

UNIVERISTE NICE
SOPHIA-ANTIPOLIS
TRAVAIL D'ETUDE
LICENCE INFO
2002-2003

NOM :

BOUKHOURROU Sophiane

GATTI Nicolas

Encadreur :

CRESCENZO Pierre

*Etude comparative des
mécanismes d'héritage dans les
langages C++ , C# , Eiffel et
Java*

**Quelles sont les similitudes , les
différences , les qualités et les
défauts de ces 4 langages du point
de vue de l'héritage**

SOMMAIRE :

1. *Qu'est ce que l'héritage ?*
2. *Différents types d'héritage*
3. *Héritage simple*
4. *Héritage multiple*
5. *Conclusion*

Définition de l'héritage

◆ Qu'est ce que l'héritage ??

◆ Pourquoi l'héritage ??

Définition de l'héritage

◆ *Intérêts principaux*

- ↗ La notion de généralisation et de spécialisation
- ↗ Récupération des attributs et méthodes, sans avoir à la réécrire totalement.
- ↗ Élimination des redondances du code

Différents types d'héritage

◆ Héritage simple de classe

➤ Qu'est ce que l'héritage simple ?

Exemple :

Un avion à réaction est un genre particulier d'avion

Différents types d'héritage

◆ Héritage multiple de classe

↗ Qu'est ce que l'héritage multiple de classe ??

Exemple :

Un hydravion est à la fois bateau et avion

↗ Inconvénient de l'héritage multiple .

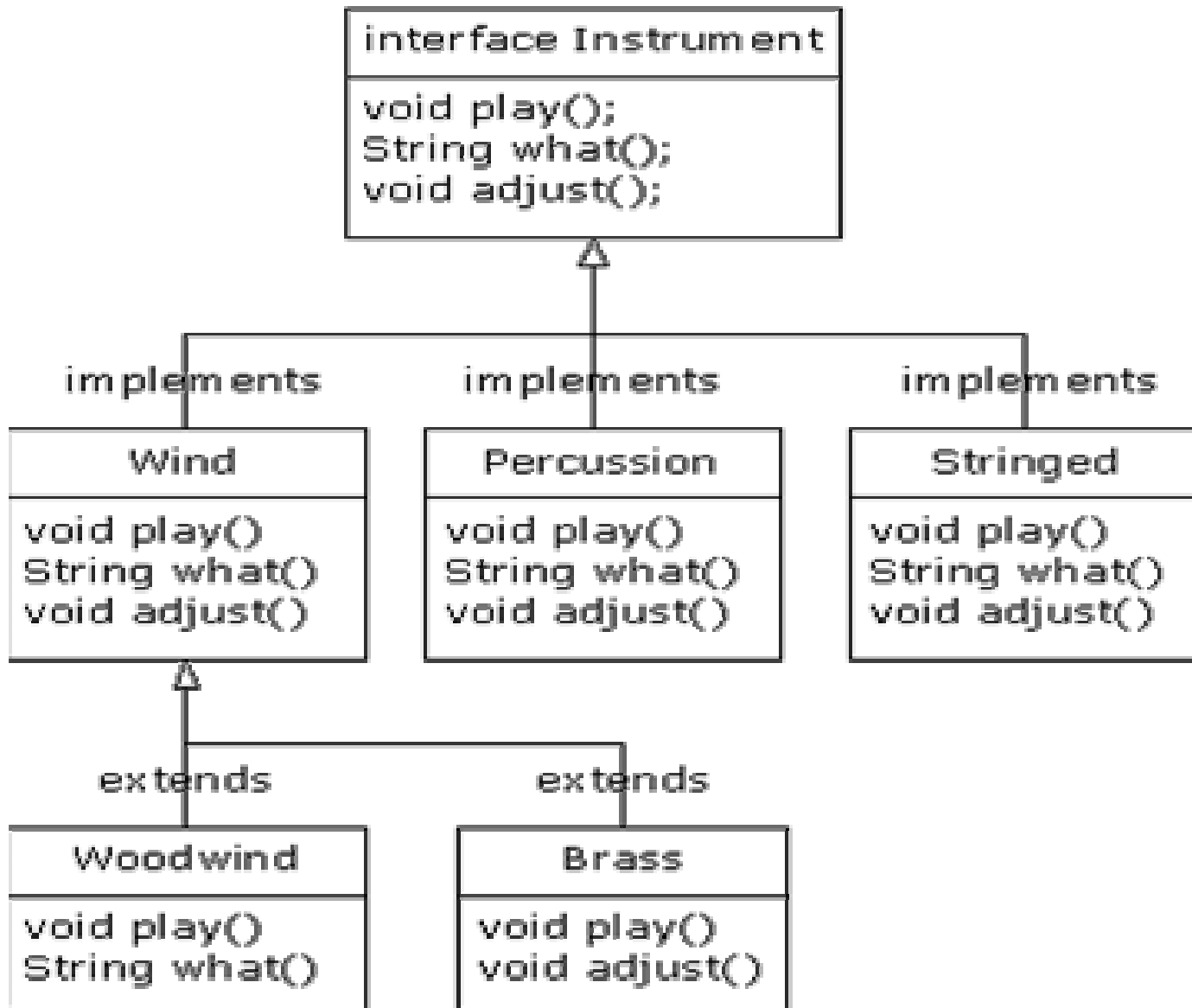
Différents types d'héritage

◆ Héritage multiple d'interface

↗ Qu'est ce que les interfaces ??

↗ Intérêt des interfaces ??

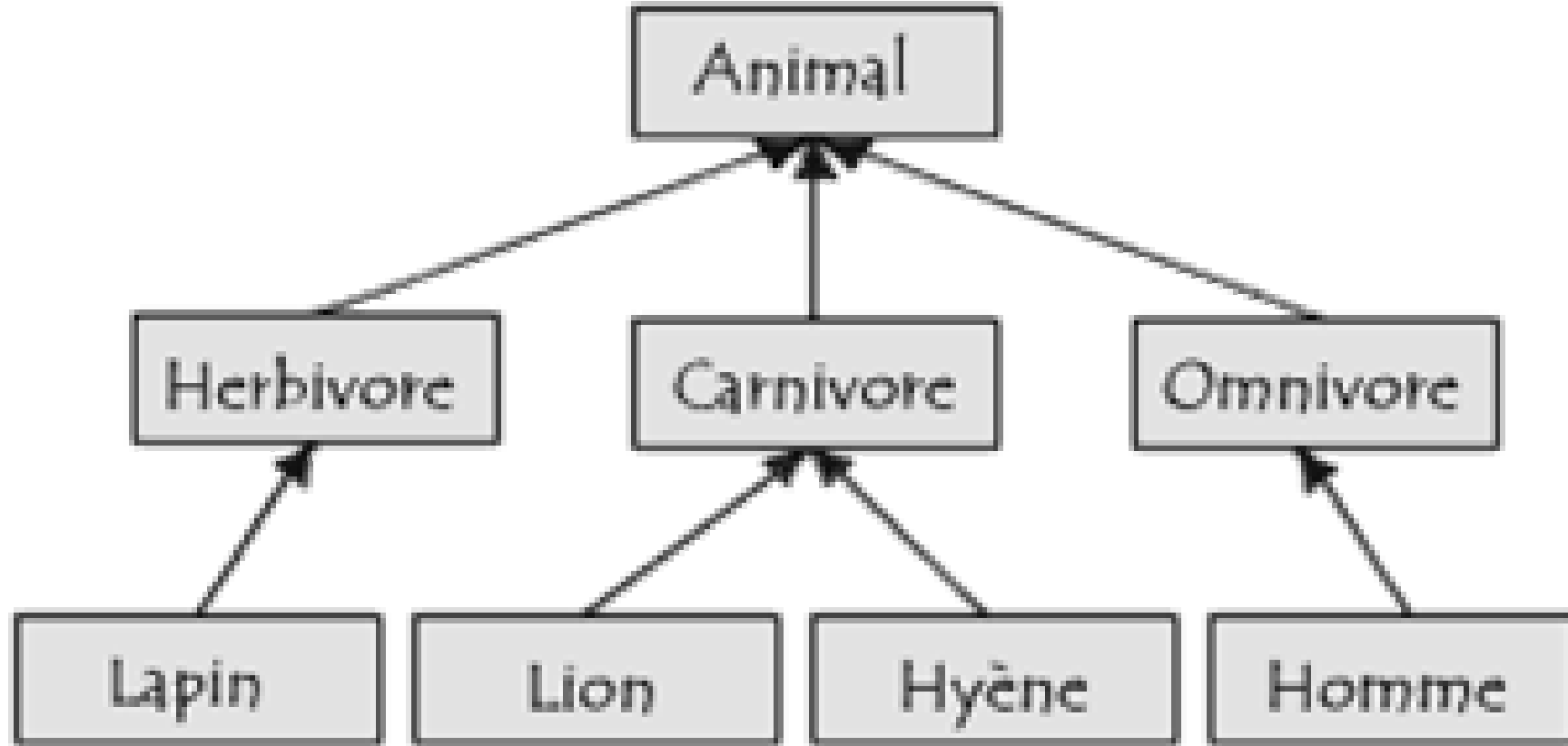
Example :



héritage simple

- ◆ Principe de l'héritage simple
- ◆ onception

Exemple :



héritage multiple

◆ 2 mises en œuvres différentes de l'héritage multiple

↗ Classes

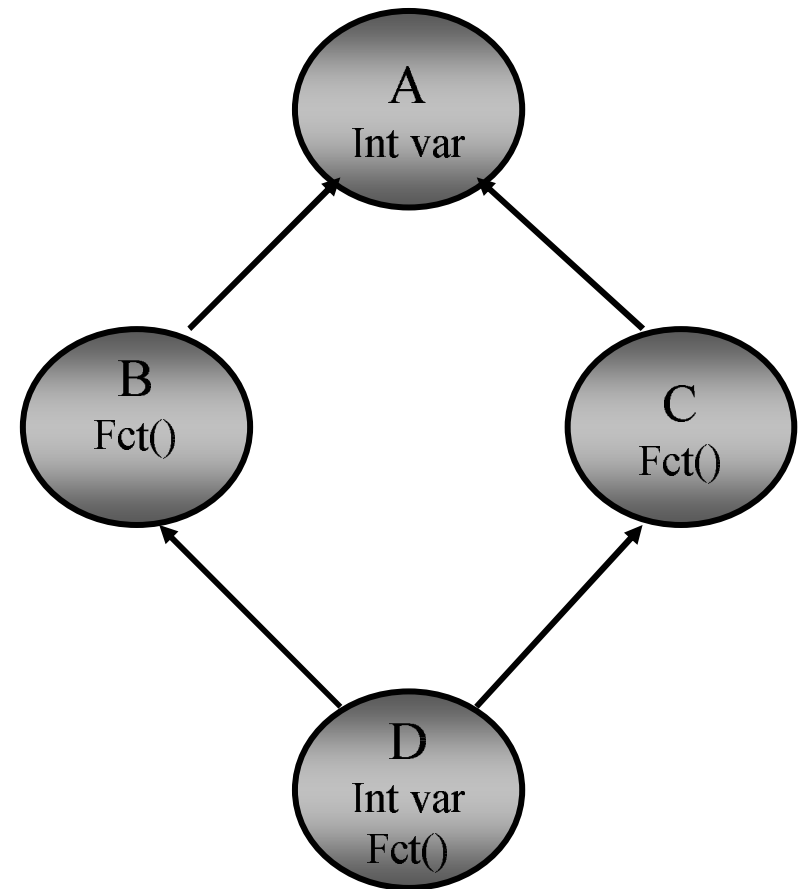
↗ Interfaces

héritage multiple

- ◆ Héritage multiple de classe
 - En C++
 - Implémentation complète de l'héritage

Problème du diamant

```
Void main(){  
D un,deux;  
Deux = new C;  
Un.var; // ambiguïté  
Un.B::var = 1; //OK  
Deux->Fct(); // ambiguïté  
Deux->C::Fct() ; //OK
```



héritage multiple

◆ Héritage multiple de classe

↗ En C++

→ Comment éviter le conflit de nom ??

→ L'héritage virtuel .

→ Mise en place du polymorphisme

```
virtual void  
    nom_méthode()
```

```
Class A{  
    public virtual m(){...}  
};  
Class B : virtual public A  
//héritage simple de A  
Class C : public A  
//héritage simple de A  
//héritage simple de A  
Class D : public B , public C{  
    //héritage multiple de B et C  
//héritage simple de A  
} //héritage multiple de B et C  
//héritage mutiple de A  
}
```

héritage multiple

◆ Héritage multiple de classes

↗ En Eiffel

→ Flexibilité de l'héritage multiple.

→ Différentes clauses d'adaptation

rename permet de modifier les membres hérités.

export permet de changer le mode de dérivation des membres hérités.

undefine pour résoudre les conflits de noms lors d'une transmission multiple

redefine pour redéfinir le membre hérité.

select pour déterminer la méthode à appliquer quand deux méthodes de même nom sont héritées.

héritage multiple

- ◆ Héritage multiple de classes

- ↗ En Eiffel

- Gestion de l'héritage répété

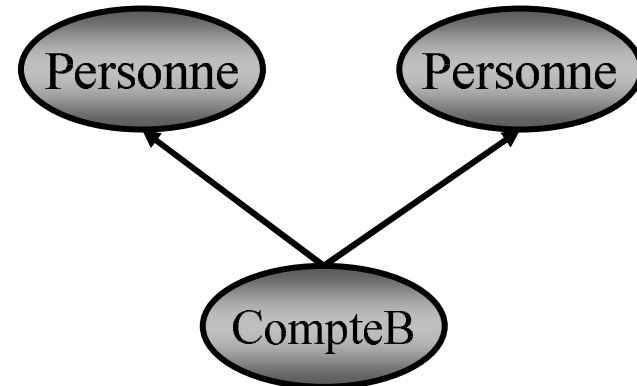
- On peut hériter plusieurs fois d'une même classe.

- Pour éviter les conflits de nom, il suffit de renommer les membres dupliqués avec *rename*.

- Le polymorphisme

- On peut redéfinir les méthodes hérités par la clause *redefine*

Deux personnes possèdent un compte bancaire en commun.



héritage multiple

◆ « Héritage multiple » en Java.

↗ Sens des Interfaces

↗ Contenu des Interfaces

Exemple :

- ◆ Déclaration de constante :

```
interface I {  
    liste constante ;  
    ...  
}
```

- ◆ Déclaration de méthode :

```
interface I {  
    void Affiche();  
    ....  
}
```

Héritage multiple

- ◆ « Héritage multiple » en Java

- ↗ But des interfaces

- ↗ Déclaration

- ↗ Implémentation

Exemple :

```
interface I {
    void Affiche();
    ...
    int Calcul(int a , int b);
}
class Test implements I{
    ....
    void Affiche(){
        System.out.println(« ok »);
    }
    int Calcul(int a , int b){
        return (Math.sqr(a) + Math.sqrt(b));
    }
}
}
```

héritage multiple

- ◆ « Héritage multiple » en C#
 - même idée que le langage Java :
éviter les collisions sur le nom des méthodes
 - les méthodes sont privées et peuvent être consultées si l'objet est casté au type exigé.
 - syntaxe conventionnelle

Conclusions

- ◆ On a regardé les principales différences entre Java , C++, C# et Eiffel.
- ◆ Lequel est le meilleur langage du point de vue de l'héritage ?

FIM