Mémo de C++ sous ROOT

- Les blocs d'instructions sont compris entre des accolades **{}**
- Les lignes peuvent avoir n'importe quelle longueur
- Les instructions se terminent par **;**
- Les minuscules et MAJUSCULES sont importantes

LaMeme ≠ laMeme ≠ lameme ≠ Lameme ≠ LAMEME

- Toutes les variables doivent être déclarées, mais on n'est pas obligé de le faire au début de la routine
- On peut initialiser une variable au moment où on la déclare

```
double MonGenou = 8.5;
```

Mémo de C++ sous ROOT (suite)

- Les variables peuvent être:
  - simples :

`int`	`double`	`char`	`float`	`short int`
*(f77)integer*4*	*real*8*	*character*	*real*4*	*integer*2*

  - complexes:
    - association de variables (*structure*)

```
struct maison{int couleur; float nombre_etage;
float longueur; float largeur;}
```

    - association de variables et routines de manipulation de ces variables (*classe*)

```
class maison{int couleur; float nombre_etage;
 float longueur; float largeur;
  mettre_couleur();quelle_couleur();calcule_surface();}
```

  - tableaux:

```
int h[10];double matrice[3][5];
maison quartier[20];
```

Mémo de C++ sous ROOT (suite)

- Bouclez les!

Equivalent Fortran

```
for(int i=0;i<10;i++) {}        do i=0,9 ... enddo
while (i != 10) {}              do while(i.ne.10) ... enddo
do {} while (k<=300)
```

- C'est logique!

==	.eq.	<	.lt.	||	.or.
!=	.ne.	<=	.le.	&&	.and.
!	.not.	>	.gt.	0	.FALSE.
>=	.ge.	≠0	.TRUE.		

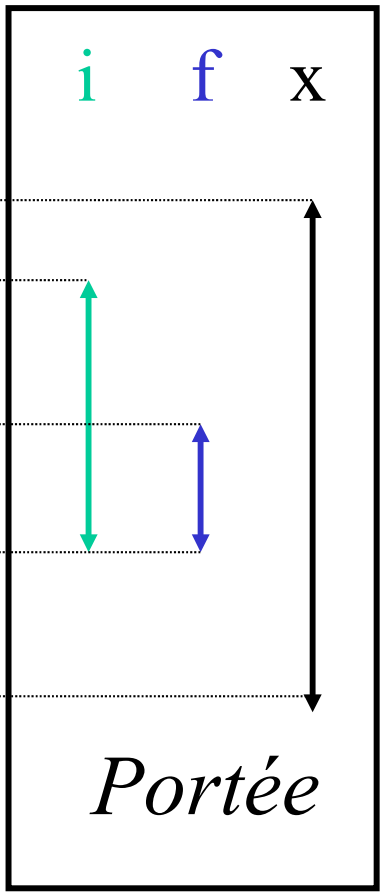
- Avec des si...

```
if(i<10) {} else {}    if(i.lt.10) then ... else ... endif
```

Mémo de C++ sous ROOT (suite)

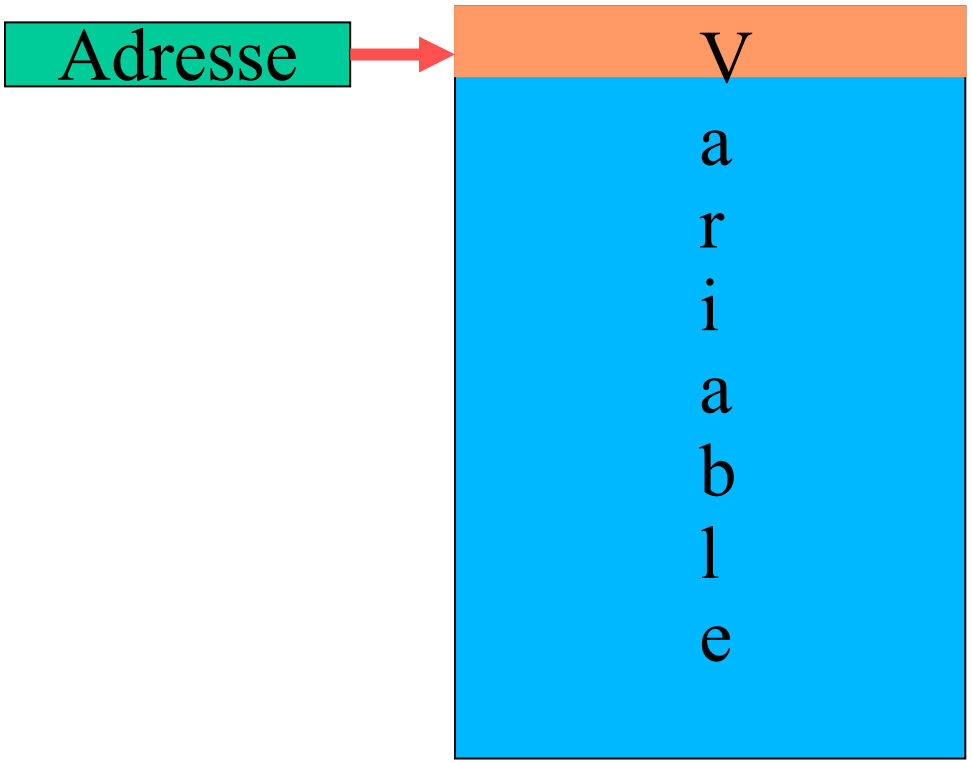
- Les variables n'existent que dans le bloc ou elles sont déclarées

```
{
double x=3;
for(int i=0;i <10;i++)
 {
 double f=pow(x,i/2.);
 cout << x << "**" << i << "=" << f <<  endl;
 }
cout <<  "c'est fini!" << endl;
}
```



Mémo de C++ sous ROOT (suite)

- On peut accéder une variable soit directement, soit par son adresse




```
Adresse  = &Variable

Variable = *Adresse
```

```
maison SweetHome();
SweetHome.quelle_couleur();
SweetHome.longueur;

maison* addSH=&SweetHome;
addSH->quelle_couleur();

maison* pMaison=new maison();
pMaison->quelle_couleur();
pMaison->longueur;
```

Mémo de C++ sous ROOT (suite)

- Passage des arguments à une routine

```
void toto1(double a)
    {
    a=3;
    }
void toto2(double *a)
    {
    (*a)=15;
    }
void test_toto(void)
 {
 double x=8;
 toto1(x);
 cout << "X=" << x << endl;
 toto2(&x);
 cout << "X=" << x << endl;
 }
```

Lors de l'appel, l'argument est copié dans **a** qui est local à **toto1**.

Lors de l'appel, l'adresse de l'argument est copié dans **a**.

x n'est pas modifié

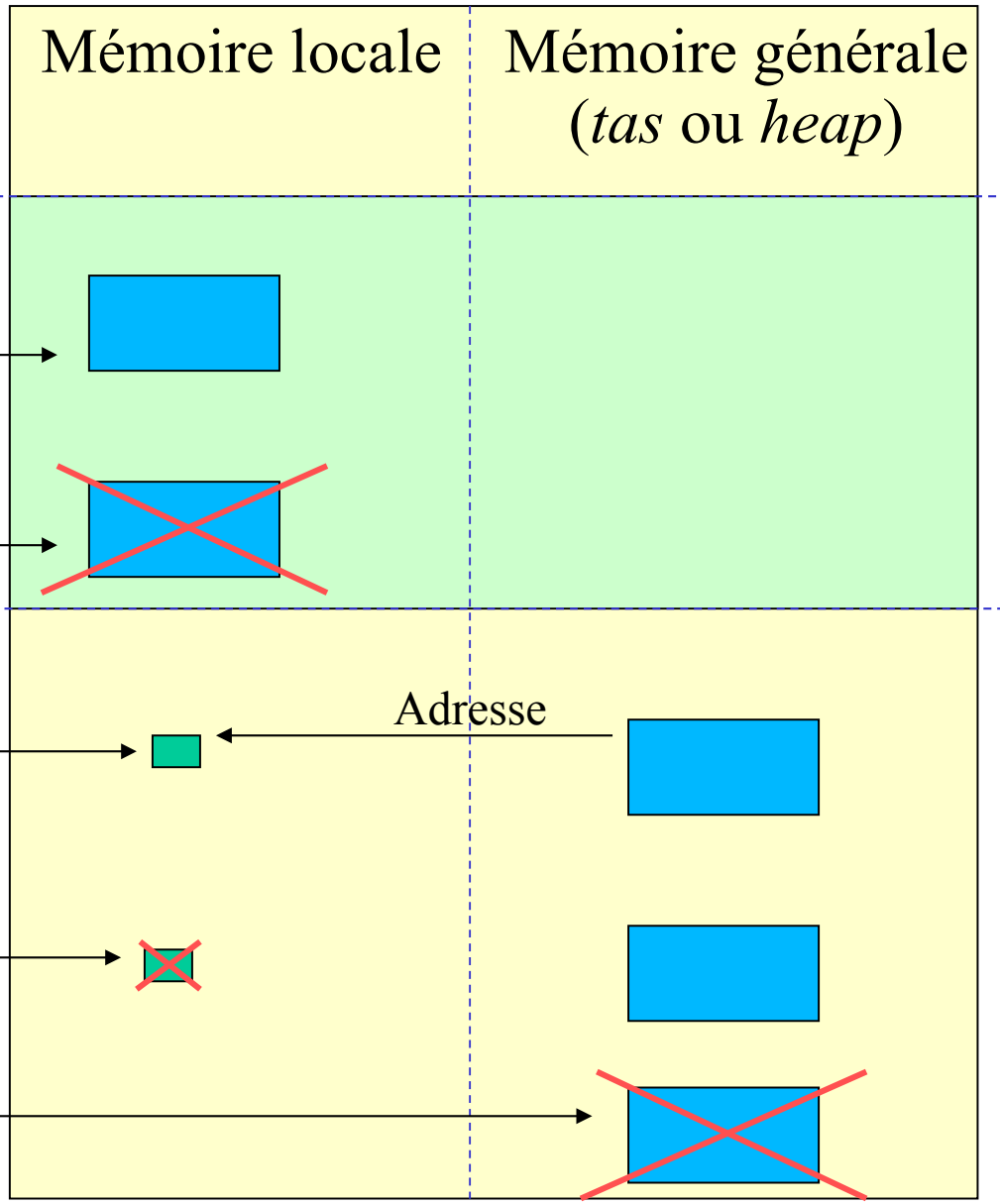
x est modifié

Mémo de C++ sous ROOT (suite)

- Attention à la portée!

```
void toto1(void)
    {
    maison ChezMoi();
    ChezMoi.mettre_couleur(rouge);
    …
    …
    }

void toto2(void)
    {
    maison *ChezMoi=new maison();
    ChezMoi->mettre_couleur(rouge);
    …
    …
    }
…
…
delete ChezMoi;
```



Mémo de C++ sous ROOT (Fin?)

- Particularités de ROOT
  - les classes ROOT commencent par **T** : `TVector`, `TH1F`, `TLine`
  - les constantes ROOT commencent par **k** : `kRed`, `kTRUE`
  - les types de base sont redéfinis, commencent par une majuscule et finissent par **"_t"** : `Double_t, Int_t`
  - on peut retrouver les variables et les actions possibles sur une classe :
    - en utilisant ".class" : `.class TLine`
    - en utilisant la touche <TAB> :

```
TLine l(0,0,1,1)
l.Set<TAB>
```

    - en utilisant l'action (méthode) **DrawClass()** :

```
l.DrawClass()
```

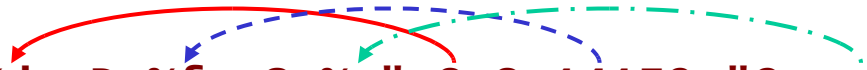
    - sur le WEB : **http://www.lpc-caen.in2p3.fr/root**

Générer les chaines de caractères

Deux solutions:

- Utilisation de sprintf (fonction C/C++: #include "Riostream.h")

```
Char_t chaine[80];
sprintf(chaine,"I=%d, D=%f, S=%s",2,3.14159,"Coucou!");
(Char_t* 0x3462fd0)"I=2, D=3.141590, S=Coucou!"
```



- Utilisation de Form (fonction ROOT: #include "TString.h")

```
Form("I=%03d, D=%6.3f, S=%-10s",2,3.14159,"Coucou!")
(char* 0x22f4249)"I=002, D= 3.142, S=   Coucou!"
```

- Explication des formats dans:

http://www.cplusplus.com/ref/cstdio/printf.html