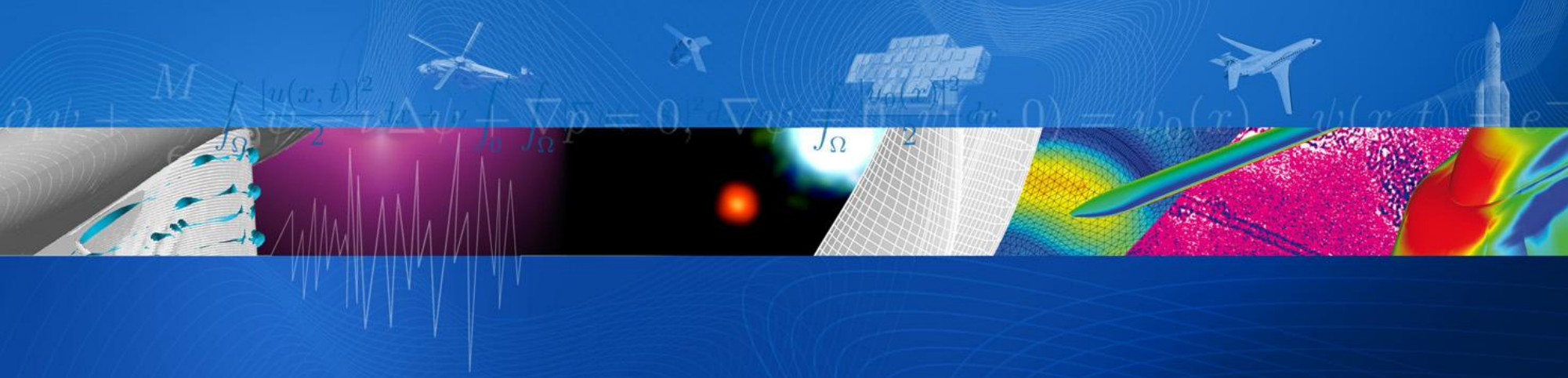


Introduction à la programmation C++

Les structures

BOULCH Alexandre

ONERA
THE FRENCH AEROSPACE LAB

retour sur innovation

Plan de la séance

Conseils

- **Indenter** : rendre le code lisible, vérifier les ouvertures/fermetures des accolades
- **Débugguer** : vérifier les valeurs prises avec les variables
- **Lire les messages d'erreur** : toujours retourner à la première erreur
- **Supprimer les warnings** : supprimer des sources de bugs

Erreurs

```
if(a=3){...}  // une expression est vraie si elle est differente de 0
if(a==3){...}
```

```
for(int i=0, i<10, i++){...}  // ERREUR: pas de ,
for(int i=0; i<10; i++){...}
```

Erreurs avec les tableaux

```
if(a && for(int i=0; i<100; i++){tab[i]}){...} // ERREUR
```

```
bool test=true;
for(int i=0; i<100; i++){
    if(! tab[i])
        test=false;
}
if(a && test){...}
```

Erreurs avec les tableaux

```
void f(double tab[8]){...} // argument : un tableau

void g(){
    ...
    double vec[8]; // tableau de 8 cases
```

```
    f(vec[8]); // ERREUR : vec[nombre] appel le contenu d'une case (un nombre)
               // et en plus, la case 8 n'existe pas
```

```
    f(vec); // OK on appelle la fonction sur la variable vec (un tableau)

}
```

Plan de la séance

Le besoin

Pour l'instant...

- factoriser le code : **les fonctions**
- regrouper les variables de même type et homogènes (même grandeur physique) : **les tableaux**

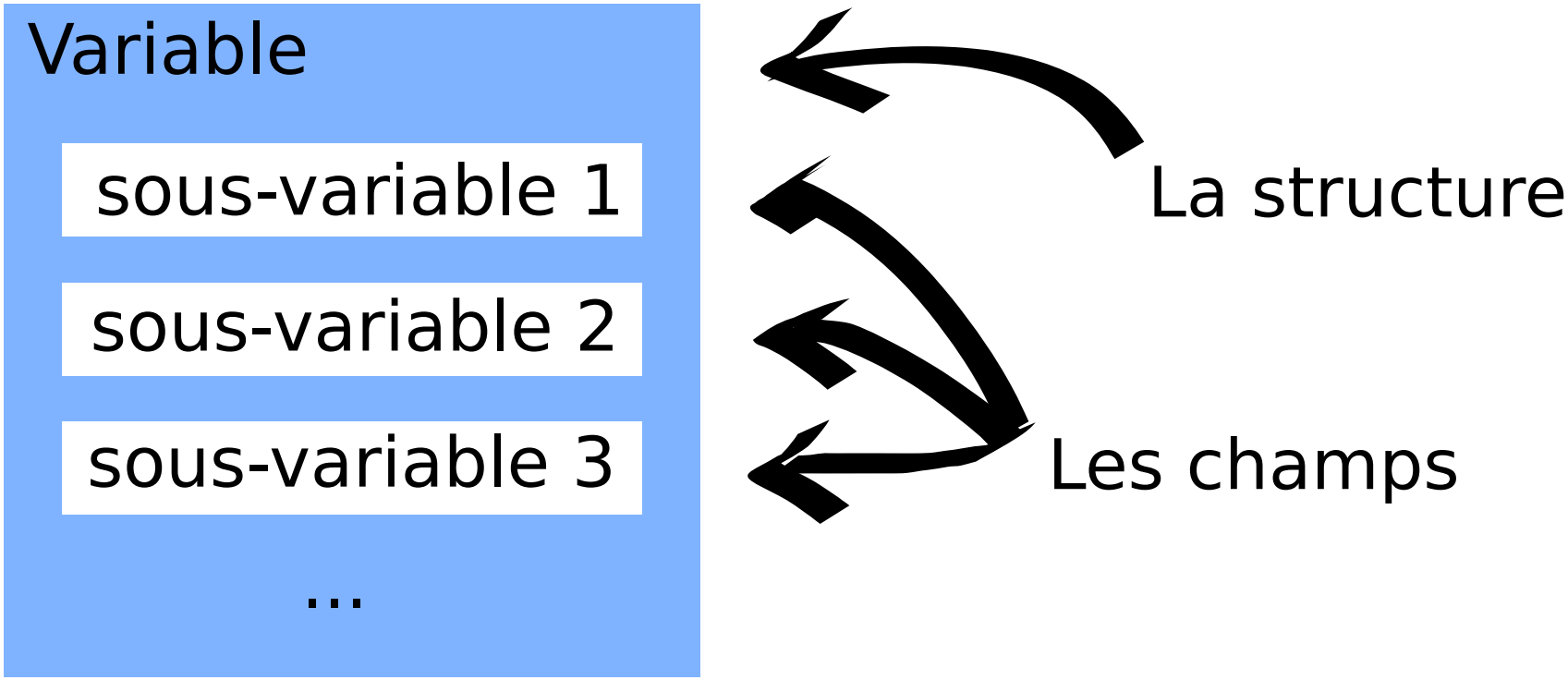
Et maintenant...

Regrouper des variables qui ne sont pas forcément du même type mais qui sont cohérentes :

- Contacts : nom, date de naissance, adresse...
- Dessin : forme, couleur, epaisseur du trait...
- ...

On utilise des **structures**.

Concept



Les structures définissent de nouveaux types.
Les éléments de la structure sont appelés des champs.

Plan de la séance

Définition

Cas général

```
struct nom_structure{
  type1 var1;  // les champs
  type2 var2;
  ...
};  // À ne pas oublier
```

La structure définit un nouveau type qui s'utilise comme les autres.
On accède aux champs avec un .

Cas général

```
variable_struct.var1 = ...
cout « variable_struct.var2 « endl;
```

Tableaux

Tableaux dans les structures

Très, très fortement DÉCONSEILLÉ : problèmes liés à l'égalité entre tableaux, notamment dans les retours de fonctions.

Tableaux de structures

Aucun problème, ils se comportent comme des variables classiques.

Exemple

```
struct Client{
    string nom, prenom;
    string adresse;
    int naissance_j; // Jour
    int naissance_m; // Mois
    int naissance_a; // Annee
    double taille;
    double poids;
    bool lunettes;
};

...
Client cli1;
cli1.nom = "Presley"
cli1.prenom = "Elvis"
cli1.naissance_j = 8;
cli1.naissance_m = 1;
cli1.naissance_a = 1935;
...
```

```
struct Point{
        double x,y;
};

struct Cercle{
    Point centre;
    double rayon;
    Color couleur;
};

...
Cercle c;
c.centre.x = 0.5;
c.couleur = RED;
Point pt;
pt.x = pt.y = 5.5;
c.centre = pt;

Point p1={1,2}, p2;
p2 = p1 // OK, recopie champ a champ
```

Plan de la séance

Initialisation

```
Point pt;
pt.x = pt.y = 5.5;

Cercle c;
c.couleur = RED;
c.rayon = 3;
c.centre.x = 5.5;
c.centre.y = 5.5
//ou
c.centre = pt;
```

```
Point pt = {5.5, 5.5};
Cercle c = {pt, 3, RED};
```

```
Cercle c={{5.5,5.5},3,RED};
```

L'ordre des éléments pour l'initialisation est l'ordre de définition des champs dans la structure.

Attention

```
Cercle c;
c = {{5.5,5.5},3,RED}; // ERREUR
```

Structures et fonctions

Les structures fonctionnent comme des types classiques :

- Argument

```
void affiche(Point p){
    cout << p.x << " " << p.y << endl;
}
```

- Retour

```
Point milieu(Point p, Point q){
    Point m;
    m.x = (p.x+q.x)/2;
    m.y = (p.y+q.y)/2;
    return m;
}
```

Structures et fonctions

Les structures fonctionnent comme des types classiques :

- Passage par référence

```
void init(Point &p){
    p.x = 0;
    p.y = 0;
}
```

Plan de la séance

TP

- Indenter
- Commenter
- Compiler ! ! ! !
- Rendre dans les temps
- Mettre les noms en commentaires dans Educnet

TP

- Simulation de système planétaire
- Système physique
- Utilisation des structures

N'oubliez pas

INDENTEZ le code ! !