Notions Générales

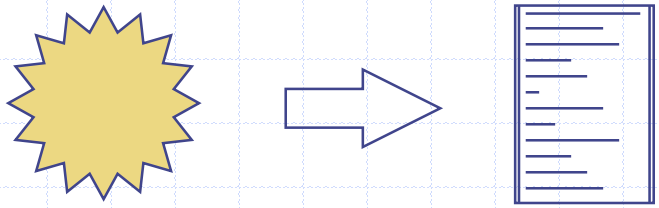
Séance 1

Année universitaire 2018/2019 – Semestre 1

Ordinateur et programmation

- L'orientation universitaire, l'inscription, la réservation d'un billet d'avion, … sont des opérations de la vie courante. Elles sont effectuées « par ordinateur ».
- On entend également des expressions du genre «l'erreur est due à l'ordinateur » ou encore «l'ordinateur s'est trompé ».
- Cela montre qu'il y a encore confusions et incertitudes mais surtout qu'il y a ignorance du rôle de la **programmation**.

La programmation



Problème **Programme**

- **Question à résoudre par une solution informatique**

- **Ensemble de <u>données</u>**
- **Ensemble de <u>résultats</u>**
  = solution informatique au problème
- **Description d'un ensemble d'actions**
- **Exécution dans un certain <u>ordre</u>**

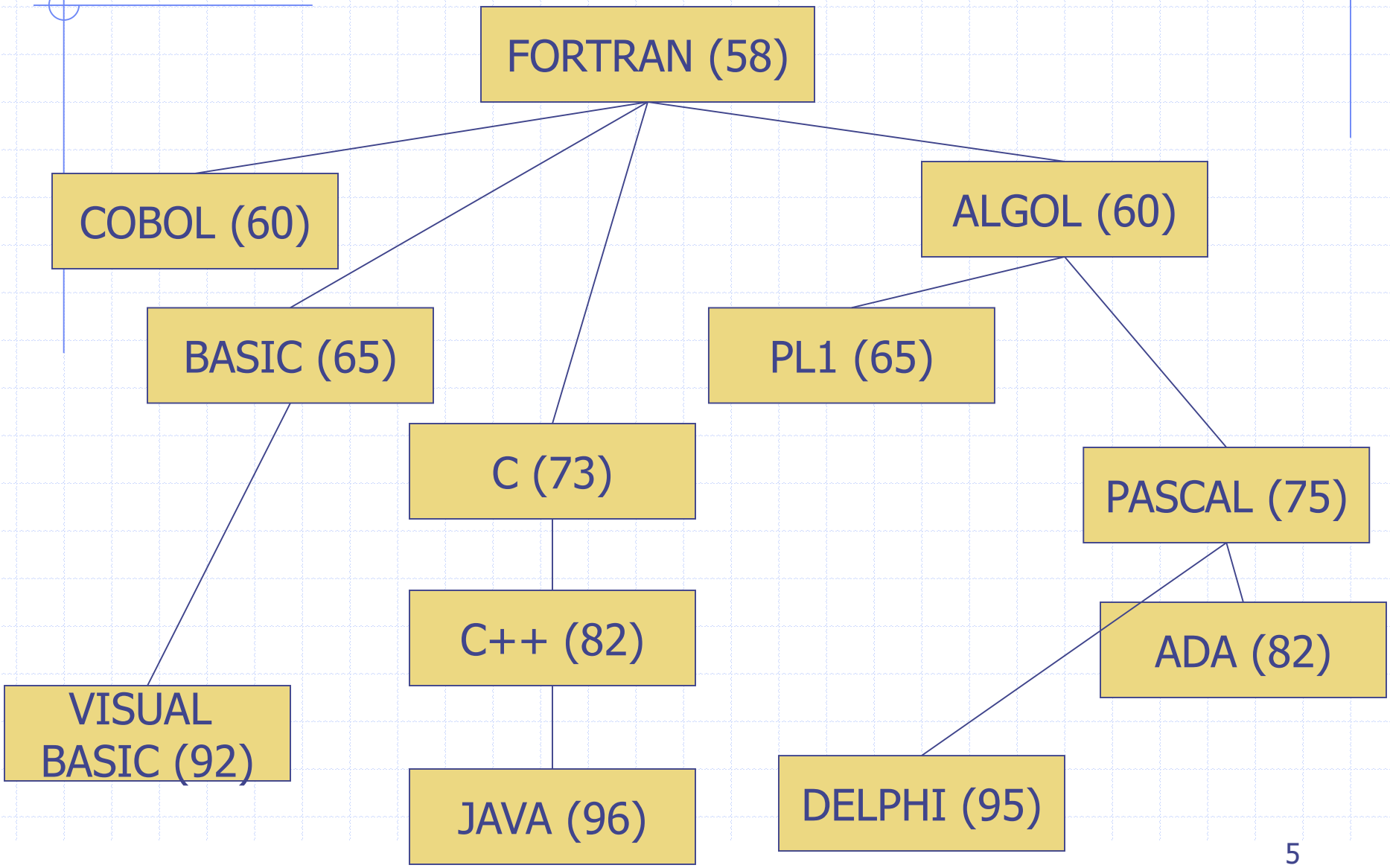
Quelle langue comprend l'ordinateur ?

- Uniquement le langage binaire
- Le processeur (élément central qui compose tout ordinateur) reconnaît un ensemble très limité d'instructions de base appelé jeu d'instructions.

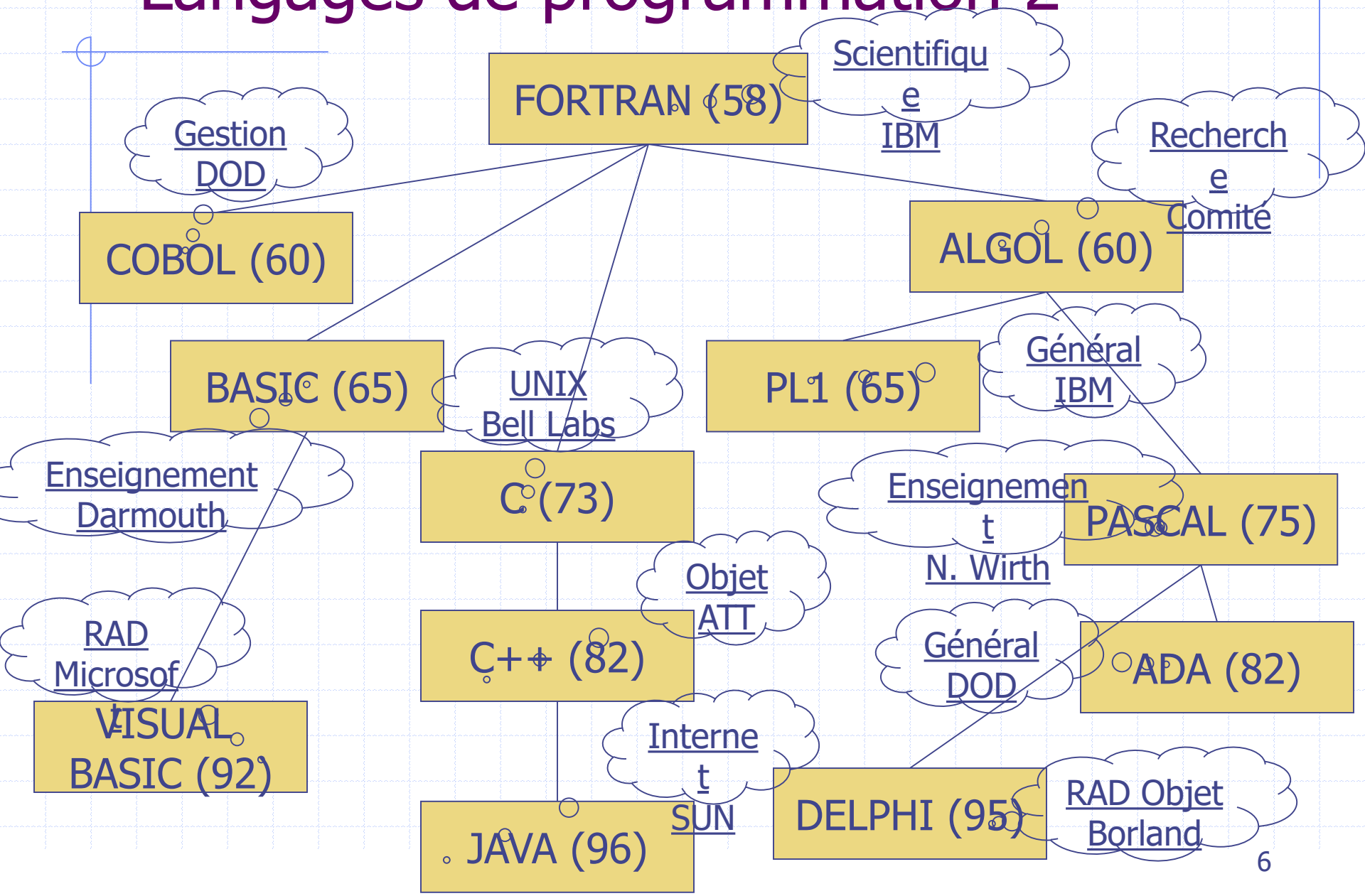
001	Charger: copie dans l'accumulateur le contenu du mot mémoire.
010	Ranger: copie le contenu de l'accumulateur dans le mot-mémoire adressé.
011	Additionner: remplace le contenu de l'accumulateur par la somme de son présent contenu et du contenu du mot-mémoire adressé.
100	Soustraire: remplace le contenu de l'accumulateur par la différence de son présent contenu et du contenu du mot-mémoire adressé.
101	Sauter: va à l'instruction dont on fournit l'adresse.
110	Sauter si non zéro: va à l'instruction dont on fournit l'adresse seulement si le présent contenu de l'accumulateur n'est pas zéro.
111	Stop: Arrêt de l'exécution du programme

- À partir de ces instructions élémentaires, des programmes complexes peuvent être construits.

Langages de programmation 1



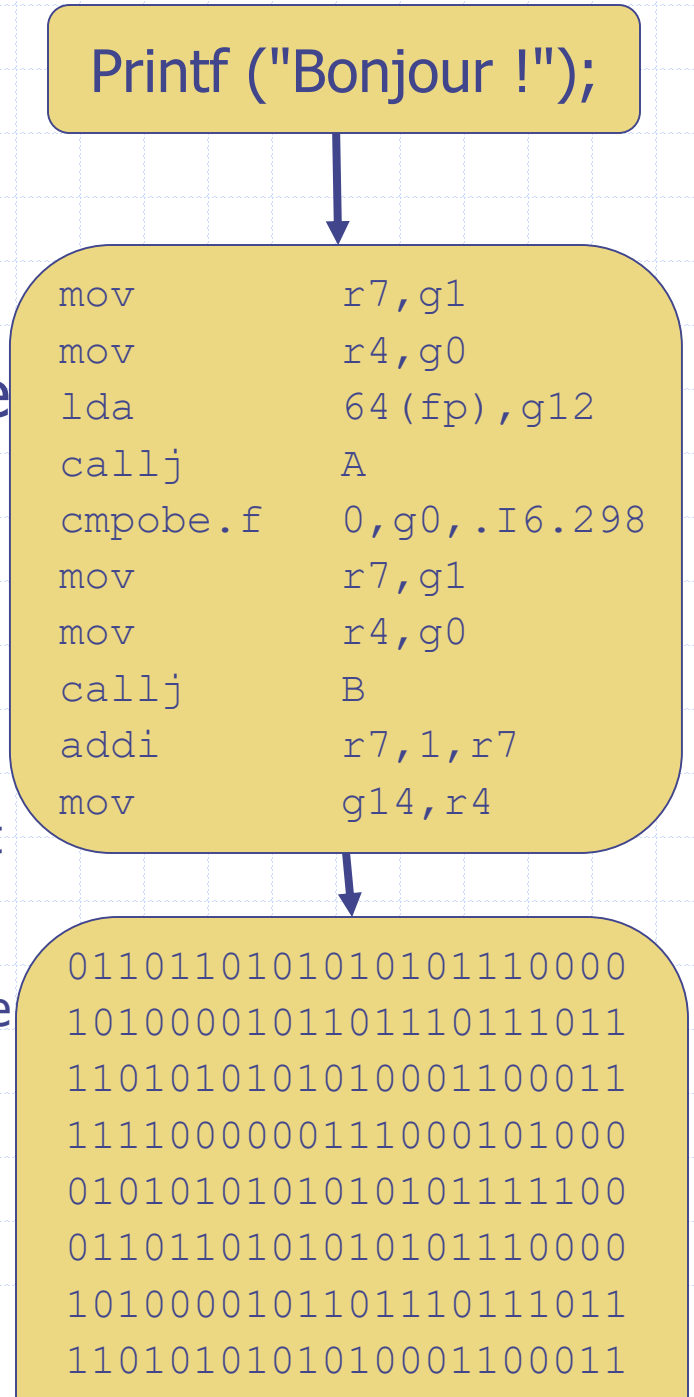
Langages de programmation 2



Compilateur

Le compilateur est un programme qui transforme un programme d'un code source vers le langage machine.

Le programmeur écrit du code source dans un langage de programmation comme **C, Ada, Pascal, Basic, Fortran ou Java**. Ces langages ont chacun une structure, une syntaxe et des règles que le programmeur doit suivre. Le compilateur traduit ce code source en des instructions que le processeur peut comprendre. Le code produit n'est compris que par un processeur spécifique, i.e Pentium, PowerPC, 68000, Alpha AXP, Intel i960 MX ou RISC 6000.



Compilation 1

Code Source (en C)

```
while (row <= max) && (col <= max)
{
    if (is_safe(row, col))
    {
        set_queen(row, col);
        col = col+1;
        row = min;
    }
    else row = row+1;
}
```

Compilation 2

Le compilateur traduit ceci en Assembleur

```
.I6.295:        # --label_295--
  ldis        40(r3),g7
  cmpibg.f    r4,g7,.I6.296
  cmpibg.f    r7,g7,.I6.296
  mov         r7,g1
  mov         r4,g0
  lda         64(fp),g12
  callj       _QueensnIs_safe289
  cmpobe.f    0,g0,.I6.298
  mov         r7,g1
  mov         r4,g0
  callj       _QueensnSet_queen283
  addi        r7,1,r7
  mov         g14,r4
  b           .I6.295
.I6.298:        # --label_298--
  addi        r4,1,r4
  b           .I6.295
.I6.296:        # --label_296--
```

Compilation 3

L'assembleur est ensuite traduit en langage binaire (représenté ici en héxadécimal)

```
40002E20:  C8B8E028
40002E24:  3925C03E
40002E28:  393DC03A
40002E2C:  5C881607
40002E30:  5C801604
40002E34:  8CE7E040
40002E38:  09FFFF08
40002E3C:  3204201E
40002E40:  5C881607
40002E44:  5C801604
40002E48:  09FFFE68
40002E4C:  59385087
40002E50:  5C20161E
40002E54:  08FFFFCC
```

```
01010101110010010010
11110010101010000010001
01010101110010010010
11110010101010000010001
01010101110010010010
11110010101010000010001
01010101110010010010
11110010101010000010001
01010101110010010010
11110010101010000010001
11110010101010000010001
01010101110010010010
11110010101010000010001
01010101110010010010
11110010101010000010001
01010101110010010010
```

Programmation …

- **Structurée**
- **Procédurale / Modulaire**
- **Impérative** / Déclarative / Fonctionnelle
- Événementielle
- Orientée objet

- Il y a aussi la programmation dite … Sauvage (Goto, les breaks,…)

Le langage C ?

```
#include <stdio.h>
int main ()
{
   printf ("Hello world !");
}
```

- C'est un langage
  - très répandu
  - très puissant
  - structuré
  - modulaire

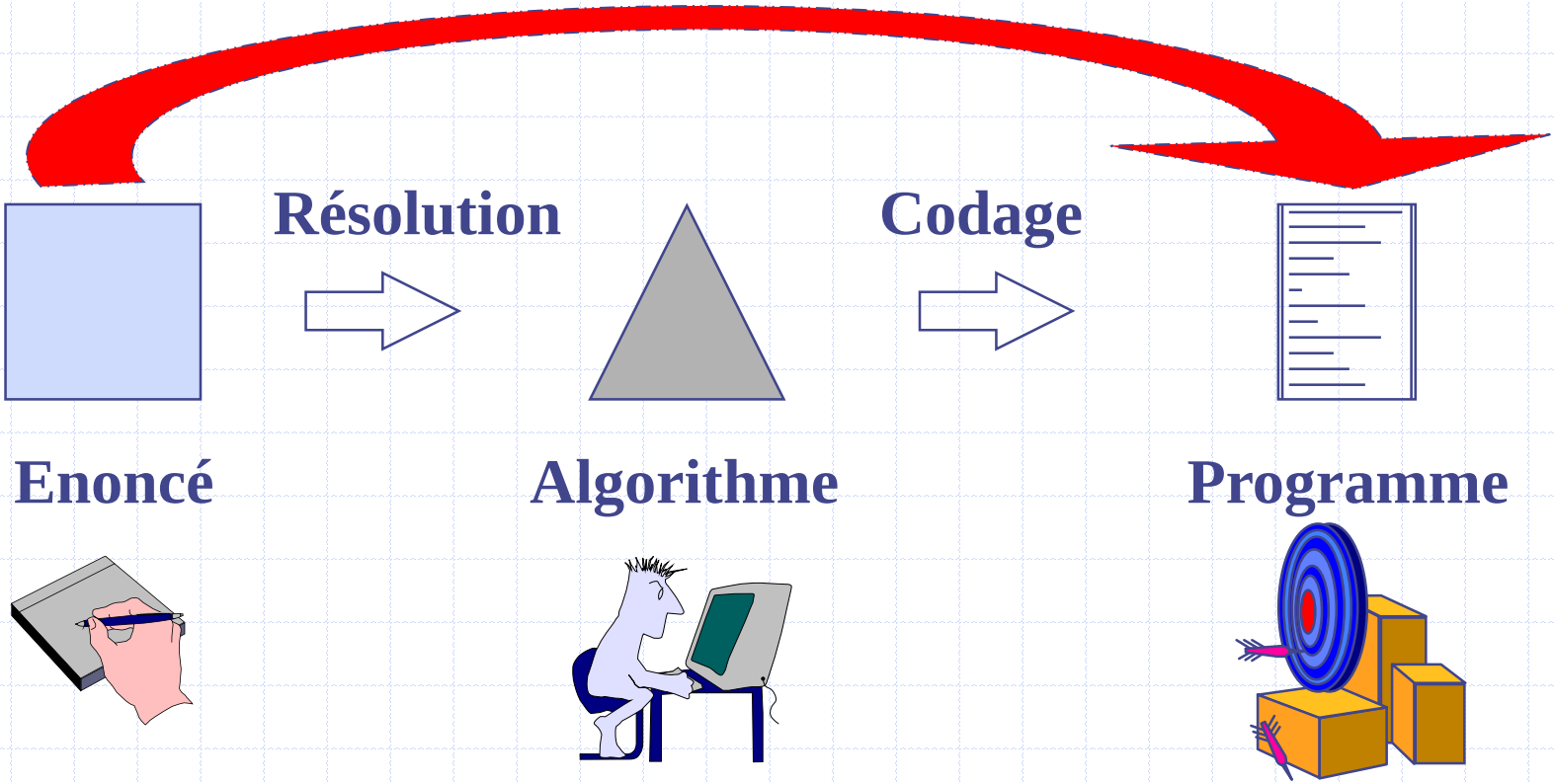
Applications

- Gestion des stocks d'une entreprise, calculs mathématiques, suivi de dossiers d'étudiants, facturation d'un magasin, gestion d'un hôpital,…
- Toutes ces applications variées sont possibles alors que l'ordinateur ne dispose que d'un jeu d'instructions très réduit. Ceci grâce à:

Des traitements répétitifs

Des traitements sélectifs

Notion d'algorithme

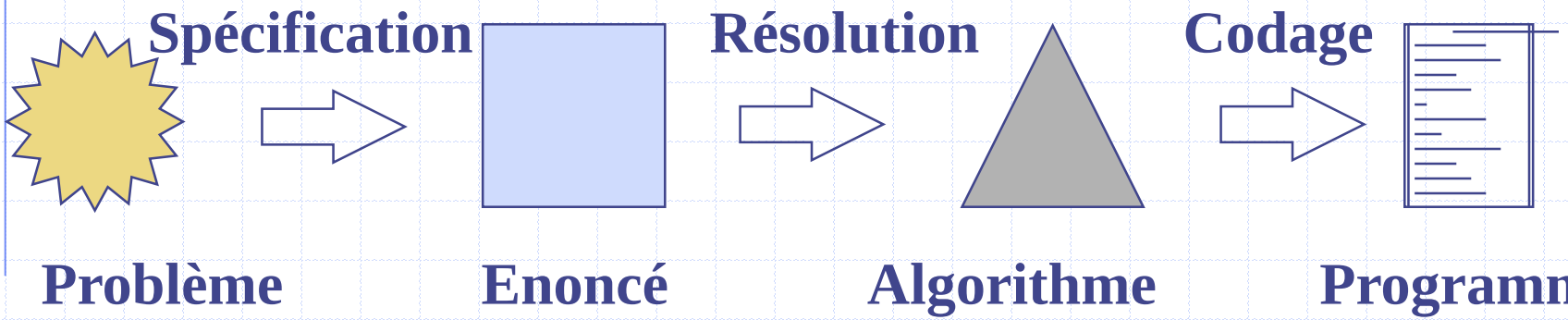


= Description d'un processus de résolution d'un problème bien défini

= Succession d'actions qui, agissant sur un ensemble de ressources (entrée), fourniront la solution (sortie) au problème

- Comment faire pour l'obtenir ?

Conception d'un programme



Problème: Division de deux nombres

Exemple de problème : division de deux nombres

Enoncé du problème : Diviser deux nombres

Questions :

- Quel genre de division ? exemple euclidienne
- Quel est le type des deux nombres ? exemple entiers strictement positifs
- Quel est le résultat demandé ? exemple quotient, reste

Problème : division de deux nombres

1. **Positionnement du problème** : Connaissant la dividende x, le diviseur y, calculer le quotient entier q, puis le reste N définis par la relation mathématique $x = yq+r$ avec $0 < r < y$
2. **Analyse du problème** :
    1. Etape 1 : Ordre de lecture du dividende x et du diviseur y.
    2. Etape 2 : Calcul du quotient entier q, calcul du reste par la formule $r = x – yq$
    3. Etape 3 : Edition des résultats
3. **Algorithme** : description systématique des étapes de l'analyse, sous forme d'une suite d 'actions convenablement enchaînées
4. **Codification** : traduire les actions (ou ensemble des étapes) dans un langage compréhensible par la machine.