المدرسة العليا للتجارة
بصفاقس

Note de cours 3

Opérateurs et expressions - Structures conditionnelles

Résumé

1. Expressions

- **Une expression** est un calcul donnant une valeur résultat qui comporte :
  - des variables ou des constantes
  - des opérateurs

- Expression est une partie d'une Instruction

```
a = b + 2 * c ;
```

expression → valeur : résultat du calcul

-
```
z = (x >= y);
```

valeurs: 1 *(vrai) ou* 0 *(faux)*
→ *z* = 1 *si x >= y*
0 *sinon.*

2. Opérateurs

comparaison	Logique	Arithmétique		
		calcul	Assignation	Incrémentation
< *(inférieur strictement)*	**&&** *(et)*	**+** *(somme)*	**+ =** *(ajouter une quantité à la valeur initiale)*	**+ +** *(incrémenter : ajouter 1)*
< = *(inférieur ou égal)*	**| |** *(ou)*	**-** *(différence)*	**- =** *(retrancher une quantité à la valeur initiale)*	**--** *(décrémenter : retrancher 1)*
> *(supérieur strictement)*	**!** *(non)*	***** *(produit)*	*** =** *(multiplier la valeur initiale par une quantité)*	
> = *(supérieur ou égal)*		**/** *(résultat division)*	**/ =** *(diviser la valeur initiale par une quantité : résultat)*	
= = *(égal)*		**%** *(reste division)*	**% =** *(diviser la valeur initiale par une quantité : reste)*	
!= *(différent)*		**=** *(affectation)*		

2.1. Opérateurs logiques et de comparaison

comparaison	Logique
< *(inférieur strictement)*	**&&** *(et)*
<= *(inférieur ou égal)*	**||** *(ou)*
> *(supérieur strictement)*	**!** *(non)*
>= *(supérieur ou égal)*	
== *(égal)*	
!= *(différent)*	

- Écrire des conditions :
  - `(a < b)`
  - `(a*2 == b)`
  - `(a != b)`

Les parenthèses sont obligatoires dans certaines structures

- Écrire des conditions composées
  - `((a < b)&&(b%2 == 0))`
  - `((a < b)||(a+b <= 10))`
  - `!(a*2 < b)`
- Une expression-condition vaut **1** si elle est satisfaite et **0** sinon

```
x = ((7 >= y)&&(y%3==0));
```

Pour y = 6, valeur de x ?
Pour y = 7 ?

2.1. Opérateurs logiques et de comparaison

comparaison	Logique
< *(inférieur strictement)*	**&&** *(et)*
<= *(inférieur ou égal)*	**||** *(ou)*
> *(supérieur strictement)*	**!** *(non)*
>= *(supérieur ou égal)*	
== *(égal)*	
!= *(différent)*	

- Les utiliser dans des **structures conditionnelles** :
  - Exemple 1 :

```
if((7 >= y)&&(y%3==0))
   x = 1;
else x = 0;
```

équivalent à :

```
x = ((7 >= y)&&(y%3==0));
```

  - Exemple 2 :

```
if(y%2==0)
{
   x = 1;
   printf("y est paire\n");
}
else x = 100;
```

2.2. Opérateurs arithmétiques

Calcul
+ *(somme)*
- *(différence)*
* *(produit)*
/
%
= *(affectation)*

- Calcul arithmétique simple
- Division euclidienne
  - `7 / 2` → `valeur (résultat) : 3`
  - `7 % 2` → `valeur (reste) : 1`

```
7 | 2
1 | 3
```

- « % » utilisable avec des entiers seulement (entier % entier)
- « / » utilisable avec des réels ou des entiers

2.3. Opérateurs arithmétiques

Assignation	
+ = *(ajouter une quantité à la valeur initiale)*	○ `x += a;` ⇔ `x = x + a;`
- = *(retrancher une quantité à la valeur initiale)*	○ `x -= a;` ⇔ `x = x - a;`
*** =** *(multiplier la valeur initiale par une quantité)*	○ `x *= a;` ⇔ `x = x * a;`
/ = *(diviser la valeur initiale par une quantité : résultat)*	○ `x /= a;` ⇔ `x = x / a;`
% = *(diviser la valeur initiale par une quantité : reste)*	○ `x %= a;` ⇔ `x = x % a;`

2.4. Opérateurs arithmétiques

Incrémentation
++ *(incrémenter : ajouter 1)*
-- *(décrémenter : retrancher 1)*

- `x++;`
- `++x;`

⇔ `x = x + 1;` ⇔ `x += 1;`

- `x--;`
- `--x;`

⇔ `x = x - 1;` ⇔ `x -= 1;`

Différence

x++ : incrémentation de x après la fin du traitement de l'instruction en cours

++x : incrémentation de x avant la fin du traitement de l'instruction en cours

Exemple 1

```
short a=1, b;
b = 10 + a++;
```

➔ **a = 2 et b = 11**

Exemple 2

```
short a=1, b;
b = 10 + ++a;
```

➔ **a = 2 et b = 12**

3. Priorité des opérateurs

++++++++++++	()	[]				
+++++++++++	--	++	!	~	-	
++++++++++	*	/	%			
+++++++++	+	-				
++++++++	<<	>>				
+++++++	<	<=	>=	>		
++++++	==	!=				
+++++	&					
++++	^					
+++	|					
++	&&	||				
+	?:					
	=	+=	-=	*=	/=	%=

4. Structure conditionnelle if… else…

- **if** est associée ou non à **else**
  - **if** (condition) **…**

```
if(a < 10)
   printf("a est un chiffre\n");
```

  - **if** (condition)**… else …**

```
if(a < 10)
   printf("a est un chiffre\n");
else printf("a n'est pas un chiffre\n");
```

4. Structure conditionnelle if… else if…else…

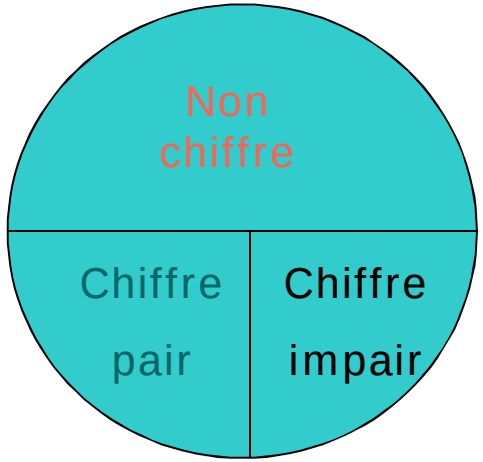
- **Succession de tests (conditions)**

  if (condition_1)…

  else if(condition_2)…

  else if(condition_N)…

  else…




```
if(a >= 10)
   printf("a n'est pas un chiffre\n");
else if(a%2==0)
   printf("a est un chiffre paire\n");
else printf("a est un chiffre impaire\n");
```

4. if… else …
avec un bloc d'instruction à exécuter

→ **Les accolades**

{ … }

- Obligatoires : en cas d'un bloc d'instructions à exécuter successivement ensemble si une condition est vérifiée
- Facultatives : en cas d'une seule instruction

Règle toujours valable quelque soit la structure

```
int x; printf("donner un entier impair : "); scanf("%d",&x);
if(x%2==0)
{
   printf("x est pair\n");
   x++;
   printf("La valeur de x est remplacée par  : %d\n", x);
}
else printf("x est bien impair\n");
```

5. Exemple et instruction (switch…case)

```
char reponse;
printf("Répondre par oui (o) ou non (n) : ");
fflush(stdin); // pour vider le buffer (voir avec le prof de TP)
scanf("%c",&reponse);
if ((reponse == 'o')||( reponse == '0'))
   printf("vous avez répondu par oui\n") ;
else if ((reponse == 'n')||( reponse == 'N'))
   printf("vous avez répondu par non\n") ;
else
printf("mauvaise reponse\n");
```

5. Exemple et instruction (switch…case)

- La structure <switch> est une structure conditionnelle permettant d'exécuter une instruction ou plusieurs suivant la valeur prise par une variable. Elle est utilisée selon la forme syntaxique suivante :

```
switch (<variable>)
{
      case <valeur 1> : <bloc instruction 1>; break;
      case <valeur 2> : <bloc instruction 2>; break;
                .
                .
                .
      case <valeur N> : <bloc instruction N>; break;
      default : <bloc instruction N+1>; break;
}
```

- Si aucune des valeurs proposées ne correspond, alors c'est l'instruction par défaut qui sera exécutée.

5. Exemple et instruction (switch…case)

```
char reponse;
printf("Répondre par oui (o) ou non (n) : ");
reponse = getchar(); getchar();
switch (reponse)
{
        case 'o' : printf("vous avez répondu par oui\n"); break;
        case 'O' : printf("vous avez répondu par oui\n"); break;
        case 'n' : printf("vous avez répondu par non\n"); break;
        case 'N' : printf("vous avez répondu par non\n"); break;
        default : printf("mauvaise reponse\n");
}
```

5. Exemple et instruction (switch…case)


```
char reponse;
printf("Répondre par oui (o) ou non (n) : ");
reponse = getchar(); getchar();
switch (reponse)
{
        case 'o' :
        case 'O' : printf("vous avez répondu par oui\n");
                   /*instructions pour les cas 'o' et 'O'*/  break;
        case 'n' :
        case 'N' : printf("vous avez répondu par non\n"); break;
        default : printf("mauvaise reponse\n");
}
```