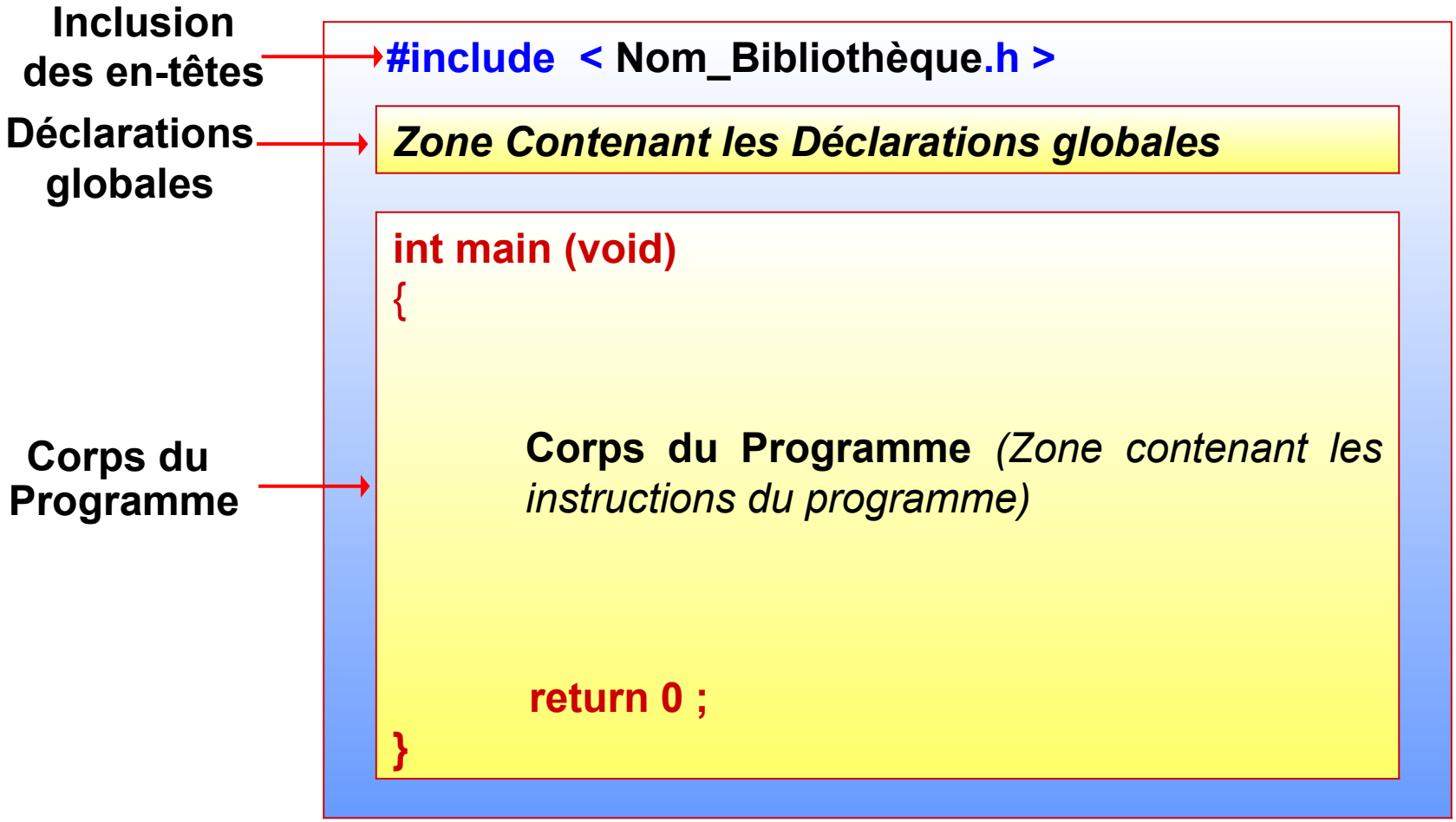
Cours N°2

Règles Générales d'Écriture d'un Programme en C

1. Structure d'un fichier en langage C



1. Structure d'un fichier en langage C (exemple)

```c
/* Programme Affiche nombre */
#include <stdio.h>
int a;
int main(void)
{
    a = -42;
    printf("La valeur du nombre est %d\n",a);
    return 0;
}
```

2. Les Identificateurs

- Pour manipuler différents objets dans un programme, il faut leur donner des noms.
- Les noms utilisés pour les objets manipulés sont des **identificateurs.**

Définition :

L'identificateur est un nom symbolique utilisé pour nommer (identifier) un objet dans un programme informatique.

2. Les Identificateurs (suite)

Les « objets » dans un programme sont des :

- ✓ ***Constantes,***
- ✓ ***Variables,***
- ✓ ***Types,***
- ✓ ***Procédures,***
- ✓ ***Fonctions.***

2. Les Identificateurs (suite)

❑ Règles d'écriture d'un identificateur :

Les identificateurs sont représentés par une suite de lettres et/ou de chiffres avec les ***restrictions*** suivantes :

- ✓ le premier caractère doit être alphabétique, donc une lettre obligatoirement ;
- ✓ les caractères suivant le premier peuvent être numériques ;
- ✓ le caractère souligné « _ » est permis;

2. Les Identificateurs (suite)

❑ Règles d'écriture d'un identificateur :

- ✓ l'utilisation des *lettres accentuées* et de la *cédille* (à ê ù ç …) est interdite ;
- ✓ les caractères dits " **spéciaux** " c'est-à-dire l'**espace** et les **symboles** : parenthèses, signe plus (**+**), signe moins (**-**), signe égal (**=**), point-virgule (**;**) … sont ***interdits*** ;
- ✓ l'utilisation des mots clés (réservés) du langage est interdite ;
- ✓ l'utilisation des minuscules ou des majuscules est permise mais **ATTENTION** le langage C fait la différence entre les minuscules et les majuscules (il est *sensible à la* ***casse***).

2. Les Identificateurs (suite)

- **Règles d'écriture d'un identificateur (suite):**

Exemples:

`ValeurM, Valeur_A, AB, B7, Nom`	***Sont des identificateurs corrects***
`Valeur M, Valeur-A, A/B, 7B, Nom$`	***Sont des identificateurs incorrects***

2. Les Identificateurs (suite)

- **Les mots clés (mots réservés) du langage C:**

auto, enum, restrict, unsigned, break, extern, return, void, case, float, short, volatile, char, for, signed, while, const, goto, sizeof, continue, if, static, default, inline, struct, do, int, switch, double, long, typedef, else, register, union.

Attention ! Un mot clé n'est ***JAMAIS*** accepté comme identificateur.

3. Les Séparateurs

Définition :

Un séparateur est une espace (un blanc), un caractère ou une série de caractères, destinés à séparer des identificateurs.

- En langage C, les différents mots du langage sont séparés soit par une espace, soit par un signe particulier ou une fin de ligne.
- Dans un programme, deux objets successifs doivent être séparés soit par une espace, soit par une fin de ligne. Sinon, le compilateur renvoie un message d'erreur.

3. Les Séparateurs (suite)

- **Exemples de séparateurs :**

Séparateurs	Définition	Exemples
;	fin d'instruction ou fin de déclaration	`int n;` `return 0;`
,	séparateur virgule pour séparer des variables	`int a,b;`

4. Constante

- Une **constante** est un identificateur qui contient une valeur qui ne sera jamais modifiée au cours du programme.
- Les constantes peuvent être:
  - des nombres entiers, exemple: **12**,
  - des nombres réels, exemple: **20.6**,
  - un caractère ('a') ou une chaîne de caractères ("bonjour").
- **Conseil :** par convention, le nom d'une constante doit être écrit en MAJUSCULES pour la différencier des autres identificateurs.
- **Déclaration :**

#define NOM_CONSTANTE valeur_constante

5. Types de données en langage C

Les types de données de base en C sont :

- les **entiers**,
- les **réels** ou **nombres à virgule flottante**,
- les **caractères** et **chaînes de caractères**,
- les **booléens**.

Remarque: Une fois qu'un type est associé à une variable, on ne peut pas lui affecter une valeur qui ne correspondrait pas à son type.

5. Types de données en langage C (suite)

Type de donnée	Signification	Taille (en octets)	Plage de valeurs acceptée
char	Caractère	**1**	-128 à 127
unsigned char	Caractère non signé	**1**	0 à 255
short int	Entier court	**2**	-32 768 à 32 767
unsigned short int	Entier court non signé	**2**	0 à 65 535
int	Entier	**2** (sur CPU 16 bits)	-32 768 à 32 767
		4 (sur CPU 32 bits)	-2 147 483 648 à 2 147 483 647
unsigned int	Entier non signé	**2** (sur CPU 16 bits)	0 à 65 535
		4 (sur CPU 32 bits)	0 à 4 294 967 295

5. Types de données en langage C (suite)

Type de donnée	Signification	Taille (en octets)	Plage de valeurs acceptée
long int	Entier long	**4**	-2 147 483 648 à 2 147 483 647
unsigned long int	Entier long non signé	**4**	0 à 4 294 967 295
float	Flottant (réel)	**4**	$3.4*10^{-38}$ à $3.4*10^{38}$
double	Flottant double	**8**	$1.7*10^{-308}$ à $1.7*10^{+308}$
long double	Flottant double long	**10**	$3.4*10^{-4932}$ à $3.4*10^{4932}$

6. Opérateurs arithmétiques

+	addition	binaires
-	soustraction	
*****	multiplication	
/	division	
%	modulo	
-	nombre négatif	unaire

***unaires* :** opérateurs agissant que sur 1 donnée ;

***binaires* :** opérateurs agissant sur 2 données.

7. Opérateurs relationnels

==	égalité *à ne pas confondre avec l'affectation =*	binaires
!=	différence	
<=	inférieur ou égal	
>=	supérieur ou égal	
>	Strictement supérieur	
<	Strictement inférieur	

8. Opérateurs logiques

&&	et	binaires
||	ou	
!	non	unaire

9. Affectations

signe	utilisation	équivalent
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
***=**	x *= y	x = x * y
/=	x /= y	x = x / y
++	x++ ++x	x = x +1
--	x-- --x	x = x - 1

10.Priorité des Opérateurs

1	**( ) [ ] .**
2	**! ~ ++x --x -x**
3	*** / %**
4	**+ -**
5	**<< >>**
6	**< > <= >=**
7	**== !=**
8	**&**
9	**^**
10	**|**
11	**&&**
12	**||**
13	**? :**
14	**= += -= *= /= %= &= ^= |= <<= >>=**

11.Fonctions Mathématiques

Pour pouvoir utiliser les fonctions de la bibliothèque **math**, il faut inclure le fichier d'en-tête **math.h**

Syntaxe: `#include <math.h>`

Les fonctions les plus courantes sont:

exp pow log log10 sqrt sin cos tan

asin acos atan atan2

12.Fonctions d'entrée/sortie de base

Pour pouvoir utiliser ces fonctions, qui font partie de la bibliothèque **stdio** *(**st**andar**d** **i**nput/**o**utput)*, il faut inclure le fichier d'en-tête **stdio.h**

Syntaxe: **#include <stdio.h>**

Il existe deux fonctions de base pour permettre au programme en langage C d'écrire sur les flux d'entrée et de sortie standard:

1. **printf:** "*print formatted*" permet d'**afficher** des données à l'écran,
2. **scanf:** "*scan formatted*" permet de **lire** à partir du clavier

Syntaxe:

1. printf(chaine de format, valeur1,valeur2,...)
2. scanf(chaine de format, &valeur1, &valeur2,...)

	Type de donnée à afficher	Caractère de formatage
Numériques	Entier décimal signé	d
	Entier décimal non signé	u ou i
	Entier octal non signé	o
	Entier hexadécimal non signé	x (avec les caractères 'a' à 'f') ou X (avec les caractères 'A' à 'F')
	Flottants (réels) et flottants de type double	f, e, g, E ou G
Caractères	Caractère isolé	c
	Chaîne de caractères	s