ALGORITHMIQUE

Samedi 19 décembre

AlsaSup
Ecole Alsacienne

Programme de la séance

1. Langages et algorithmes
2. Variables
3. Constructions classiques

Algorithmes et Programmes

UNE RECETTE DE CUISINE

Leçon du jour : le crumble aux pommes

Une recette de cuisine

1. Eplucher les 6 pommes, en retirer les pépins, et les couper en petits morceaux

2. Faire cuire les morceaux de pommes à la poêle avec 50 g de sucre et un peu de cannelle

3. Faire préchauffer votre four à 180 °C et beurrer un plat

4. Mélanger dans un saladier 150g de farine, 100g de sucre et 75g de beurre pour obtenir un mélange sableux

5. Mettre les pommes dans le plat et verser le mélange par-dessus, et laisser cuire au four pendant 30 minutes.

Une recette de cuisine

1. Eplucher les 6 pommes, en retirer les pépins, et les couper en petits morceaux

2. Faire cuire les morceaux de pommes à la poêle avec 50 g de sucre et un peu de cannelle

3. Faire préchauffer votre four à 180 °C et beurrer un plat

4. Mélanger dans un saladier 150g de farine, 100g de sucre et 75g de beurre pour obtenir un mélange sableux

5. Mettre les pommes dans le plat et verser le mélange par-dessus, et laisser cuire au four pendant 30 minutes.

ALGORITHME

- Une description précise d'une méthode de résolution
- Une suite de tâches élémentaires connues de l'utilisateur qui va devoir les effectuer
- Caractéristiques requises :
  - Non-ambiguïté : il n'y a aucune initiative à prendre
  - Exhaustivité : tous les cas de figure sont prévus
  - Terminaison : on a toujours un résultat

La communication humaine

Au fur et à mesure que vous lisez cette phrase (un peu longue), vous identifiez inconsciemment les différents éléments qui la composent, ce qui vous permet de donner du sens à cette suite de symboles.

- La langue française est un moyen de communication, avec ses règles, ses codes, et son alphabet

- Lors que vous y êtes confronté, par écrit ou par oral, vous en déduisez "naturellement" le sens

Langages

- Il existe différents types de langages, dont tout le monde n'a pas la même maitrise :
  - Do you speak english ?
  - 你说汉语吗 ?
  - 
- Même s'il y a des erreurs dans une phrase, on parvient à en retrouver le sens
  - Exemple : *Sleon une edtue de l'Uvinertise de Cmabrigde, l'odrre des ltteers dnas les mtos n'a pas d'ipmrotncae*

Langages et machines

- Le problème est que les ordinateurs d'aujourd'hui sont beaucoup moins doués qu'un humain
  - Ils ne maitrisent ni le français, ni l'anglais, ni le chinois, ni l'elfique
  - S'il y a une erreur quelque part, ils sont perdus

- Pour contrôler un ordinateur, il est donc important de
  - Lui parler avec des mots simples
  - Dans une syntaxe clairement définie
  - Sans faire d'erreur

Algorithme et programme

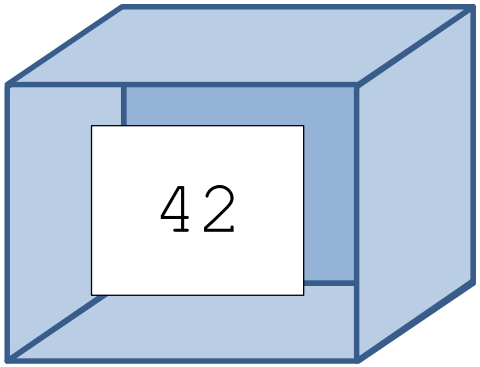
- Un programme, c'est la traduction d'un algorithme dans un langage compréhensible par une machine

- Il existe donc plusieurs programmes possibles pour un même algorithme
  - En fonction du langage de programmation choisi
  - En fonction des outils utilisés dans le programme

Variables

Qu'est-ce qu'une variable ?

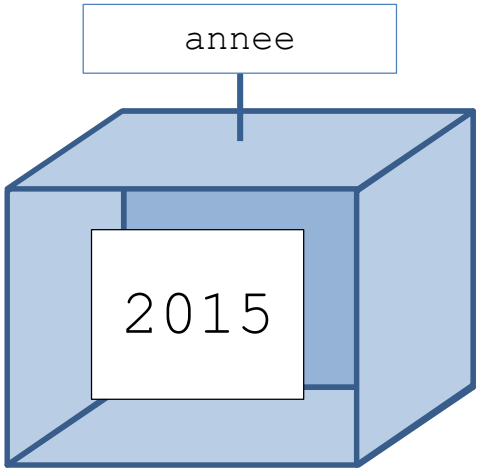
- Une variable est un objet stocké par l'ordinateur
- Informellement, c'est une sorte de boîte



- Et on va pouvoir stocker un élément à l'intérieur

Le nom d'une variable

- Une variable est désignée par son nom :
  - Choisir ce nom intelligemment
  - Eviter les espaces, les accents et les caractères spéciaux
  - Utiliser éventuellement des "traits bas" (underscore) (`par_ex`)
  - Jamais deux variables avec le même nom au même moment !

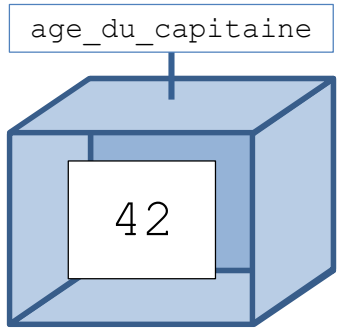


Déclaration de variable

- Une **déclaration** de variable consiste à indiquer à l'ordinateur la création d'une variable pour qu'il prenne les mesures nécessaires (notamment l'allocation de mémoire)

- Pseudo-code :

  `age_du_capitaine` ← `42` (nom pas encore utilisé)

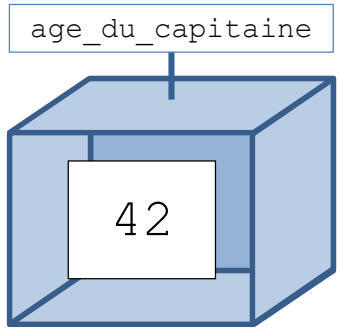


Accès à une variable

- Pour accéder au contenu d'une variable, il suffit d'utiliser son nom, et l'ordinateur se chargera de récupérer la valeur correspondante.

- Pseudo-code :

  `age_du_capitaine` (correspond au nombre 42)

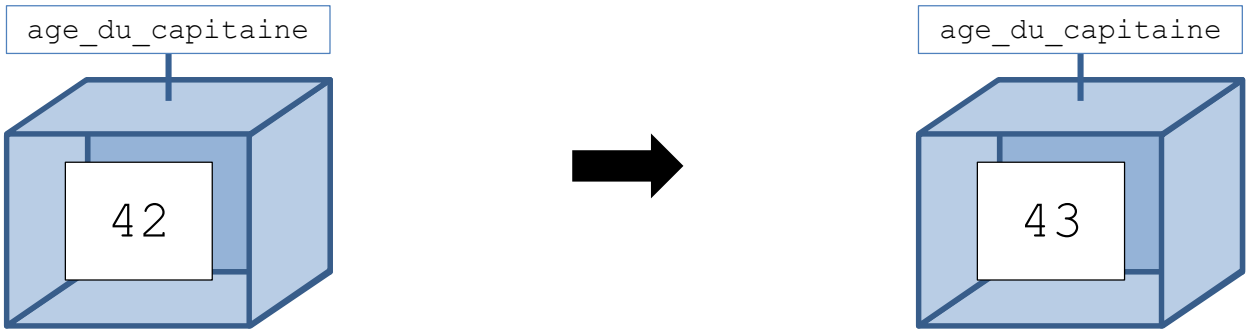


Changer la valeur d'une variable

- L'**affectation** consiste à donner une nouvelle valeur à une variable

- Pseudo-code :

  `age_du_capitaine` ← `43` (nom déjà existant)



Petit exercice

- Voici quelques lignes de pseudo-code

```
x ← 10
y ← x + 5
x ← 1
z ← y - 3
```

- **Question** : Que vaut `z` ?

CONSTRUCTIONS CLASSIQUES

Séquence

- Une séquence d'instructions permet d'exécuter plusieurs instructions l'une après l'autre

- Pseudo-code :

```
instruction1
instruction2
```

Séquence

- Dans une séquence, chaque instruction n'est exécutée qu'une fois que la précédente a été traitée

- On peut donc utiliser les résultats d'une instructions dans la suivante :

```
x = 24
resultat1 = gros_calcul_complexe(x)
resultat2 = autre_calcul_complexe(resultat1)
```

- La plupart des langages permettent de faire des calculs en parallèle : plusieurs instructions sont alors traitées en même temps (par exemple pour améliorer les performances).

Itération

- Parfois, il est nécessaire d'effectuer plusieurs fois la même action, voire la même suite d'actions : on parle d'itération, ou de boucle.

```
Pour chaque pomme,
     Faire
          Eplucher(pomme)
          RetirerPépins(pomme)
          CouperEnMorceaux(pomme)
     Fin faire
```

Boucles

Dans la plupart des langages, il existe deux types de boucles :

- Boucle `for`
  - Exécuter *n* fois un bout de code, ce nombre *n* étant défini à l'avance
  - Bien adapté aux tableaux et aux chaînes de caractères
- Boucle `while`
  - Exécuter un bout de code tant qu'une certaine condition n'est pas remplie (un test est effectué à chaque fois)
  - Bien adapté aux structures récursives
  - Risque de boucle infinie (le programme ne s'arrête jamais)

Boucle `for`

- Une boucle `for` commence par la déclaration d'un indice et de deux bornes (de type `int`) entre lesquelles cet indice va progressivement évoluer.

- Sauf indication contraire, après chaque passage dans le corps de boucle, l'indice est incrémenté (on lui ajoute 1).

- Lorsque l'indice atteint la deuxième borne, la boucle s'arrête, et on passe à la suite du programme
  - Attention, en général, la borne supérieure n'est donc pas incluse

UNE BOUCLE FOR EN PSEUDO-CODE

- Formulation propre en pseudo-code :

```
Pour p allant de 1 (inclus) à 7 (exclus)
    Faire
            Eplucher(p)
            RetirerPépins(p)
            CouperEnMorceaux(p)
    Fin faire
```

- Le corps de boucle se situe entre `Faire` et `Fin faire`

Boucle `while`

- Une boucle `while` dépend avant tout d'une condition, c'est-à-dire une question fermée (à laquelle on répond par oui ou par non).

- Au début de chaque passage dans la boucle, on vérifie que cette condition est vérifiée :
  - Si c'est le cas, on exécute le corps de boucle et on revient tester la condition
  - Sinon on passe à la suite du programme

UNE BOUCLE **WHILE** EN PSEUDO-CODE

- Formalisation propre en pseudo-code :

```
Tant que (il reste une pomme à couper)
    Faire
        p ← ChoisirUnePomme()
        Eplucher(p)
        RetirerPépins(p)
        CouperEnMorceaux(p)
    Fin faire
```

- Le corps de boucle se situe entre `Faire` et `Fin faire`

ATTENTION AUX BOUCLES INFINIES

- Si le test d'une boucle `Tant que` renvoie toujours vrai, le programme reste indéfiniment dans cette boucle

- Exemple idiot :

```
x = 0
Tant que (1 < 2)
   Faire
      x = x + 1
   Fin faire
```

- Exemple un peu moins idiot :

```
x = 0
Tant que (x < 10)
   Faire
      x = x + 1
   Fin faire
```

```
x = 0
Tant que (x < 10)
   Faire
      x = x - 1
   Fin faire
```

- Conclusion : attention avec les boucles `Tant que`!

PRÉTRAITEMENT ET POSTTRAITEMENT

- Il est fréquent de rencontrer des séquences d'instructions composées de :
  - Un pré-traitement
  - Une boucle (`for` ou `while`)
  - Un post-traitement

- Exemple :
  - Comment calculer la température moyenne de la semaine précédente ?
  - On suppose qu'on dispose d'une fonction `temperature` telle que `temperature(i)` est la température du $i^{ème}$ jour
  - Solution :

```
somme ← 0
Pour jour allant de 0 (inclus) à 7 (exclus)
   Faire
      somme = somme + temperature(jour)
   Fin faire
moyenne ← somme / 7
```

EXERCICES

- **Question** : Réécrivez ce programme en utilisant une boucle `Tant que` :

```
Pour indice allant de 0 (inclus) à 10 (exclus)
    Faire
        Afficher(indice)
    Fin faire
```

- **Réponse** :

```
indice ← 0
Tant que (indice < 10)
    Faire
        Afficher(indice)
        indice ← indice + 1
    Fin faire
```

Les conditions

- Parfois, plusieurs cas de figures sont possibles, et l'action à entreprendre dépend de la situation :

```
Si (Le feu est rouge)
    Alors
        S'arrêter
    Sinon
        Passer
```

ECRITURE EN PSEUDO-CODE

- Pseudo-code :

```
Si (condition)
    Alors
            Instruction1
    Sinon
            Instruction2
Fin si
```

SANS LE **ELSE**

- En pseudo-code :

```
Si (condition)
    Alors
            instruction1
Fin si
```