

# THE MAIN TITLE OF THE LECTURE IN TWO LINES

## THE CHAPTER TITLE IN ONE LINE

🎓 The program name

🏛️ The university name - 2025/2026



**Idir AIT SADOUNE** 🌐

[idir.aitsadoune@centralesupelec.fr](mailto:idir.aitsadoune@centralesupelec.fr) ✉️

# IDIR AIT SADOUNE



- **Docteur en Informatique** diplômé par l'**ENSMA** en **2010**.
  - **Thèse** sur la **modélisation** et la **vérification** des services par une approche basée sur le **raffinement** et sur la **preuve**.



# IDIR AIT SADOUNE



- **Docteur en Informatique** diplômé par l'**ENSMA** en **2010**.
  - **Thèse** sur la **modélisation** et la **vérification** des services par une approche basée sur le **raffinement** et sur la **preuve**.
- **Enseignant** au sein du département **informatique** de **CentraleSupélec - Université Paris-Saclay**.



# IDIR AIT SADOUNE



- **Docteur en Informatique** diplômé par l'**ENSMA** en **2010**.
  - **Thèse** sur la **modélisation** et la **vérification** des services par une approche basée sur le **raffinement** et sur la **preuve**.
- **Enseignant** au sein du département **informatique** de **CentraleSupélec - Université Paris-Saclay**.
- **Chercheur** membre des pôles **Modèles** et **Preuve** du **LMF - Laboratoire Méthodes Formelles**.



Laboratoire  
Méthodes  
Formelles



# UN TITRE PRINCIPAL

## UN SOUS-TITRE

What are the main views on the relation between **logic and human** question ?

- What are the main views on the relation between **logic and human** question ?
- Why are straight-in approaches dangerous at uncontrolled airfields?
- Why the serpent was more crafty than any of the wild animals?
- Clone Kubuntu to different computer, different hardware
- Multirow colour and vertical alignment using tabularx
- What are the main views on the relation between logic and human question ?
- Why are straight-in approaches dangerous at uncontrolled airfields?

# OUTLINE

- The first chapter title
- The second chapter title
- The third chapter title
- Une image dans le texte

[Back to the begin](#) - [Back to the outline](#)

# OUTLINE

- The first chapter title
- The second chapter title
- The third chapter title
- Une image dans le texte

[Back to the begin](#) - [Back to the outline](#)

# LE TITRE DE LA SLIDE

- Un premier item pour introduire le point à aborder dans cette slide.
- Un **deuxième item** pour parler d'un concept lancé en 2025
  - un sous item pour détailler ce qui se passe au **premier semestre**
  - un autre sous item pour détailler ce qui se passe au **deuxième semestre**
    - juste pour préciser que la fin du deuxième semestre était **magnifique**
    - un lien aussi vers un site à visiter **adipiscing elit** 🌐
    - peut être un contact aussi **personne@exemple.com** ✉
- Un troisième item pour introduire le troisième point du cours
  1. on commence par le début d'un **point important**.
  2. on termine aussi par un point important à ne pas négliger.
- **void assertEquals(Object e, Object a)**
  - ➡ vérifie l'égalité entre deux objets  $e = a$  or  $e = a$ .
  - ✓ vérifie l'équivalence entre deux objets  $e \equiv a$  or  $e \equiv a$ .
  - ✗ vérifie la différence entre deux objets  $e \neq a$  or  $e \neq a$ .



# LE TITRE DE LA SLIDE SUR DEUX LIGNES

## Beast of Bodmin

A large feline inhabiting Bodmin Moor.

## Beast of Bodmin

A large feline inhabiting Bodmin Moor.

## Beast of Bodmin

A large feline inhabiting Bodmin Moor.

# DEFAULT BLOCKS AND **grep**

## Block Title

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

## Block Title

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

## Example

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

# SPECIFIC BLOCKS

## Definition

A prime number is a number that...

## Theorem (Pythagoras)

$$a^2 + b^2 = c^2$$

## Corollary

$$x + y = y + x$$

## Démonstration.

$$\omega + \phi = \epsilon$$



# OUTLINE

- The first chapter title
- The second chapter title
- The third chapter title
- Une image dans le texte

[Back to the begin](#) - [Back to the outline](#)

# EXAMPLE MATH

$$\dot{x} = \sigma(y - x)$$

$$\dot{y} = \rho x - y - xz$$

$$\dot{z} = -\beta z + xy$$

$$x = 3.14159265359 = \underbrace{314159265359}_{\text{significand}} \times \underbrace{10}_{\text{base}}^{\overbrace{-11}^{\text{exponent}}}$$

# TEST CENTER TEXT

# TEST CENTER TEXT

- Les **règles** et les **techniques** de programmation.

# TEST CENTER TEXT

- Les **règles** et les **techniques** de programmation.
- Le **support** des langages de programmation.



# TEST CENTER TEXT

- Les **règles** et les **techniques** de programmation.
- Le **support** des langages de programmation.
- Les **méthodologies de conception** et de développement.

# TEST CENTER TEXT

- Les **règles** et les **techniques** de programmation.
- Le **support** des langages de programmation.
- Les **méthodologies de conception** et de développement.
- Le **test**.

# TEST CENTER TEXT

- Les **règles** et les **techniques** de programmation.
- Le **support** des langages de programmation.
- Les **méthodologies de conception** et de développement.
- Le **test**.
- Les **méthodes formelles**.

# TEST CENTER TEXT

- Les **règles** et les **techniques** de programmation.
- Le **support** des langages de programmation.
- Les **méthodologies de conception** et de développement.
- Le **test**.
- Les **méthodes formelles**.

# OUTLINE

- The first chapter title
- The second chapter title
- The third chapter title
- Une image dans le texte

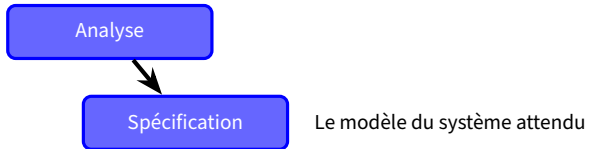
[Back to the begin](#) - [Back to the outline](#)

# CYCLE DE DÉVELOPPEMENT

# CYCLE DE DÉVELOPPEMENT

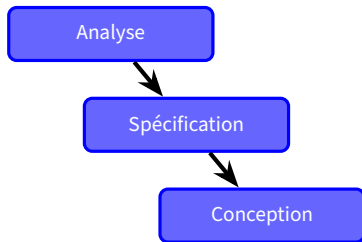
Analyse

# CYCLE DE DÉVELOPPEMENT

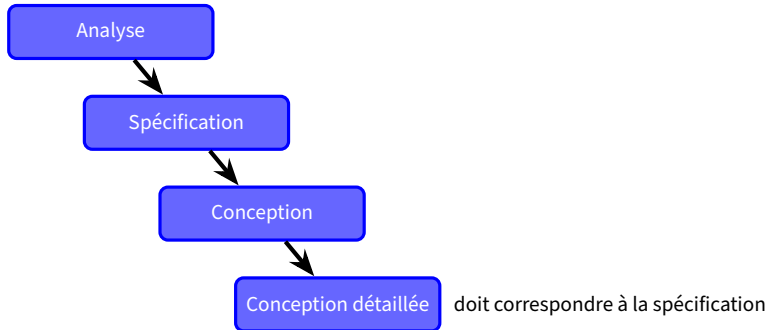




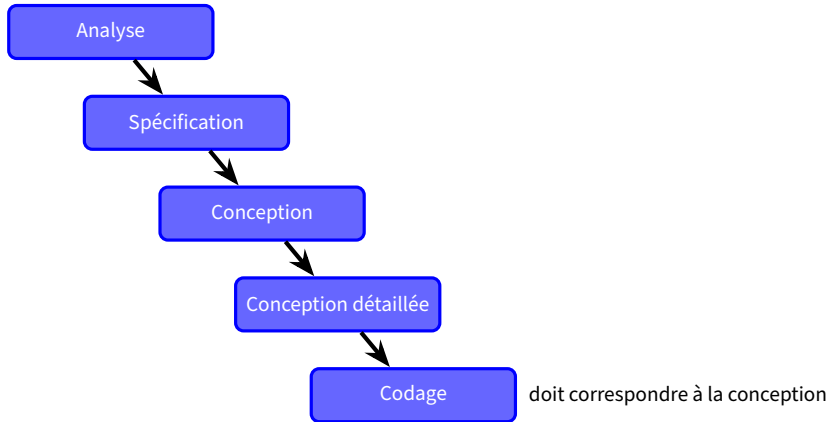
# CYCLE DE DÉVELOPPEMENT



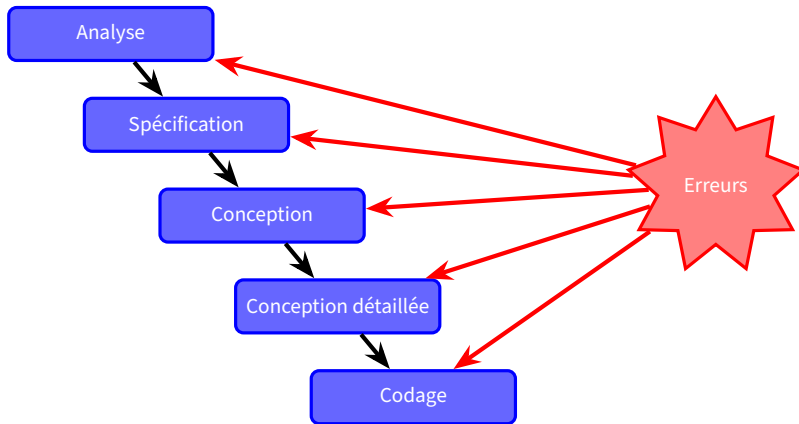
# CYCLE DE DÉVELOPPEMENT



# CYCLE DE DÉVELOPPEMENT



# CYCLE DE DÉVELOPPEMENT



Des **erreurs** possibles à toutes les étapes du développement.

# EXAMPLE CODE

```
1 #!/bin/bash
2 valid=true
3 count=1
4 while [ $valid ]
5     do
6         echo $count
7         if [ $count -eq 5 ];
8             then
9                 break
10            fi
11        ((count++))
12    done
```

```
$ cp file.txt directory
$ cd directory
$ ls -al .
```

# ASSEMBLER CODE

```
1 ;balance = balance + 1
2 mov eax, balance
3 add eax, 1
4 mov balance, eax
```

```
1 .MODEL SMALL
2 .STACK 100H
3 .CODE
4
5 MOV AX, 0x3C
6 MOV BX, 0000000000001010B
7 ADD AX, BX
8 MOV BX, 14
9 SUB AX, BX
10
11 MOV AH, 04FF
12 INT 21H
```

# ANIMATED CODE V1

```
1 package ltof.gameserver.model;  
2 /* *****  
3 Un commentaire sur plusieurs lignes  
4 ***** */
```

# ANIMATED CODE V1

```
1 package ltof.gameserver.model;
2 /* *****
3  Un commentaire sur plusieurs lignes
4  ***** */
5 public abstract strictfp class LtoChar extends LtoObject {
6     // Un commentaire sur une seule ligne
7     public static final Short ERROR = 0x0001;
8
9
10
11
12
13
14 }
```



# ANIMATED CODE V1

```
1 package ltof.gameserver.model;
2 /* *****
3  Un commentaire sur plusieurs lignes
4  ***** */
5 public abstract strictfp class LtoChar extends LtoObject {
6     // Un commentaire sur une seule ligne
7     public static final Short ERROR = 0x0001;
8
9     public void moveTo(int x, int y, int z) {
10         _ai = null;
11         log("Should not be called");
12         if (1 > 5) return;
13     }
14 }
```

# PROGRAMME vs PROCESSUS

```
1 int a = 3;  
2 a = a + 2;
```

```
1 @a: memval 3  
2   mov eax, a  
3   mov ebx, 2  
4   add ecx, eax, ebx  
5   mov a, ecx
```

```
1 2B50: mov eax, 2B1E  
2 2B52: mov ebx, #0002  
3 2B54: add ecx, eax, ebx  
4 2B55: mov 2B1E, ecx  
5 ...  
6 2B1E: 0003
```

```
1 2B50: mov eax, 2B1E  
2 2B52: mov ebx, #0002  
3 2B54: add ecx, eax, ebx  
4 2B55: mov 2B1E, ecx  
5 ...  
6 2B1E: 0003
```

# ANIMATED CODE V2

```
1 package ltof.gameserver.model;
2 /* *****
3  Un commentaire sur plusieurs lignes
4  ***** */
5 public abstract strictfp class LtoChar extends LtoObject {
6     // Un commentaire sur une seule ligne
7     public static final Short ERROR = 0x0001;
8
9     public void moveTo(int x, int y, int z) {
10         _ai = null;
11         log("Should not be called");
12         if (1 > 5) return;
13     }
14 }
```

# ANIMATED CODE V2

```
1 package ltof.gameserver.model;
2 /* *****
3 Un commentaire sur plusieurs lignes
4 ***** */
5 public abstract strictfp class LtoChar extends LtoObject {
6     // Un commentaire sur une seule ligne
7     public static final Short ERROR = 0x0001;
8
9     public void moveTo(int x, int y, int z) {
10         _ai = null;
11         log("Should not be called");
12         if (1 > 5) return;
13     }
14 }
```

# ANIMATED CODE V2

```
1 package ltof.gameserver.model;
2 /* *****
3  Un commenaire sur plusieurs lignes
4  ***** */
5 public abstract strictfp class LtoChar extends LtoObject {
6     // Un commentaire sur une seule ligne
7     public static final Short ERROR = 0x0001;
8
9     public void moveTo(int x, int y, int z) {
10         _ai = null;
11         log("Should not be called");
12         if (1 > 5) return;
13     }
14 }
```

# THE POWER OPERATOR

THEORY thy\_power\_operator

AXIOMATIC DEFINITIONS

operators

pow( $x \in \mathbb{Z}$ ,  $n \in \mathbb{N}$ ) :  $\mathbb{Z}$  INFIX //  $x$  pow  $n = x^n$

wd condition :  $\neg (x = 0 \wedge n = 0)$  //  $0^0$  is not defined

END

# THE POWER OPERATOR

THEORY thy\_power\_operator

AXIOMATIC DEFINITIONS

operators

pow( $x \in \mathbb{Z}, n \in \mathbb{N}$ ) :  $\mathbb{Z}$  INFIX //  $x$  pow  $n = x^n$

wd condition :  $\neg (x = 0 \wedge n = 0)$  //  $0^0$  is not defined

axioms

@axm1:  $\forall n. n \in \mathbb{N}_1 \Rightarrow 0 \text{ pow } n = 0$

@axm2:  $\forall x. x \in \mathbb{Z} \wedge x \neq 0 \Rightarrow x \text{ pow } 0 = 1$

@axm3:  $\forall x, n. x \in \mathbb{Z} \wedge x \neq 0 \wedge n \in \mathbb{N}_1 \Rightarrow x \text{ pow } n = x \times (x \text{ pow } (n - 1))$

...

END

# THE POWER OPERATOR

THEORY thy\_power\_operator

## AXIOMATIC DEFINITIONS

### operators

pow( $x \in \mathbb{Z}, n \in \mathbb{N}$ ) :  $\mathbb{Z}$  INFIX //  $x \text{ pow } n = x^n$

wd condition :  $\neg (x = 0 \wedge n = 0)$  //  $0^0$  is not defined

### axioms

@axm1:  $\forall n. n \in \mathbb{N}_1 \Rightarrow 0 \text{ pow } n = 0$

@axm2:  $\forall x. x \in \mathbb{Z} \wedge x \neq 0 \Rightarrow x \text{ pow } 0 = 1$

@axm3:  $\forall x, n. x \in \mathbb{Z} \wedge x \neq 0 \wedge n \in \mathbb{N}_1 \Rightarrow x \text{ pow } n = x \times (x \text{ pow } (n - 1))$

...

## THEOREMS

@thm1:  $\forall x, n, m. \dots \Rightarrow x \text{ pow } (n + m) = (x \text{ pow } n) \times (x \text{ pow } m)$

@thm2:  $\forall x, n, m. \dots \Rightarrow (x \text{ pow } n) \text{ pow } m = x \text{ pow } (n \times m)$

@thm3:  $\forall x, y, n. \dots \Rightarrow (x \times y) \text{ pow } n = (x \text{ pow } n) \times (y \text{ pow } n)$

...

END



# OUTLINE

- The first chapter title
- The second chapter title
- The third chapter title
- Une image dans le texte

[Back to the begin](#) - [Back to the outline](#)

# UNE IMAGE DANS LE TEXTE

## UN SOUS TITRE



CentraleSupélec

# UNE IMAGE DANS LE TEXTE

## UN SOUS TITRE

- Le Langage de Modélisation Unifié, (Unified Modeling Language - UML), est un langage de modélisation graphique à base de pictogrammes.



CentraleSupélec

# UNE IMAGE DANS LE TEXTE

## UN SOUS TITRE

- Le Langage de Modélisation Unifié, (Unified Modeling Language - UML), est un langage de modélisation graphique à base de pictogrammes.
- L'UML est une synthèse de langages de modélisation objet antérieurs : Booch, OMT, OOSE.



CentraleSupélec

# UNE IMAGE DANS LE TEXTE

## UN SOUS TITRE

- Le Langage de Modélisation Unifié, (Unified Modeling Language - UML), est un langage de modélisation graphique à base de pictogrammes.
- L'UML est une synthèse de langages de modélisation objet antérieurs : Booch, OMT, OOSE.
- UML 1.0 a été normalisé en janvier 1997; UML 2.0 a été adopté par l'OMG en juillet 2005. L'UML est une synthèse de langages de modélisation objet.



CentraleSupélec

# THANK YOU

[Back to the begin](#) - [Back to the outline](#)