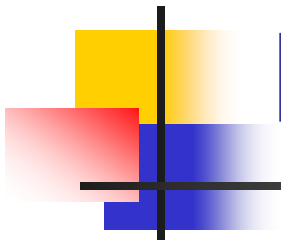
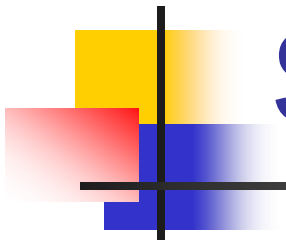


PRINTF

printf ("Texte", expr1, expr2, expr3) ;

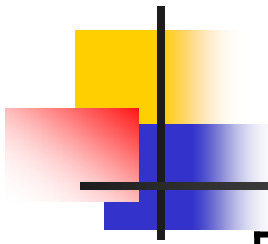
- Ecriture formatée à l'écran.
- Le Texte contient des caractères à imprimer et des formats (%).
- Il y a autant d'arguments (exprn) que de formats dans le **"Texte " .**



Spécification de format

% [drapeaux] [largeur] [.precision] [modificateur] type

- **drapeaux :** il précise la justification et le préfixe.
- **largeur :** nombre de caractères pour le nombre.
- **précision :** pour un réel, le nombre de chiffres après la virgule.
- **modificateur** : modifie le type.



Largeur

Largeur	Effet sur l'affichage
n	Affiche au moins n caractères (avec des espaces à gauche
0n	Affiche au moins n caractères avec des 0 à gauche
*	La largeur est donnée en paramètre

Largeur – exemples 1

```
int val;
val = 333;
printf ("*%5d*\n", val);
*  333*
printf ("*%05d*\n", val);
*00333*
printf ("*%2d*\n", val);
*333*
```

Précision

rien	Entiers : pas de chiffre après la virgule
.n	Au plus n chiffres après la virgule
.*	Nombre de chiffres après la virgule en paramètre

Précision – Exemples 2

```
float nbre;
nbre = 123.456;
printf ("*%12f*\n", nbre);
*  123.456000*
printf ("*%7.2f*\n", nbre);
* 123.46*
printf ("*%07.2f*\n", nbre);
*0123.46*
printf ("*%2.2f*\n", nbre);
*123.46*
```

Modificateur

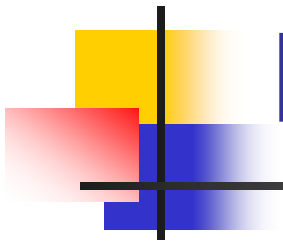
h	short : %hd
l	long : %ld
L	double : %Lf

Types 1

d (ou i)	int en décimal
o	int en octal (base 8)
u	unsigned int
x ou X	int en hexadécimal

Types 2

f	Réel de la forme [-]dddd.ddd
E ou e	Réel de la forme [-]d.ddd e [+/-]ddd
G ou g	Comme f ou e suivant la valeur
c	Caractère
s	Chaîne de caractères



Précision .n

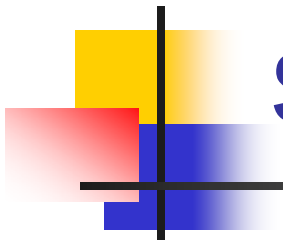
- N : nombre de chiffres après la virgule
- Rien : 6 chiffres après la virgule pour les nombres à virgule flottante (%f)
- .* : précision dynamique

Drapeau

rien	Justifié à droite Complété à gauche par des espaces
-	Justifié à gauche Complété à droite par des espaces
+	Impose le signe même aux valeurs > 0
#	Préfixe o pour octal, 0x pour hexadécimal

Caractères spéciaux

\a	Sonnerie
\n	Retour à la ligne
\b	Retour d'un caractère
\t	Tabulation
\\	\
\ "	"
\%	%



scanf

int scanf ("format", adresseVariable1, adresseVariable2, ...);

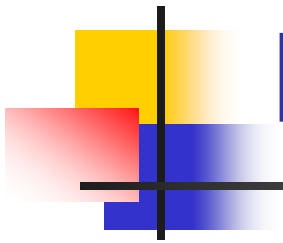
Format contient les formats des variables à initialiser.

Paramètres : adresse des variables à initialiser.

Valeur de retour

```
int n;
int val;
float reel:
n = scanf ("%d %f" , &val, &reel);
```

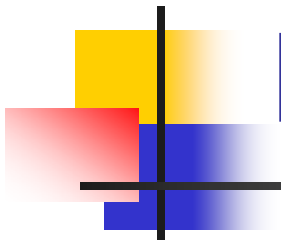
n sera égale au nombre de variables
correctement initialisées (0, 1 ou 2);



Format

% [largeur] [modificateur] type

- **Largeur : nombre de caractères lus**
- **Modificateur :**
  - **h : short,**
  - **l : long**
  - **L : double**
- **Type : idem printf**



Problèmes avec scanf

- Mémoire tampon : caractères saisis au clavier
- scanf lit séquentiellement dans cette mémoire tampon.
- Quand la mémoire tampon est vide :
  - Nouvelle saisie au clavier
- S'il reste quelque chose dans la mémoire tampon : pas de nouvelle saisie.