Ministère de l'enseignement supérieur et de la recherche scientifique
Université Djillali BOUNAAMA - Khemis Miliana (UDBKM)
Faculté des Sciences et de la Technologie
Département de Mathématiques et d'Informatique

Chapitre 1

Introduction :
2. Introduction à l'Algorithmique

MI-L1-UEF121 : Algorithmiques et Structures de Données I

Noureddine AZZOUZA

n.azzouza@univ-dbkm.dz

Plan du Cours

1. L’algorithmique
   - 1.1 Définitions
   - 1.2 Exemples
   - 1.3 Historique
2. La programmation
   - 2.1 Définitions
   - 2.2 Langages et outils
3. Résolution d’un problème en informatique

L’algorithmique

Définition

- ✓ Un **ALGORITHME** c’est une ***suite d’instructions*** qui, une fois ***exécutée correctement***, conduit à un ***résultat donné*** (désiré).

- ✓ **Exemples : Algorithmes**

  - ➢ indiquer un chemin à un touriste égaré;
  - ➢ rédiger une recette de cuisine ;
  - ➢ Distribuer des boisson automatiquement;
  - ➢ Lire une vidéo sur YouTube

Exemple : Recette des crêpes

Préparation des crêpes

L’algorithme

1. Mettre la farine dans une terrine
2. Former un puits
3. Mettre les œufs entiers, le sucre, l’huile et le beurre
4. Mélanger délicatement avec un fouet en ajoutant le lait
5. Faire chauffer une poêle et y déposer quelques goutes d’huile
6. Faire cuire les crêpes à feu doux

Les actions

Le processeur

La personne qui réalise la recette

Définition : Processeur

- ✓ Un **algorithme** est toujours exécuté par un ***PROCESSEUR***.
- ✓ Un **algorithme** doit donc contenir uniquement des instructions compréhensibles par un ***PROCESSEUR.***
- ✓ **Exemples : Algorithmes (Processeur)**
  - ➢ indiquer un chemin à un touriste égaré (***une personne***);
  - ➢ rédiger une recette de cuisine (***une personne***);
  - ➢ Distribuer des boisson automatiquement (***une machine - distributeur***);
  - ➢ Lire une vidéo sur YouTube (***un programme***)

Définition : Environnement

- ✓ ***Environnement*** :
  - ➢ C'est l'ensemble des **objets** ou **éléments** nécessaires à la réalisation d'un travail décrit par un algorithme,
  - ➢ On distingue :
    - ❖ des ***objets d'entrée*** : fournies à l'algorithme.
    - ❖ des ***objets de sortie*** : produits par l'algorithme.
    - ❖ des ***objets internes*** : manipulation internes de l'algorithme.
  - ➢ L'environnement d'un algorithme peut être aussi appelé : le **paramétrage**

Définition : Action

✓ ***Actions :***

- Ce sont les « **suite d'instructions** » ou **étapes** de l'algorithme,
- C'est un ***événement*** de durée finie qui ***modifie*** l'environnement .
- Il faut noter que :
  - La modification de l'ordre des actions peut transformer le résultat
  - Une même action peut apparaitre plusieurs fois dans le même algorithme
- Une **action primitive** est une action exécutée (par un processeur) sans ***aucune information*** complémentaire.

Définition : Algorithme

Un **algorithme** est une ***séquence*** (suite) d’***actions primitives***, qui exécutées par un ***processeur*** bien défini réalisera un travail bien ***précis*** (demandé)

Propriétés

1. ***Généralité :*** un algorithme doit toujours être conçu de manière à envisager *toutes les éventualités* d'un traitement (tenir compte de *tous les cas possibles*).
2. ***Finitude :*** Un algorithme *doit s'arrêter* au bout d'un temps fini (*nombre fini d'actions* primitives).
3. ***Définitude :*** toutes les actions d'un algorithme doivent être *définies sans ambiguïté*
4. ***Répétitivité :*** généralement, un algorithme contient *plusieurs itérations*, c'est-à-dire des actions qui se *répètent* plusieurs fois.
5. ***Efficacité :*** Idéalement, un algorithme doit être conçu de telle sorte qu'il se déroule en un *temps minimal* et qu'il consomme un *minimum de ressources*.
6. ***Indépendance*** : un algorithme doit être *indépendant* des langages de programmation et des ordinateurs

Apprendre l'Algorithmique

Pour maitriser l'algorithmique, trois (3) qualités sont requises :

1. ***être méthodique*** :
   - ✓ Avant d'écrire les instructions d'un algorithme, il faut *analyser le problème* à résoudre. Il faut ensuite *définir les entrées et les sorties* de l'algorithme.
2. ***avoir de l'intuition*** :
   - ✓ Aucune recette ne permet de savoir a priori quelles instructions permettront d'obtenir le résultat voulu. Les *réflexes* du raisonnement algorithmique *deviennent spontanés* avec *l'expérience*.
3. ***être rigoureux. :***
   - ✓ Chaque fois qu'on écrit une série d'instructions, il faut systématiquement *se mettre* mentalement *à la place de la machine* qui va les exécuter.

Historique de l'Algorithmique

1. ***XVIII ème siècle av. J.-C.*** :

   ✓ les **Babyloniens** ont défini des descriptions exhaustives d'algorithmes pour des calculs concernant le commerce et les impôts ;

2. ***III ème siècle av. J.-C.*** :

   ✓ **Euclide** a introduit (dans son ouvrage les Eléments) le célèbre algorithme qui permet de trouver le plus grand diviseur commun (PGCD) de deux nombres ;

Historique de l'Algorithmique

1. ***IX ème siècle*** :

   ✓ **Al Khuwarizmi** été le premier qui a formalisé la notion d'algorithme dans son ouvrage L'algèbre et le balancement ;

2. ***XII ème siècle*** :

   ✓ **Adelard de Bath** a introduit le terme latin de algorismus (par référence au nom de Al Khuwarizmi);

La Programmation

Programme

- ✓ Un **programme** est une ***séquence d'instructions*** écrites dans un ***langage*** de programmation ***traduisant*** un algorithme. .
- ✓ Chacune de ses instructions spécifie l'opération que doit exécuter l'ordinateur .

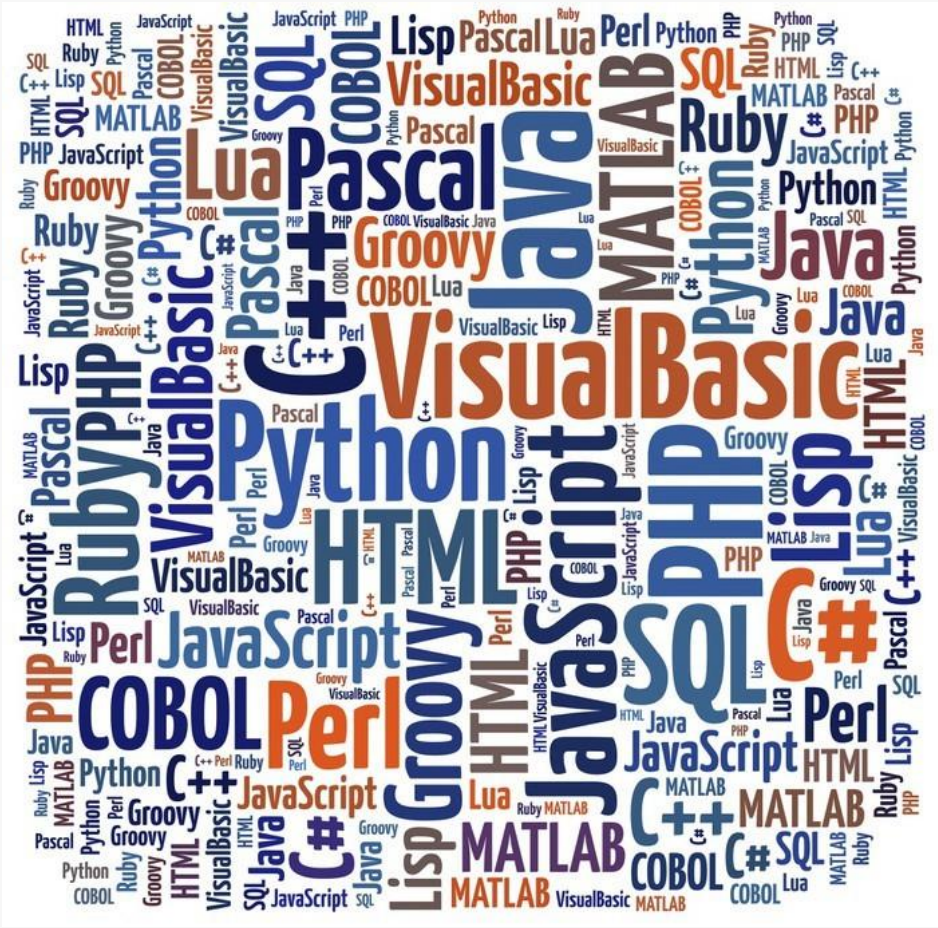
Algorithme vs. Programme

Algorithme	Programme
Nécessite pas de machine	Nécessite une machine
S'écrit selon un formalisme ou une description algorithmique	S'écrit selon un langage de programmation
Se déroule	S'exécute
Solution logique	Solution physique
C'est le résultat de l'analyse	C'est la traduction d'un Algorithme

Langage de Programmation

- ✓ Un **langage de programmation** est une notation conventionnelle destinée à ***formuler*** (traduire) des algorithmes et ***produire*** (développer) des programmes
- ✓ c'est une **abstraction** des opérations réalisable sur ordinateur.

- ✓ **Exemples :**
  - ✓ Pascal, C, Python, Java, C++, C#, PHP, JavaScript



Langage de Programmation

C

```c
#include <stdio.h>

int main()
{
    printf("Hello World");
    return 0;
}
```

C#

```csharp
using System;
namespace HelloWorldApp {

  class Geeks {

    static void Main(string[] args)
    {
        Console.WriteLine("Hello World");
        Console.ReadKey();
    }
  }
}
```

Java

```java
class HelloWorld {

    public static void main(String args[])
    {
        System.out.println("Hello World");
    }
}
```

Python

```python
print("Hello World")

```

Outils de Programmation

1. ***Editeur***:

   ✓ Un éditeur de texte ou de ***code source*** est un logiciel destiné à la création et l'édition de fichiers textes (fichiers ***source*** des programmes).

   Exemples: NotePad++, Sublime Text, Atom, Brackets …

2. ***Compilateur***:

   ✓ C'est un programme qui transforme un ***code source*** (écrit dans un langage de programmation) en un ***code objet*** afin de créer un ***programme exécutable*** par une machine.

   ✓ **Exemples:**

   - ***Langage C*** : GCC, Borland C,
   - ***Langage Pascal*** : Turbo Pascal, Free Pascal …

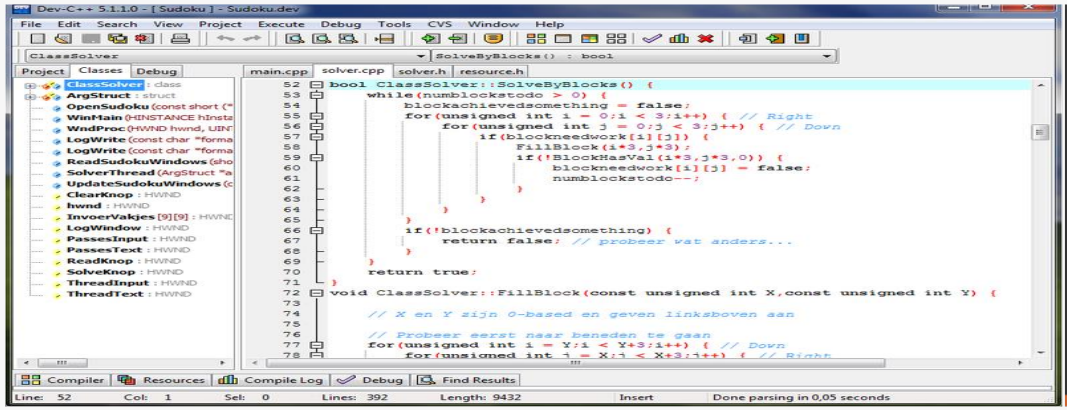
Outils de Programmation

3. ***IDE***:

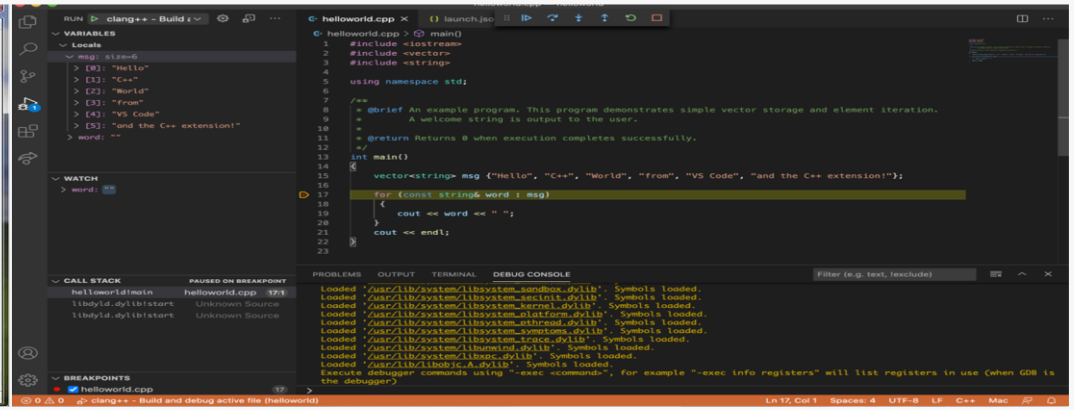
- ✓ L'environnement de Développement Intégré est (IDE) regroupe un ensemble d'outils spécifiques au développement des programmes. Il peut contenir :
  - ➢ Un éditeur de texte
  - ➢ Un compilateur
  - ➢ Un débogueur
  - ➢ Un créateur d'interface graphique ….etc
- ✓ **Exemples:**
  - ➢ ***Langage C/C++*** : DevC++, Code::Blocks, Visual Studio Code, Eclipse + CDT …
  - ➢ ***Langage Pascal*** : Lazarus, Free Pascal …
  - ➢ ***Python*** : PyCharm, Spyder, Visual Studio Code, Jupyter Notebook …

Outils de Programmation

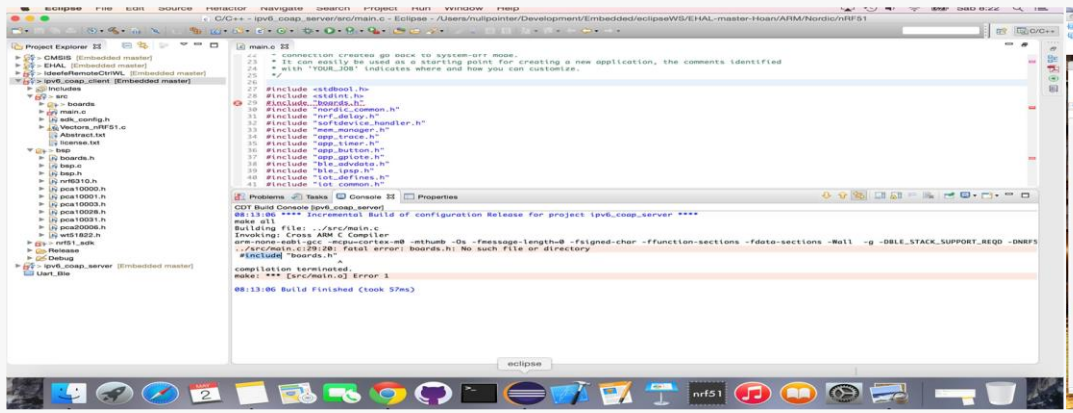
DevC++



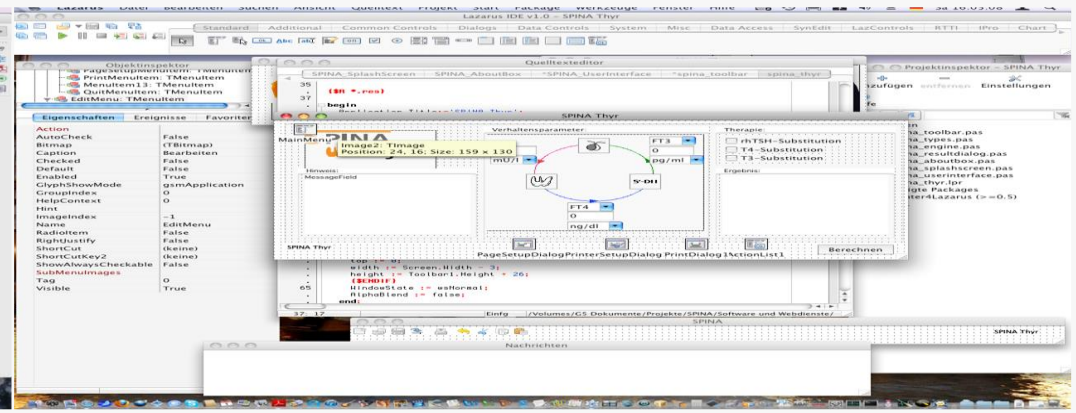
Visual Studio Code



Eclipse

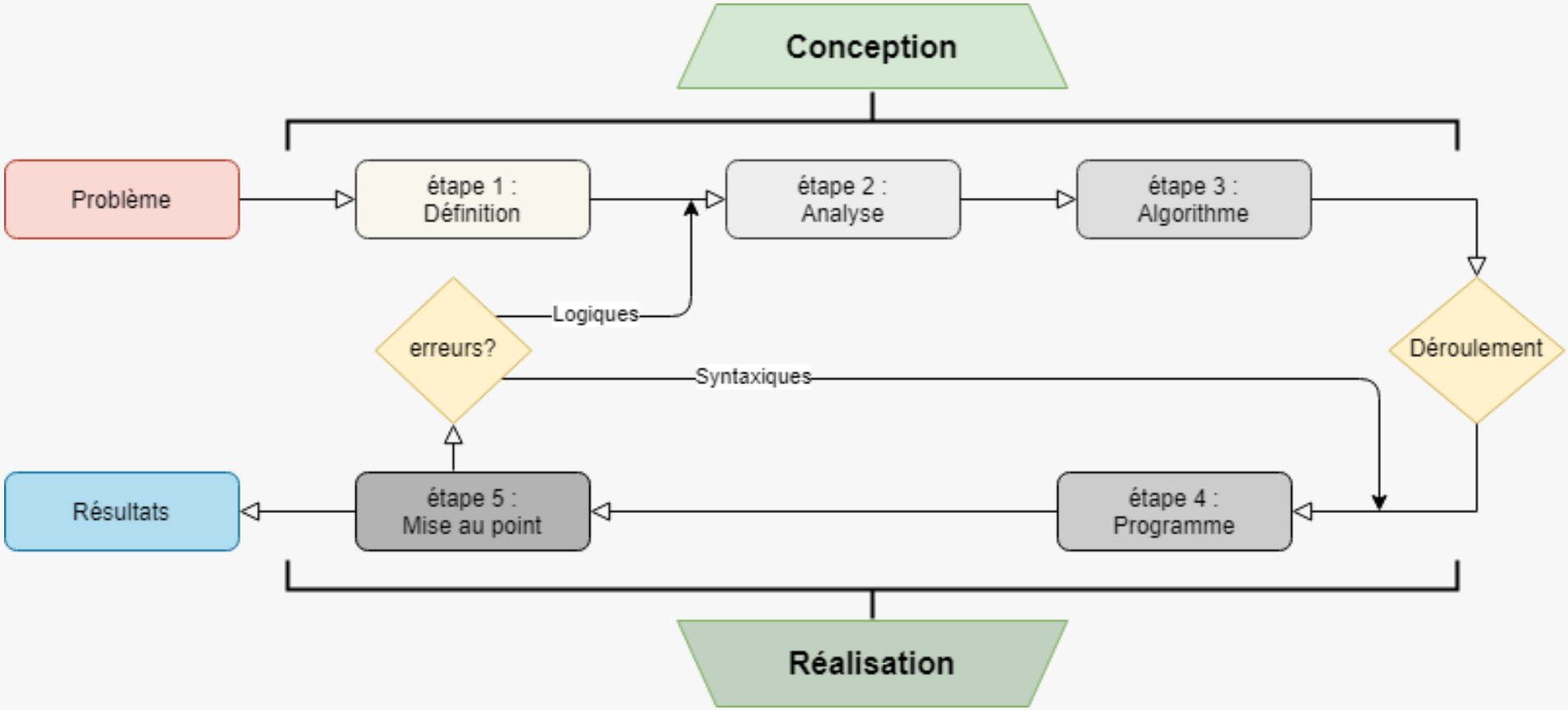


Lazarus



Du Problème au Résultats

Résolution d'un problème en informatique



Etape 1 : Définition du problème

- Il s'agit de :
  1. ***Déterminer*** toutes les ***informations disponibles***
  2. ***Définir*** la forme des ***résultats désirés***.

- Dans cette étape, on pose les questions suivantes :
  1. ***Qu***'est ce qui est ***donné*** comme ***entrées*** (***input*** ou ***données initiales***) ?
  2. ***Qu***'est ce qui est ***demandé*** comme ***sorties*** (***output*** ou ***résultats***) ?

Etape 2 : Analyse du problème

- Elle consiste de :
  1. ***Trouver*** le moyen de passer des ***données*** aux ***résultats***.
     a. Langage naturel, formule scientifique, schéma ou dessin, …
  2. ***Acquérir*** le réflexe de ***proposer des solutions*** adéquates à problème donné.
     a. ***Solution par théorie*** : problème mathématiques, physique, …
     b. ***Solution par synthèse*** : gestion d'un magasin, gestion des ventes, …
     c. ***Solution par expérience*** : problème de stockage, de recherche , …
- Dans cette étape, on pose la question suivante:
  1. ***Comment*** résoudre le ***problème*** ? Comment arrivé au ***résultats*** ?
  2. ***Quelle*** est la solution de ce ***problème*** ? Quelle est la forme de ***résultat***?

Etape 3 : Ecriture d'un Algorithme

- Il faut:
  1. ***Rédiger*** une ***solution claire*** et ***non ambiguë***.
  2. ***Représenter*** la solution par ***un algorithme*** écrit en « ***Langage Algorithmique*** » ou « ***Formalisme Algorithmique*** »
  3. ***Dérouler*** l'algorithme et vérifier son fonctionnement (cas ***général***, cas ***particuliers***)

- Dans cette étape, on pose les questions suivantes :
  1. ***Comment*** représenter la solution dans un algorithme ?
  2. ***Quelle structures de contrôle*** à utiliser ? Quel ***variables*** et ***structures de données*** à utiliser?

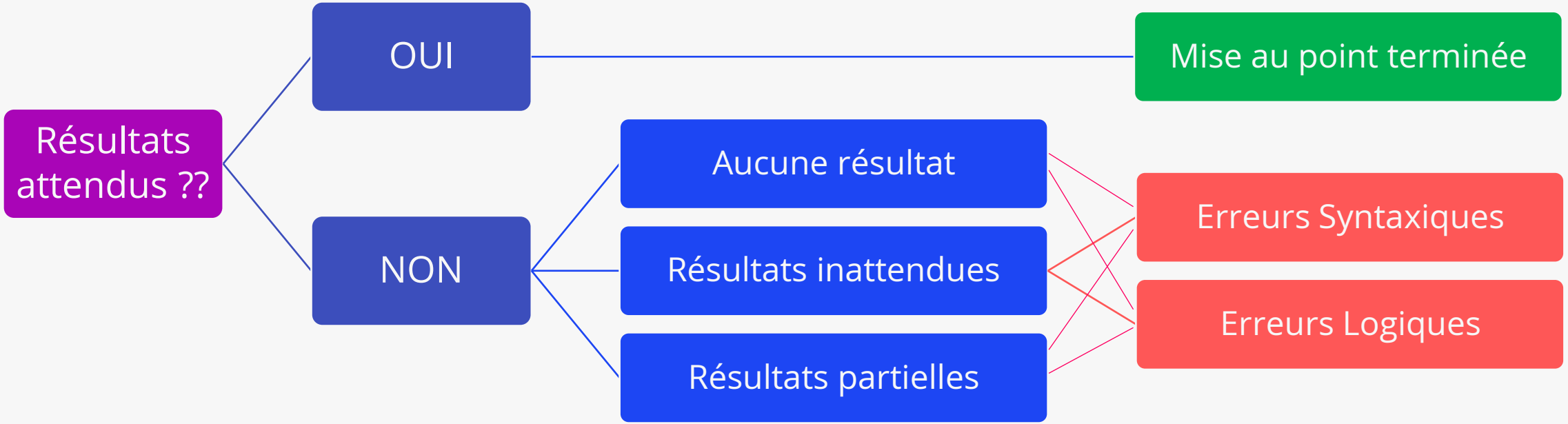
Etape 4 : Implémentation du Programme

- Pour implémenter concrètement l'algorithme, on doit:
  1. ***Choisir*** un ***langage de programmation***, et ***installer*** ces ***outils*** sur une machine (PC)
  2. ***Traduire*** l'***algorithme*** dans ce langage de programmation

- Dans cette étape, on pose les questions suivantes :
  1. ***Quel langage de programmation*** à utiliser ? (procédural, orienté objet, ...)
  2. ***Quel IDE*** ou ***éditeur de code*** à utiliser? (Payant, open source, gratuit, ...)

Etape 5 : Mise au Point

- Lors de l'exécution du programme sur une machine:
  1. La machine ***vérifie*** la ***syntaxe*** (orthographe) du ***programme***
  2. La machine ***traduit*** et ***exécute*** le sens exprimé par le ***programme***



Exemple : Diviseurs d'un Nombre

Problème : Retourner la liste des diviseurs d'un nombre

Problème : Retourner la liste des diviseurs d'un nombre

Etape 1 – Définition : Soit N entier; trouvez une solution qui permet d'afficher ces diviseurs.

Etape 2 – Analyse :

- On **divise** **N** par i=1, i=2, i=3, ….jusqu'à i=N/2 (la moitié de N)
  - à chaque fois on **vérifie** le reste de la division de **N** par i
    - Si c'est « 0 », on **affiche** le diviseur (i)

Exemple : Diviseurs d'un Nombre

Etape 3 – Algorithme

```
Algorithme Diviseurs;
Var N,i : entier ;

Debut
   //les entrées
   Lire (N);

    //manipulation des données
   Pour i ← 1 à N DIV 2 Faire
      Dpour
         Si N MOD i = 0 Alors
            Dsi
               //les sorties
               Ecrire (i,' est un diviseur de', N);
            Fsi
      Fpour
Fin.
```

Exemple : Diviseurs d'un Nombre

Etape 4 – Programme en langage PASCAL

```pascal
program diviseurs;
    uses Crt;
    var N, i : integer;

begin
    Readln(N);
    for i := 1 to N DIV 2 do
        begin
            if N mod i = 0 then
                begin
                    Writeln(i,' est un diviseur de ', N);
                end
        end
end.
```

Etape 4 – Programme en langage C

```c
#include <stdio.h>

int main()
{
    int N, i;
    scanf("%d", &N);

    for (i = 1; i < N / 2 ; i++)
    {
        if (N % i == 0)
        {
            printf("%d est un diviseur de %d \n", i, N);
        }

    }
    return 0;
}
```

Ministère de l'enseignement supérieur et de la recherche scientifique
Université Djillali BOUNAAMA - Khemis Miliana (UDBKM)
Faculté des Sciences et de la Technologie
Département de Mathématiques et d'Informatique

Chapitre 1

Introduction :
2. Introduction à l'Algorithmique

MI-L1-UEF121 : Algorithmiques et Structures de Données I

Noureddine AZZOUZA

n.azzouza@univ-dbkm.dz