Algorithmique
&
Langage C
IUT GEII S1

2 Notes de cours

cours_algo_lgc2.08.odp

deuxième partie : Bases Algorithmiques

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Licence

creative commons
COMMONS DEED

•Paternité - Pas d'Utilisation Commerciale -
• Partage des Conditions Initiales à l'Identique 2.0 France

•Vous êtes libres :
• * de reproduire, distribuer et communiquer cette création au public
• * de modifier cette création, selon les conditions suivantes :
•
•**Paternité**. Vous devez citer le nom de l'auteur original.
•
•**Pas d'Utilisation Commerciale**.
•Vous n'avez pas le droit d'utiliser cette création à des fins commerciales.
•
•**Partage des Conditions Initiales à l'Identique**.
•Si vous modifiez, transformez ou adaptez cette création,
•vous n'avez le droit de distribuer la création qui en résulte
•que sous un contrat identiqueà celui-ci.
• * A chaque réutilisation ou distribution, vous devez faire apparaître clairement aux autres les
• conditions contractuelles de mise à disposition de cette création.
•* Chacune de ces conditions peut être levée si vous obtenez l'autorisation du titulaire des droits
•Ce qui précède n'affecte en rien vos droits en tant qu'utilisateur (exceptions au droit d'auteur :
•copies réservées à l'usage privé du copiste, courtes citations, parodie...)
•voir le contrat complet sous : http://fr.creativecommons.org/contrats.htm
•

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Plan du cours

- Opérateurs relationnels
- Opérateurs logiques
- Expressions avec les opérateurs relationnels et logiques

Bases Algorithmiques

- Haut -> Bas
- Alternative : if ... else

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Vrai / Faux

VRAI : ≠ 0

FAUX: 0

En C toute expression non nulle est dite VRAIE

La valeur 0 est dite FAUSSE

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Opérateurs relationnels

Math	C
=	==
≠	!=
>	>
≥	>=
<	<
≤	<=

Une expression formée avec ces opérateurs vaut :

0 si elle est fausse
1 si elle est vraie

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Opérateurs relationnels

```
int i = 2 , j = 3 , k ;
```

k = (i==j) ; k vaut :

k = i >0 ; k vaut :

k = ((i+1) == 3) ; k vaut :

k = (i >= j) ; k vaut :

k = 2 * (j > i) + ((i+j) == (2*i + 1)) + (i==j) ; k vaut :

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Opérateurs relationnels

Ils s'appliquent à la plupart des types.

On peut mélanger certains types dans les expressions.

Ils retournent toujours une valeur entière (0 ou 1).

(Attention à ne pas comparer des valeurs signées
à des valeurs non signées.)

float x = 3.7 ; int i= 2 , j ;

j = (i < x) ;

Attention à cause de la précision limité, l'égalité de deux réels est rarement vraie.

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Opérateurs logiques

ET	&&
OU	||
NON	!

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Opérateurs logiques

```
int i=3, j=4, k ;
k = (i<6) && (j>=i) ;          k vaut :
k = !( (i>2) || (j < 3 ) ) ;   k  vaut :
```

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Bases Algorithmiques

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Haut / Bas

Les instructions s'exécutent de haut en bas,
les unes après les autres.

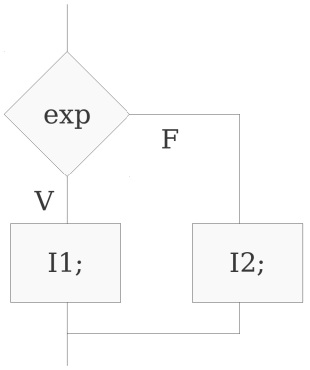
Dès qu'une instruction est exécutée, la suivante commence.

La vitesse est déterminée par les performances de l'ordinateur.

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Alternative (ou choix)

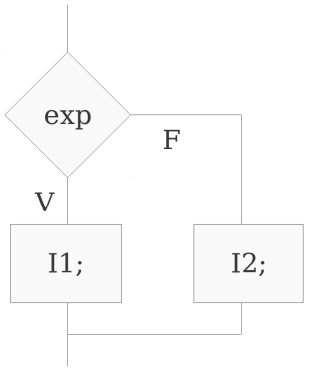
Permet d'exécuter une séquence d'instructions ou une autre selon la valeur vraie ou fausse d'une expression.



Si exp est vraie la séquence I1; est exécutée.
Sinon (si exp est fausse) c'est la séquence I2; qui est exécutée.

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Alternative C : if...else




```
if ( exp )
  {
    I1;
  }
else
  {
    I2 ;
  }
```

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Exemple : if ... else

```
int i=3 , j = 5 , k ;

k = ( i>j ) ;

if ( k )
{
  printf ("A") ;
}
else
{
printf ("B") ;
}
```

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Exemple : if ... else

```
int i=3 , j = 5 ;

if ( i > j )
{
  printf ("A") ;
}
else
{
printf ("B") ;
}
```


© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

if sans else


```
if ( exp )
{
     I ;
}
```


© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Partage en trois

On imbrique des instructions if/else.

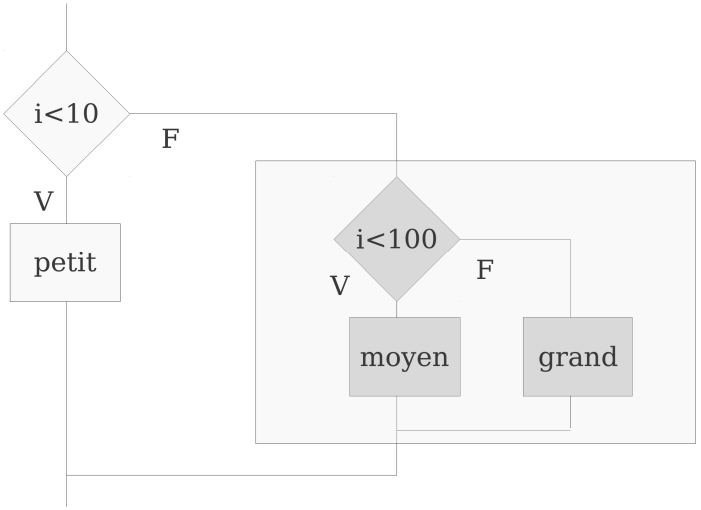
Exemple :

Afficher petit, moyen ou grand en fonction de la valeur de x

Un if/else permet de séparer deux zones.
Par exemple : petit et moyen/grand.
Dans la seconde zone, une autre if/else va permettre de séparer moyen et grand.

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Partage en trois



© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Partage en trois

```
if ( i < 10 )
{
   printf( "petit" );
}
else
{
   if ( i < 100 )
   {
      printf( "moyen" );
   }
   else
   {
      printf( "grand" );
   }
}
```

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Encadrements

Exemple : Afficher ok si x est compris entre -3 et 12 :

```
if ( ( -3 < x ) && ( x <12 ) )
  {
     printf( "ok" );
  }
```

Attention en C l'expression -3<x<12 n'est pas vraie si x est compris entre -3 et 12 !

(-3<x<12 est une expression toujours vraie !
En effet : (-3<x) vaut 0 ou 1 . Mais 0<12 et 1<12 donc -3<x<12 vaut 1 quelque soit x !
De même x<12 vaut 0 ou 1 or -3<1 et -3<0 donc -3<x<12 vaut 1 quelque soit x !)

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Factorisation

Lorsqu'une instruction (ou une partie d'instruction) est identique dans la clause if et dans la clause else,
c'est qu'elle ne dépend pas de l'expression évaluée dans le if().

Elle peut donc être placée soit avant soit après le if/else :
C'est la *factorisation*.

Essayer de factoriser vos programmes.

Factorisation => Programmes souvent plus simples
Programmes plus concis
Programmes plus faciles à modifier

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Factorisation

Non factorisé :

```
if ( x>0 )
{
   x = x+3 ;
   printf("x vaut %d",x);
}
else
{
   x = -x+3 ;
   printf("x vaut %d",x);
}
```

Première factorisation :

```
if ( x>0 )
{
   x = x+3 ;
 }
else
{
   x = -x+3 ;
}
printf("x vaut %d",x);
```

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Factorisation

deuxième factorisation :

```
if ( x < 0 )
{
   x = - x ;
 }
x = x + 3 ;
printf("x vaut %d",x);
```

Si on doit modifier ce programme pour modifier la valeur d'incrémentation ou modifier la langue d'affichage, il n'y a à chaque fois qu'une seule ligne à modifier.

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Ne pas remplacer if/else par deux if !

On est parfois tenté de remplacer un if/else par deux if successifs.

Outre le fait que ce n'est pas efficace car l'expression doit être évaluée deux fois, c'est parfois *dangereux*.

En effet :

```
if ( exp ) { I1; }
     else  { I2; }
```

n'est pas toujours équivalent à :

```
if ( exp ) { I1; }
if (!exp)  { I2; }
```

Le contraire d'une expression n'ait pas toujours trivial. Laissez le compilateur évaluer !exp !!

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Ne pas remplacer if/else par deux if !

Soit le problème suivant : Une valeur x doit être multipliée par 2 si elle est inférieure ou égale à 15 et doit être multipliée par 4 si elle est strictement supérieure à 15.

```
if ( x <= 15 )
{
  x = x * 2 ;
}
if ( x > 15 )
{
  x = x * 4 ;
}
```

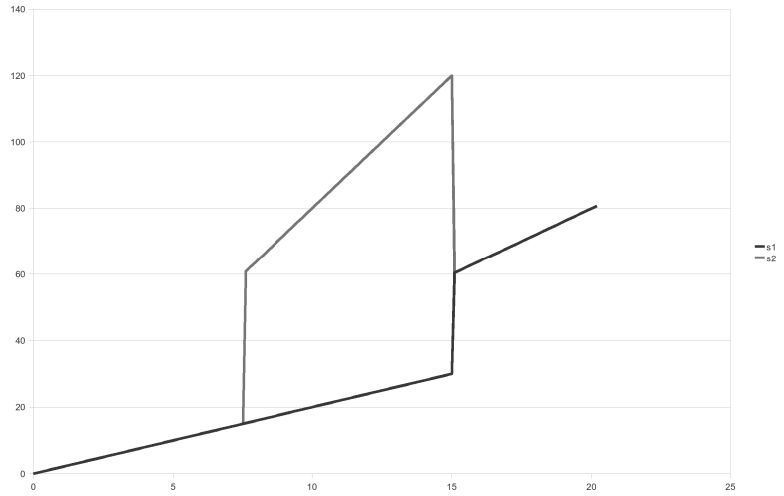
Programme faux

Programme correct :

```
if ( x <=15 )
{
   x = x * 2 ;
}
else
{
  x = x * 4 ;
}
```

© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license

Ne pas remplacer if/else par deux if !



© Copyright 2005, Philippe Arlotto http://arlotto.univ-tln.fr
Creative Commons Attribution-ShareAlike 2.0 license