

# Algorithmes et langages

Fabrice EUDES, Pascal EVRARD, Philippe MARQUET,  
François RECHER & Yann SECQ

# Algorithmes

- Retour sur les mises en situation:
  - l'expression d'un algorithme (objets en Lego)
  - les problèmes de tri (crêpier) et de routage (baseball)
- Difficulté d'exprimer sans ambiguïté un procédé de résolution d'un problème
- Nécessite une « mécanisation » du processus de résolution à partir d'un nombre restreint d'opérations

# Algorithme & langages

- Comment exprimer la résolution automatisée d'un problème ?
- Le langage naturel est ambigu et les machines ne disposent que de primitives très simples
- Algorithme = description formalisée d'un processus automatisé de résolution d'un problème donné
- Langage = grammaire formelle permettant l'expression d'un programme transformable en un code interprétable par une machine

# Du problème à l'algorithme

- Plusieurs étapes pour aboutir à l'algorithme:
  - Compréhension du problème à résoudre (exemple)
  - Recherche d'un procédé (général!) de résolution
  - Description détaillée de ce procédé de résolution avec un ensemble restreint de « mots »
  - Vérification de la validité de l'algorithme sur différents jeux de données
- Loin d'être simple ... mais c'est tout l'intérêt ;)

# Langages informatiques

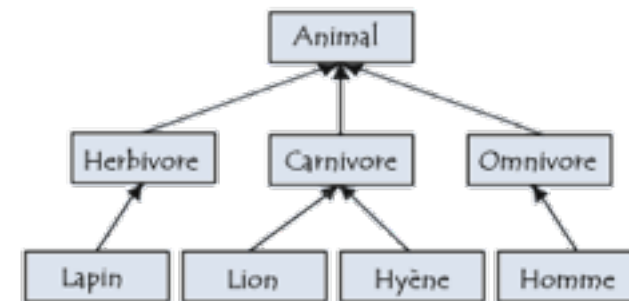
- Comment structurer les données et les traitements lorsque les algorithmes ou systèmes à réaliser deviennent complexes ?
- Plusieurs manières de modéliser le monde :)
  - Programmation impérative/structurée
  - Programmation orientée objets
  - Programmation fonctionnelle
  - Programmation logique
  - ...

# Programmation impérative

- Métaphore de la « recette de cuisine »
- L'algorithme décrit une séquence d'instructions transformant les données pour aboutir au résultat
- Organisation de la complexité grâce aux sous-programme et modules
- Typiquement: **C**, Pascal, BASIC, assembleur

```
int factorial(int n) {  
    int result = 1;  
    for (int i = 1; i <= n; ++i)  
        result *= i;  
    return result;  
}
```

# Programmation orientée objets



- Métaphore de la taxonomie:
- Regroupement de l'état (données) et du comportement (traitements) au sein d'un **objet**
- Relation est-un (héritage) et constitué-de (agrégation)
- Organisation de la complexité par des classes (le plus souvent)
- Langages: smalltalk, C++, Java, C#, ...

# Programmation fonctionnelle

- Métaphore ... programmer par composition de fonctions ?

```
(defun factorial (n)  
  (reduce #'* (loop for i from 1 to n collect i)))
```
- Concepts: fonction et liste, récursivité, immutabilité
- La référence: LISP (J. Mc Carthy)
- Revient en force avec l'explosion du parallélisme (avec erlang par exemple)
- Langages: LISP, Scheme, Haskell, JavaScript, OCaml  
...



# Programmation logique

- Métaphore de l'intelligence artificielle :)
- Principe: modélisation de faits et description de règles, ensuite un moteur de déduction tente de résoudre votre problème !
- Exemple classique: placement des 8 reines
- Langages: Prolog, Datalog, Oz, ...

# Algorithmes & langages

- Lego/crêpier/base-ball: illustration du processus de résolution et de la difficulté de son expression de manière non ambigu
- Importance de la compréhension du travail de filtrage des informations du problème et du jeu d'instructions disponible pour exprimer l'algorithme
- Possibilité d'expérimenter la construction d'algorithme dans des environnements plus simple que les langages évoqués avant: LOGO ou Scratch

# Le jeu dont vous êtes le héros

Fabrice EUDES, Pascal EVRARD, Philippe MARQUET,  
François RECHER & Yann SECQ

# Initiation à l'algorithmique

- Objectif: démystifier l'ordinateur en initiant de jeunes enfants à l'algorithmique
- Contexte: création d'un jeu de société
- Notions mobilisées: représentation dans l'espace (positionnement relatif/absolu), instruction, séquence d'instruction, exécution, validité, alternative (voire répétition) et créativité :)
- Expérience réalisée dans le cadre d'un TAP à V.A.

# Jeu dont vous êtes le héros

- Activité de création d'un jeu de société:
  - construction d'un plateau (grille 2D)
  - cartes pour les entités (héros, obstacles, objectif)
  - cartes pour les ordres (instructions)
- Support à la création d'algorithme de déplacement du héros sur le plateau

# Progression pédagogique

- Séance 1: création des éléments du jeu
- Séance 2: séquence, exécution, déplacements simples
- Séance 3: notion de condition (pas alternative) + positionnement relatif (jeu de rôle)
- Séance 4: notion de répétition
- Séance 5: notion de fonction

# Observations

- Curiosité et intérêt des enfants pour ce « jeu »
- Importance de l'aspect créatif et du jeu de rôle
- Notion de répétition: pas si simple :)
- Différences importantes entre CE2 et CM2 !
- Difficulté de gestion d'un groupe de 11 élèves dans un cadre non scolaire ...
- Expérience terriblement enrichissante pour tous :)