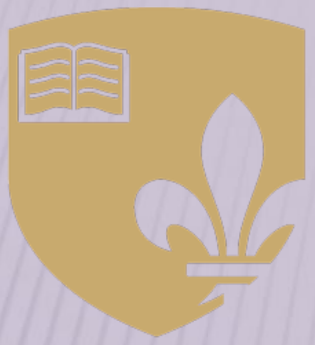

COLLEGE DE PARIS SUPÉRIEUR

DAKARTECH

PROGRAMMATION C

Dr Pape Abdoulaye **BARRO**
Enseignant – Chercheur
Spécialiste en Télémétrie & Systèmes Intelligents

PLAN

- Généralités, Types et Operateurs
- Variables, Lecture/écriture, conditions et expression
- Structures conditionnelles et itératives
- Tableaux
- Sous programmes

GÉNÉRALITÉS, TYPES ET OPERATEURS

PRÉAMBULE

- Créé en 1972 par Denis Ritchie avec un objectif: écrire un système d'exploitation (UNIX).
- Mais ses qualités opérationnelles l'ont très vite fait adopter par une large communauté de programmeurs.
- Une première définition apparue en 1978 avec l'ouvrage de Kernighan et Ritchie The C programming language.
- Son succès international a contribué à sa normalisation, d'abord par l'ANSI (American National Standard Institute), puis par l'ISO (International Standards Organization) puis en1993 par le CEN (Comité Européen de normalisation) et en fin, en 1994, par l'AFNOR.

GÉNÉRALITÉS, TYPES ET OPERATEURS
CRITÈRES

- **Modulaire**: peut être découpé en modules qui peuvent être compilés séparément.
- **Universel**: n'est pas orienté vers un domaine d'application particulier.
- **Typé**: tout objet C doit être déclaré avant d'être utilisé.
- **Portable**: sur n'importe quel système en possession d'un compilateur C.

GÉNÉRALITÉS, TYPES ET OPERATEURS
MOTS CLEFS

- Certains « mots-clés » sont réservés par le langage à un usage bien défini et ne peuvent pas être utilisés comme identificateurs.
- En voici la liste, classée par ordre alphabétique.

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

GÉNÉRALITÉS, TYPES ET OPERATEURS
NOTION DE TYPE

- Un **type** est un nom pour un ensemble de valeurs. Il sert à préciser :
  - La nature des valeurs acceptables
  - Les opérations autorisées sur ces valeurs
  - La taille mémoire utilisée
- Il peut être :
  - **Simple** : entier, réel, booléen, caractère, etc.
  - **Composé**: tableau, énumération,
- **Trois types de base** :caractères, entier relatif et réel

GÉNÉRALITÉS, TYPES ET OPERATEURS
CARACTERE

- Symboles alphanumériques (a,z,!,1,9), caractères spéciaux (retour à la ligne, beep, tabulation, etc..)
- Un **caractère** est représenté sur un octet (8 bits) suivant la table ASCII (American Standard Code for Information Interchange)
- **Table ASCII**
  - code ASCII du 'A' = 65
  - 'a' = 97
  - 'A'<'B'<.....< 'Z'
  - '0'<'1'<'2'<.....<'9'
  - 'a'<'b'<.....<'z'
- **Déclaration**
  - char c;
  - c = 'A'; ou c=65;
- **Constante de type caractère**
  - #define caractere_a 'a'

Caractères spéciaux:

- \a : beep
- \n : retour à la ligne
- \t : tabulation
- \' : apostrophe
- \0 : caractère nul (indique la fin d'une chaîne)
- Etc.

GÉNÉRALITÉS, TYPES ET OPERATEURS

CODE ASCII

Code ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.asciitable.com

GÉNÉRALITÉS, TYPES ET OPERATEURS
ENTIERS NATURELS

- Codage sur 2 ou 4 octets selon le calculateur

$$x = \sum_{i=0}^{n-1} x_i 2^i, x_i \in \{0,1\}, \text{n} = 16 \text{ ou } 32$$

- Sur n bits on peut coder les nombres de 0 à 2^n-1
- Nombre représenté en base 2.
  - Les bits sont rangés dans des cellules correspondant à leur poids. On complète à gauche par des 0
- Exemple (sur deux octets) :
  - 12 = 8+4+0 = $1*2^3+1*2^2+0*2^1+0*2^0$ = 0000000000001100
- Déclaration d'une variable entier naturel x
  - unsigned int x;
  - short unsigned int x; (on force sur 16 bits)
  - long unsigned int x; (on force sur 32 bits)

GÉNÉRALITÉS, TYPES ET OPERATEURS
ENTIERS RELATIFS

- Implantation sur 2 ou 4 octets suivant le compilateur
- Codage complément à 2

$$x = -x_{n-1}2^{n-1} + \sum_{i=0}^{n-2} x_i 2^i, x_i \in \{0,1\}$$

  - Si x_{n-1} = 0 le nombre positif sinon négatif
- Exemple sur 16 bits
  - +5 = $1*2^2+0*2^1+1*2^0$= 0000 0000 0000 0101
  - -3 = $-32768+32765 = -2^{15}+2^{14}+2^{13}+2^{12}+2^{11}+2^{10}+2^9+2^8+2^7+2^6+2^5+2^4+2^3+2^2+0.2^1+1.2^0$
  - = 1111 1111 1111 1101
- Sur 16 bits (2 octets)

  $-32768 \leq x \leq 32767$
- Sur 32 bits

  $-2147483648 \leq x \leq 2147483647$

- plus grand entier positif est $2^{n-1}-1$
- le plus petit entier négatif est -2^{n-1}

- Déclarations

```
int a,b;
a = 1;
b= 3;
```

GÉNÉRALITÉS, TYPES ET OPERATEURS
ENTIERS RELATIFS

Le C prévoit que, sur une machine donnée, on puisse trouver jusqu'à trois tailles différentes d'entiers, désignées par les **mots-clés** suivants :

- short int (qu'on peut abréger en short),
- int (entier relatif),
- long int (qu'on peut abréger en long).

- Chaque taille impose naturellement ses limites. Toutefois, ces dernières dépendent, non seulement du mot-clé considéré, mais également de la machine utilisée: tous les int n'ont pas la même taille sur toutes les machines !
- Fréquemment, deux des trois **mots-clés** correspondent à une même taille (par exemple, sur PC, short et int correspondent à **16 bits**, tandis que long correspond à **32 bits**).

GÉNÉRALITÉS, TYPES ET OPERATEURS
RÉEL

- Implantation sur 4 octets
- Représentation suivant le compilateur et en général sous format mantisse exposant (norme IEEE)
- Codage des réels : virgule flottante, Norme IEEE
- **Déclaration**
  - **float** x,y;
  - x = 3.14;
  - y = 2.0 e+3;
- **Extension :**
  - **double** x; **x** est codé sur 8 octets

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEURS ARITHMÉTIQUES

- Opérateurs bi-opérandes
  - +, -
  - * , / , % (modulo)
- Les opérandes doivent être des valeurs numériques.
  - entier **opérateur** entier -> **entier**
  - réel **opérateur** réel -> **réel**
  - entier **opérateur** réel -> **réel**
- Exemples

```
int a,b;
a=10; b= 3
a+b     13
a-b     7
a*b     30
a/b     3 (division euclidienne)
a%b     1
```

```
float a,b;
a=12.6; b= 3.0
a+b     13.6
a-b     9.6
a*b     37.8
a/b     4.2 (division réelle)
a%b     erreur de syntaxe
```

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEURS ARITHMÉTIQUES

- Opérateurs unaires (un opérande)
  - signe :+, -
    - exemple : a = -a;
  - incrémentation, décrémentation :++ (+1) , -- (-1)
  - exemple :
    - int i=1;
    - ++i; donne 2
  - Syntaxes : ++i ou i++
    - ++i : la valeur de i est d'abord incrémenté, la valeur résultat est utilisée dans l'expression courante
    - i++: la valeur courante de i est utilisée dans l'expression courante, puis i est incrémenté
  - Syntaxes : - -i ou i- - : Même principe mais en décrémentant

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEURS D'AFFECTATION

- **Affectation simple:**
  - **syntaxe :** variable = expression

  la valeur de l'expression est stockée dans la mémoire à l'endroit réservé pour la variable
  - **Exemples :**
    - a = 2; b=1; c=0;
    - a = b+c;
    - a = b && c;

    la valeur de l'expression vaut la valeur affectée
- **Affectation et opération** :+=, -=, *=, /=, %=,<<= , >>=, &=, |=, ^=
  - **Syntaxe :** variable **opérateur** expression

  équivalent à : variable = variable **opérateur** expression
  - **Exemple :**
    - int i;
    - i+=2;

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEURS LOGIQUES

- **Valeur logique :**
  - 0 : faux
  - ≠ 0: vrai
- **Relationnels :** >= , > , == , <, <= , !=
  - La valeur de l'expression est 1 si l'expression est vraie , 0 sinon
  - **Exemple :** 2 < 3 vaut 1 alors que 2 > 3 vaut 0
- **Logiques :** && "et", || "ou", ! "non"
  - Dans l'évaluation de l'expression, 0 =>"faux", toute valeur ≠ 0=>"vraie"
  - La valeur de l'expression est 1 ou 0
  - **Exemples:**
    - 2 && 0 vaut 0 et donc est faux
    - 2 || 0 vaut 1 et donc est vrai
    - !0 vaut 1
    - !4 vaut 0

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEURS BIT À BIT

- Opèrent sur les représentations des valeurs
- & et , | ou, ^ ou-exclusif, ~ complément à 1 ,
- << décalage à gauche, >> décalage à droite
- Attention : & ≠ &&
- **Exemples**

5	0000 0000 0000 0101
20	0000 0000 0001 0100
5 & 20	0000 0000 0000 0100 => 5 & 20 => 4
5 | 20	0000 0000 0001 0101 => 5 | 20 => 21
5 ^ 20	0000 0000 0001 0001 => 5 ^ 20 => 17
~5	1111 1111 1111 1010 => -6

- Affectation/bit-à-bit : &=, |=, ^=, ~=

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEUR CONDITIONNEL

- **Syntaxe**

  expression1 ? expression2 : expression3 ;

- **à peu près équivalent à :**

```
si (expression1) alors
        expression2
 sinon
        expression3
 finsi
```

- **Exemple :**

```
vAbsolue = (x>=0) ? x : -x;
```

- **Conseil** : ne pas utiliser (peu clair)

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEUR D'ADRESSAGE

- Adresse de : &
  - Syntaxe : &variable , donne l'adresse mémoire de la variable
  - Exemple :

```
int i;
scanf(''%d'',&i);
```

Ne pas confondre avec l'operateur bit à bit

- Valeur dont l'adresse est : *
  - Syntaxe *adress : donne le mot mémoire dont l'adresse est donnée par adress
  - Exemple :
    - int i, *adr;
    - i=1;
    - adr = &i;
    - printf("%d", *adr); donne 1

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEURS DE TAILLE

- L'operateur **sizeof** donne la taille mémoire (en octet) d'une expression
- Donne la taille de l'implantation
- **Deux syntaxes:**
  - **sizeof expression**

```
int i,j ;
j= sizeof(i);  donne 2 ou 4 (octets)
```

  - **sizeof(type) ou sizeof(expression)**

```
typedef char tab[100];
tab t;
int n;
n = sizeof(int); donne 2 ou 4 (octets)
n = sizeof(tab); donne 100 (char)
```

GÉNÉRALITÉS, TYPES ET OPERATEURS
OPÉRATEURS DIVERS

- ( ) : force l'ordre des calculs
  - **Exemple** :
    - 1+2 * 3; donne 7
    - (1+2) * 3; donne 9
- [ ] pour les tableaux
  - t[2] équivalent à *(t+2)
- -> et . :opérateurs sur les structures et les classes(notion avancée)

GÉNÉRALITÉS, TYPES ET OPERATEURS
PRIORITES

Priorité	Opérateurs	Description	Associativité
15	() ,[ ] ,-> ,.	opérateurs d'adressage	->
14	++, --	incrément/décrément	<-
	~	complément à un (bit à bit)	
	!	non unaire	
	&, *	adresse et valeur (pointeurs)	
	(*type*)	conversion de type (cast)	
	+,-	plus/moins unaire (signe)	
13	* , /, %	opérations arithmétiques	->
12	+,-	""	->
11	<<, >>	décalage bit à bit	->
10	<, <=, >, >=	opérateur relationnels	->
9	==, !=	""	->
8	&	et bit à bit	->
7	^	ou exclusif bit à bit	->
6	|	ou bit à bit	->
5	&&	et logique	->
4	||	ou logique	->
3	?:	conditionnel	<-
2	=,+=, -=, *=, /=, %= >>=, <<=, &=, ^=, |=	assignations	<-
1	,	séparateur	->

Feedback sur:
pape.abdoulaye.barro@gmail.com