrangé en ordre décroissant, C en ordre croissant, on va faire la différence de C moins B. On va le remettre dans A, et on va refaire avec B et C.
E2 : Ben si A c'est une variable où c'est un nombre, pour qu'il puisse le ranger dans l'ordre croissant et décroissant, faut d'abord séparer les chiffres des dizaines, des unités et des centaines. Donc il faut trois variables, un pour le chiffre des unités, un pour le chiffre des dizaines et un pour le chiffre des centaines.
E3 : On pourrait faire lire directement en entrée les 3 chiffres ?

À la fin de cette phase, normalement les élèves :

- ont pris conscience de la nécessité de représenter les chiffres
- ont élaboré un plan en trois traitements partiels
  (l'enseignant peut être amené à suggérer une segmentation de l'algorithme)

Phase 2 : Rédiger les traitements partiels, chacun étant confié à un groupe différent

Phase 2 : Rédiger les traitements partiels, chacun étant confié à un groupe différent

Décomposer le nombre choisi en chiffres *Ex : 538 -> 5, 3, 8*	Tri des chiffres *Ex : 5, 3, 8 -> 3, 5, 8*	Passer des chiffres au nombre *Ex : 3, 5, 8 -> 358*

(Un autre plan serait possible en ne manipulant que des triplets possibles, mais il faut alors re-coder la soustraction.)

Phase 2 : Rédiger les traitements partiels, chacun étant confié à un groupe différent

Décomposer le nombre choisi en chiffres *Ex : 538 -> 5, 3, 8*	Tri des chiffres *Ex : 5, 3, 8 -> 3, 5, 8*	Passer des chiffres au nombre *Ex : 3, 5, 8 -> 358*

(Un autre plan serait possible en ne manipulant que des triplets possibles, mais il faut alors re-coder la soustraction.)

Trois types d'algorithmes : ▶ avec partie entière ▶ avec modulo ▶ avec les deux	Trois types d'algorithmes : ▶ 6 conditions décrivant les 6 permutations ▶ tests imbriqués ▶ fonctions `min` et `max` du langage	Un seul algorithme : ▶ $100 \times c + 10 \times d + u$

Phase 3 : Rédiger le traitement complet en organisant les traitements partiels

Les élèves choisissent librement les algorithmes pour chaque étape parmi ceux proposés à l'étape précédente.

Phase 3 : Rédiger le traitement complet en organisant les traitements partiels

Les élèves choisissent librement les algorithmes pour chaque étape parmi ceux proposés à l'étape précédente.

Que répondre à l'élève 3 (qui voulait prendre 3 chiffres en entrée) ?

Phase 3 : Rédiger le traitement complet en organisant les traitements partiels

Les élèves choisissent librement les algorithmes pour chaque étape parmi ceux proposés à l'étape précédente.

Que répondre à l'élève 3 (qui voulait prendre 3 chiffres en entrée) ?

Problème spécifique de la condition de l'itération

Deux points fixes possibles (0 et 495) à prendre en compte dans une *condition de continuation*

```
A != 495 and A != 0
```
ou bien
```
A != 495 or A != 0
```

Phase 3 : Rédiger le traitement complet en organisant les traitements partiels

Les élèves choisissent librement les algorithmes pour chaque étape parmi ceux proposés à l'étape précédente.

Que répondre à l'élève 3 (qui voulait prendre 3 chiffres en entrée) ?

Problème spécifique de la condition de l'itération

Deux points fixes possibles (0 et 495) à prendre en compte dans une *condition de continuation*

```
A != 495 and A != 0
```
ou bien
```
A != 495 or A != 0
```

Question laissée ouverte par l'enseignante : à décider après exécution sur machine (ce qui donne vite la réponse)

Phase 3 : Rédiger le traitement complet en organisant les traitements partiels

Les élèves choisissent librement les algorithmes pour chaque étape parmi ceux proposés à l'étape précédente.

Que répondre à l'élève 3 (qui voulait prendre 3 chiffres en entrée) ?

Problème spécifique de la condition de l'itération

Deux points fixes possibles (0 et 495) à prendre en compte dans une *condition de continuation*

```
A != 495 and A != 0
```
ou bien
```
A != 495 or A != 0
```

Question laissée ouverte par l'enseignante : à décider après exécution sur machine (ce qui donne vite la réponse)

Autres stratégies proposées :

- laisser à l'utilisateur la charge d'arrêter l'itération
- comparer le nouveau nombre à 3 chiffres avec l'ancien

Qu'ont appris les élèves ?

Qu'ont appris les élèves ?

- Le traitement informatique nécessite des étapes qui n'apparaissent pas explicitement dans le traitement manuel.

Qu'ont appris les élèves ?

- Le traitement informatique nécessite des étapes qui n'apparaissent pas explicitement dans le traitement manuel.
- Modularité : les traitements partiels peuvent être écrits et validés de façon indépendante.
  Ils sont en général assez simples pour être pris en charge par les élèves.

Qu'ont appris les élèves ?

- Le traitement informatique nécessite des étapes qui n'apparaissent pas explicitement dans le traitement manuel.
- Modularité : les traitements partiels peuvent être écrits et validés de façon indépendante.
  Ils sont en général assez simples pour être pris en charge par les élèves.
- Mais il faudra les intégrer, ce qui peut (doit) être prévu en amont

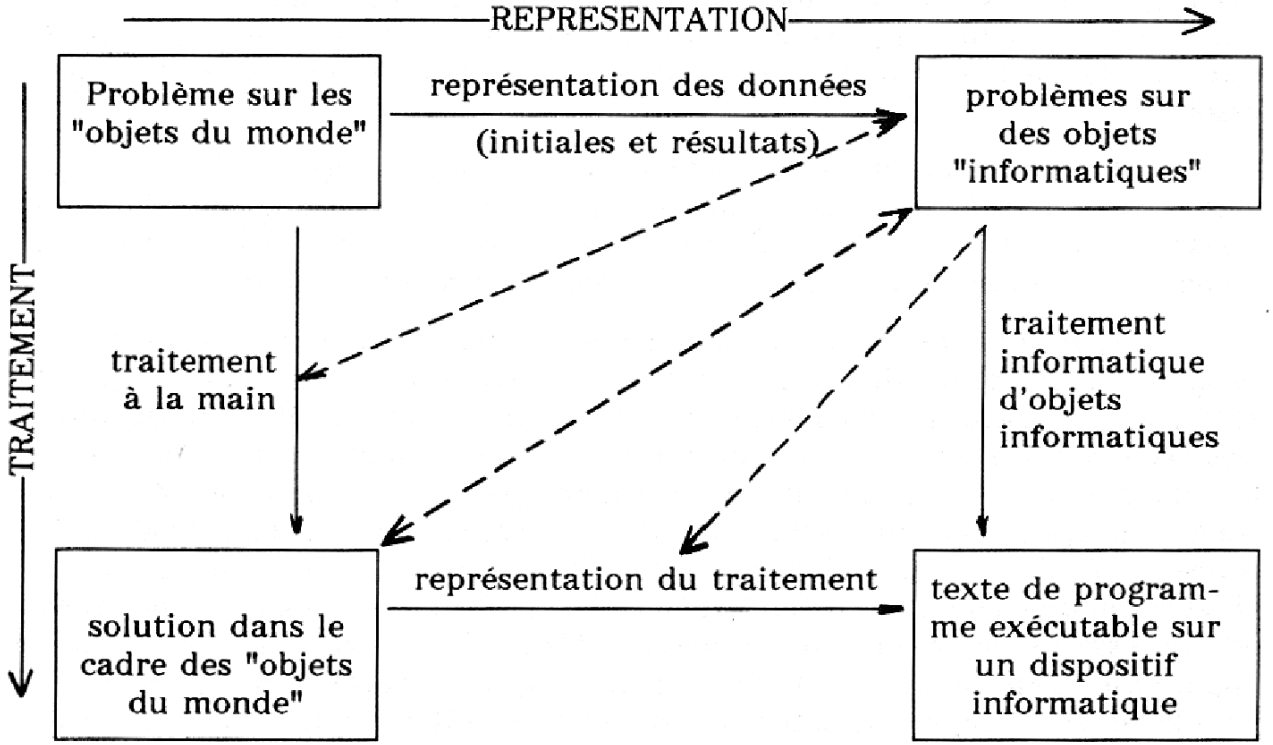
Qu'ont appris les élèves ?

- Le traitement informatique nécessite des étapes qui n'apparaissent pas explicitement dans le traitement manuel.
- Modularité : les traitements partiels peuvent être écrits et validés de façon indépendante.
  Ils sont en général assez simples pour être pris en charge par les élèves.
- Mais il faudra les intégrer, ce qui peut (doit) être prévu en amont

Remarque : Le problème de la condition de continuation est ici relégué à une vérification expérimentale par l'exécution et n'est pas traité sur le fond.

Plan

Activités cognitives dans les tâches de programmation (Rogalski)



- Deux dimensions : représentation des objets et traitement des données
- Deux voies de résolution
- En tirets : interactions entre les deux voies de résolution

Deux classes de problèmes de programmation

« Programmer c'est faire faire un calcul »

- À partir d'un problème concret, sur des objets bien connus : traitement (à la main) puis représentation du traitement
- Problèmes traitant d'objets déjà informatiques (comme sommer une liste de nombres) : la représentation des données est déjà fournie, il ne reste plus qu'à effectuer un traitement informatique

Deux classes de problèmes de programmation

« Programmer c'est faire faire un calcul »

- À partir d'un problème concret, sur des objets bien connus : traitement (à la main) puis représentation du traitement
- Problèmes traitant d'objets déjà informatiques (comme sommer une liste de nombres) : la représentation des données est déjà fournie, il ne reste plus qu'à effectuer un traitement informatique

Comment passer à « programmer c'est concevoir un algorithme » ?
(= représenter les objets pour faciliter leur traitement)

- Le programmeur doit déjà disposer de connaissances informatiques
- Il doit passer d'un modèle d'exécution pas-à-pas (comment agit le programme ?) à une conception du programme comme une fonction données $\rightarrow$ résultat (quelle fonctionnalité le programme réalise-t-il ?)
- Le problème doit être suffisamment complexe pour qu'il y ait un gain à le faire

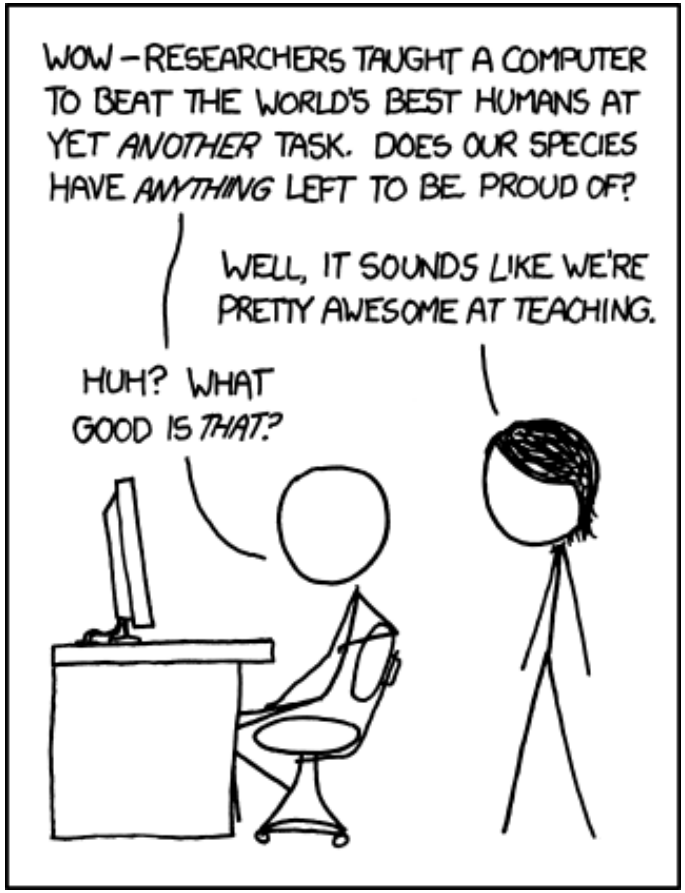
Incontournable : référence à la machine qui exécutera

Dans les débats entre élèves en phase 1 :

- « pour qu'il puisse le ranger dans l'ordre croissant et décroissant, faut d'abord séparer... »
- « L'ordinateur pour arranger les chiffres en ordre décroissant, ça va être important »

A person does not really understand something until he teaches it to someone else. Actually a person does not really understand something until he can teach it to a computer, that is, express it as an algorithm.

— Knuth, 1974



Notion de planification

Planification

anticipation d'une suite d'actions

Déjà présente en résolution de problèmes :

- identification dans l'énoncé des buts à atteindre
- choix des moyens à mettre en œuvre pour les atteindre

Notion de planification

Planification

anticipation d'une suite d'actions

Déjà présente en résolution de problèmes :

- identification dans l'énoncé des buts à atteindre
- choix des moyens à mettre en œuvre pour les atteindre

Pour la programmation :

- demande de se représenter le fonctionnement de la machine (opérations autorisées)
- la production d'un résultat n'est plus le seul critère d'analyse
- la production du programme lui-même devient un objectif ; celui-ci devra être explicité formellement
- la procédure élaborée ne pourra pas être modifiée en cours d'exécution

Deux notions utilisées (pas toujours consciemment) en planification informatique

Schéma

structure utilisée dans le traitement des informations pour atteindre des objectifs à petite échelle

Plan

ensemble organisé de schémas

Exemple : somme des n premiers carrés

On peut identifier deux schémas :

- calcul des n premiers carrés
- somme de n valeurs par accumulation

Le plan consiste à combiner ces deux schémas :

- deux boucles successives (et alors il faut mémoriser les valeurs)
- une seule boucle effectuant simultanément les deux tâches

Démarches de construction d'un programme

Approche descendante ou *top-down*

Penser le plan dans sa globalité, puis spécifier les sous-tâches jusqu'aux schémas élémentaires

Approche ascendante ou *bottom-up*

Partir de schémas connus pour les organiser en un plan

Un programmeur expert combinera les deux approches.

Et le programmeur « débutant » ?

Ce qui manque au débutant pour planifier efficacement :

- un répertoire de schémas dans lequel piocher selon ses besoins
- reconnaître les situations où de tels schémas s'appliquent
- penser des plans qui ne soient pas calqués sur le traitement manuel
- percevoir la nécessité d'une représentation adéquate des données (mais les problèmes pour débutants portent rarement sur des données complexes à représenter)

Les traitements partiels doivent être reconnus comme :

- nécessaires au traitement global
- possibles automatiquement
- pouvant s'insérer dans un traitement global

Un exemple de problème de planification (informatique *et* mathématique) (Saliba & Chiprianov)

On veut automatiser la construction du patron d'un prisme droit à base pentagonale régulière représenté ci-contre.

1. Construire le patron sur feuille.
2. En utilisant Scratch, automatiser la construction de votre patron. Vous pouvez choisir les longueurs que vous souhaitez pour AB et pour AH mais elles devront être précisées et notées. N'utilisez que des instructions de déplacements relatifs.
3. Modifier le programme pour construire un prisme avec une base à n cotés (n saisi par l'utilisateur).
   On pourra commencer par essayer de faire construire un patron de prisme à base hexagonale régulière.

Des solutions

Solutions attendues

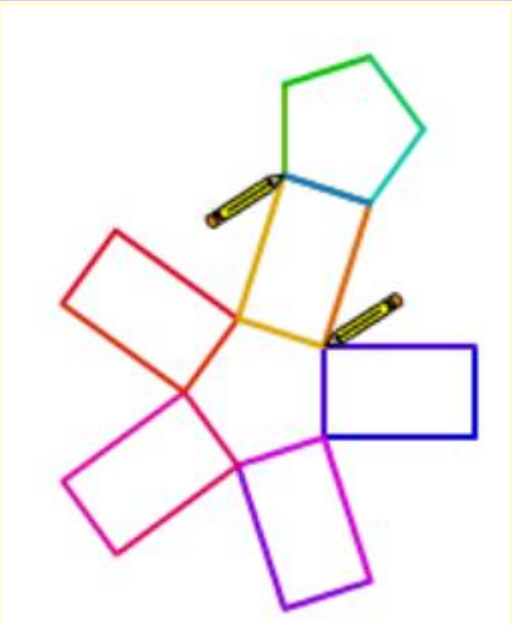
- à base de *motifs* (rectangle, côté) reproduits dans des *boucles*
- à base de *motifs* tracés par des *clones*

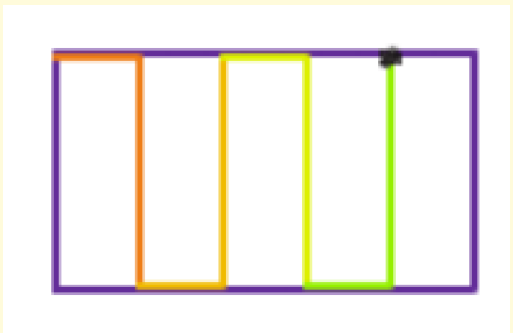
Des solutions

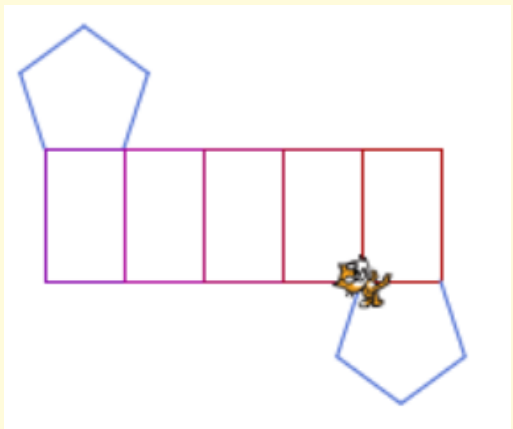
Solutions attendues

- à base de *motifs* (rectangle, côté) reproduits dans des *boucles*
- à base de *motifs* tracés par des *clones*

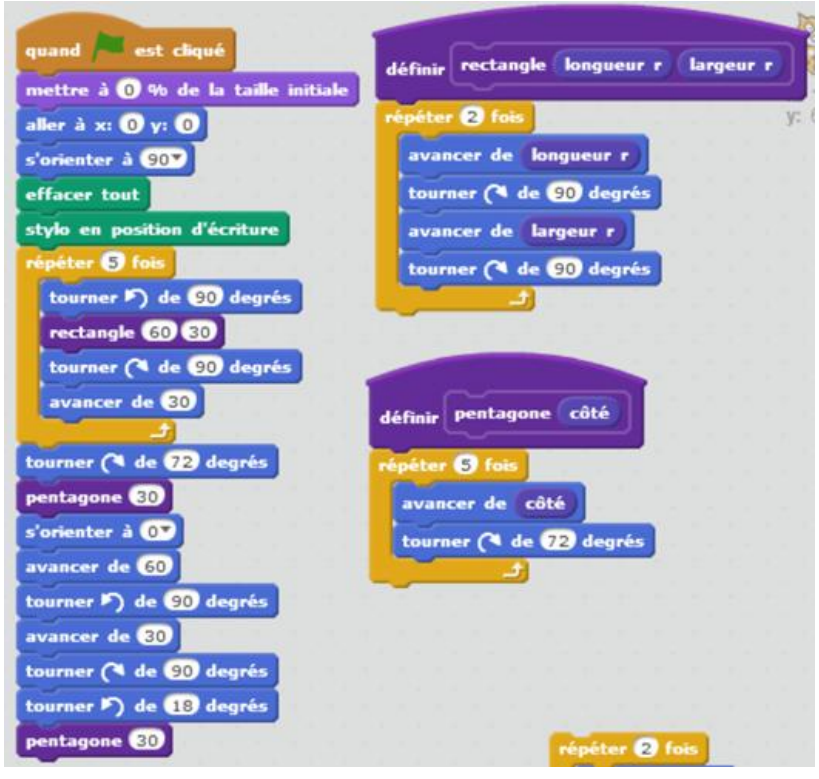
Des stratégies d'élèves







Où l'on retrouve deux classes de programmes



Stades d'apprentissage de la programmation (d'après Claude Pair)

1. Faire avec une machine
   (LOGO en commande directe, console Python, mais aussi d'autres logiciels de traitement de texte, de dessin...)

Stades d'apprentissage de la programmation (d'après Claude Pair)

1. Faire avec une machine
   (LOGO en commande directe, console Python, mais aussi d'autres logiciels de traitement de texte, de dessin...)
2. Faire faire

(Écrire un programme à l'avance comme une suite d'instructions)

Stades d'apprentissage de la programmation (d'après Claude Pair)

1. Faire avec une machine
   (LOGO en commande directe, console Python, mais aussi d'autres logiciels de traitement de texte, de dessin...)
2. Faire faire

   (Écrire un programme à l'avance comme une suite d'instructions)
3. Décrire un ensemble de calculs
   Un seul niveau d'itération, conditionnelles à une seule instruction : programmes encore très séquentiels

Stades d'apprentissage de la programmation (d'après Claude Pair)

4. Commenter et spécifier le programme
   (travail sur la dimension Représentation
   + recentre le programme comme objet plutôt qu'outil de l'activité)

Stades d'apprentissage de la programmation (d'après Claude Pair)

4. Commenter et spécifier le programme
   (travail sur la dimension Représentation
   + recentre le programme comme objet plutôt qu'outil de l'activité)
5. Concevoir à l'aide de procédures (éventuellement réalisées plus tard)

Stades d'apprentissage de la programmation (d'après Claude Pair)

6. Utiliser des variables avec des rôles avancés (accumulateur, information la plus pertinente)

```
pour i allant de 1 à 95:
  s = 0
  répéter
    lire c
    s = s + population de c
    jusqu'à fin de fichier ou département de c != i
  écrire i, s
```

Attention

- Ce découpage est discutable
- Pour le rendre opérationnel il faudra imaginer des situations d'apprentissage qui amènent à passer d'un stade au suivant

Plan

Généralisation

Quelles variables didactiques pourrait on identifier ou introduire dans cette situation ?
Pour viser quels apprentissages ?

Généralisation

Quelles variables didactiques pourrait on identifier ou introduire dans cette situation ?
Pour viser quels apprentissages ?

- Simple programmation de la recette pour un nombre donné ou vérification exhaustive ?
  $\Rightarrow$ itération simple ou imbriquée

Généralisation

Quelles variables didactiques pourrait on identifier ou introduire dans cette situation ?
Pour viser quels apprentissages ?

- Simple programmation de la recette pour un nombre donné ou vérification exhaustive ?
  $\Rightarrow$ itération simple ou imbriquée
- Changer le nombre de chiffres
  $\Rightarrow$ rendre un traitement manuel impraticable
  $\Rightarrow$ procédure de tri des chiffres plus générale
  $\Rightarrow$ procédure explicite de détection des cycles, donc mémorisation et parcours d'une suite de valeurs

Généralisation

Quelles variables didactiques pourrait on identifier ou introduire dans cette situation ?
Pour viser quels apprentissages ?

- Simple programmation de la recette pour un nombre donné ou vérification exhaustive ?
  $\Rightarrow$ itération simple ou imbriquée
- Changer le nombre de chiffres
  $\Rightarrow$ rendre un traitement manuel impraticable
  $\Rightarrow$ procédure de tri des chiffres plus générale
  $\Rightarrow$ procédure explicite de détection des cycles, donc mémorisation et parcours d'une suite de valeurs
- Écrire les nombres dans une autre base donnée, dans une base b quelconque
  $\Rightarrow$ renforcement sur la numération à position et/ou sur la base choisie

On peut même faire mettre au point certains de ces questionnements par la classe en demandant « *Pouvez-vous généraliser votre conjecture (et la vérifier à nouveau) ?* »

Autres paramètres (non manipulables en cours d'activité)

Autres paramètres (non manipulables en cours d'activité)

- Tâches à la charge des élèves et à celle de l'enseignant
- Travail préalable sur certains traitements partiels (décomposition en chiffres, tri)
  $\Rightarrow$ Facilite l'écriture de l'algorithme mais induit une démarche très ascendante

Autres paramètres (non manipulables en cours d'activité)

- Tâches à la charge des élèves et à celle de l'enseignant
- Travail préalable sur certains traitements partiels (décomposition en chiffres, tri)
  $\Rightarrow$ Facilite l'écriture de l'algorithme mais induit une démarche très ascendante
- Le langage de programmation utilisé
  - il doit être déjà maîtrisé des élèves
  - présence ou non d'une boucle « Répéter . . . jusqu'à »
  - primitives fournies (tri, sous-chaîne)

Autres paramètres (non manipulables en cours d'activité)

- Tâches à la charge des élèves et à celle de l'enseignant
- Travail préalable sur certains traitements partiels (décomposition en chiffres, tri)
  $\Rightarrow$ Facilite l'écriture de l'algorithme mais induit une démarche très ascendante
- Le langage de programmation utilisé
  - il doit être déjà maîtrisé des élèves
  - présence ou non d'une boucle « Répéter . . . jusqu'à »
  - primitives fournies (tri, sous-chaîne)
- Conception des traitements partiels par groupes distincts
  $\Rightarrow$ travail sur la modularité

Un prolongement (Laval)

Adaptez votre algorithme pour prouver la conjecture émise à propos de la recette de Kaprekar
Il s'agit ici de procéder par exhaustion de cas.

Un prolongement (Laval)

Adaptez votre algorithme pour prouver la conjecture émise à propos de la recette de Kaprekar
Il s'agit ici de procéder par exhaustion de cas.

E1 : Pour les 20 premiers entiers, ça marche à chaque fois... mais c'est long.
E2 : Tu veux que je te remplace ? On peut faire à tour de rôle les vérifications.
P : Où en êtes-vous ?
E2 : Nous vérifions à tour de rôle pour des groupes de nombres, mais c'est très long. Monsieur, il y a une astuce ?
E1 : Et si nous faisions un nouveau programme, où nous mettrions une boucle « Pour de compteur = 0 à 1000 ».
P : Ta proposition est pertinente. Mais est-il nécessaire de refaire tout l'algorithme ?
E1 : On va ajouter de nouvelles lignes au programme. Le prof a raison, nous n'avons pas besoin de tout refaire.

Plus tard...

E1 : Regarde, nous pouvons pas vérifier facilement la conjecture avec ces résultats.
E2 : Oui. Mais je vois pas ce qu'il faudrait faire.
E1 : Je pense qu'il est pas nécessaire d'afficher les étapes de calcul pour un nombre entier. Enlevons les lignes en trop.

```
***Algorithme lancé***
Pour n = 0
0 - 0 = 0
Pour n = 1
100 - 1 = 99
990 - 99 = 891
981 - 189 = 792
972 - 279 = 693
963 - 369 = 594
954 - 459 = 495
954 - 459 = 495
Pour n = 2
200 - 2 = 198
981 - 189 = 792
972 - 279 = 693
963 - 369 = 594
954 - 459 = 495
954 - 459 = 495
Pour n = 3
300 - 3 = 297
972 - 279 = 693
963 - 369 = 594
```

Plus tard...

E1 : Regarde, nous pouvons pas vérifier facilement la conjecture avec ces résultats.
E2 : Oui. Mais je vois pas ce qu'il faudrait faire.
E1 : Je pense qu'il est pas nécessaire d'afficher les étapes de calcul pour un nombre entier. Enlevons les lignes en trop.

```
***Algorithme lancé***
Pour n = 0
0 - 0 = 0
Pour n = 1
100 - 1 = 99
990 - 99 = 891
981 - 189 = 792
972 - 279 = 693
963 - 369 = 594
954 - 459 = 495
954 - 459 = 495
Pour n = 2
200 - 2 = 198
981 - 189 = 792
972 - 279 = 693
963 - 369 = 594
954 - 459 = 495
954 - 459 = 495
Pour n = 3
300 - 3 = 297
972 - 279 = 693
963 - 369 = 594
```

```
***Algorithme lancé***
Pour n = 0, le résultat obtenu est :0
Pour n = 1, le résultat obtenu est :495
Pour n = 2, le résultat obtenu est :495
Pour n = 3, le résultat obtenu est :495
Pour n = 4, le résultat obtenu est :495
Pour n = 5, le résultat obtenu est :495
Pour n = 6, le résultat obtenu est :495
Pour n = 7, le résultat obtenu est :495
Pour n = 8, le résultat obtenu est :495
Pour n = 9, le résultat obtenu est :495
Pour n = 10, le résultat obtenu est :495
Pour n = 11, le résultat obtenu est :495
```