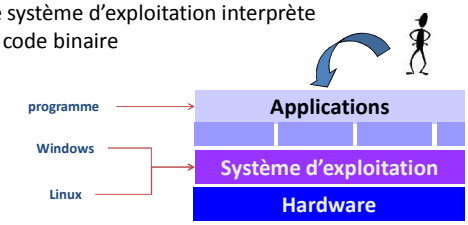
Informatique S1
Programmation C

- *Objectifs* :
  - Introduction système binaire
  - Représentation des caractères

Exécution d'un programme

- Code binaire
  - Le « langage » compris par les machines
  - Le système d'exploitation interprète le code binaire



Codage binaire

- Toutes communications à l'intérieur de l'ordinateur sont faites avec des signaux électriques
- Pour simplicité et fiabilité, ces signaux ont deux états seulement :
  - 0 – éteint (absence de signal électrique)
  - 1 – allumé (présence de signal électrique)
- Une **unité d'information** (**0** ou **1**) est appelée **bit** (de l'anglais *binary digit*)

Décimal X Binaire

- Le système décimal
  - Représentation : 10 symboles différents
    - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  - Représentation d'un numéro (580) :
    - **5** centaines, **8** dizaines, **0** unités
  - Équivalent mathématique :
    - $\mathbf{5}x10^2 + \mathbf{8}x10^1 + \mathbf{0}x10^0$

Décimal X Binaire

- Système binaire :
  - Représentation : 2 symboles différents
    - **0 (faux)** et **1 (vrai)**
  - Représentation d'un numéro (6) :
    - **110** → **1**x4 + **1**x2 + **0**x1
  - Équivalent mathématique :
    - **1**x2^2 + **1**x2^1 + **0**x2^0

Bits, Bytes, octets, etc...

- ***bit*** *– une unité binaire (0 ou 1)*
- ***octet*** *(ou* ***Byte****) – groupe de 8 bits*
- ***Kilo-octets*** *(Ko) – 1024 octets*
- ***Méga-octets*** *(Mo) – 1024Ko - 1048576 octets*
- ***Giga-octets*** *(Go) – 1024 Mo - 1073741824 octets*

- *Pourquoi 1Ko ≠ 1000 octets?*
  - *Encore, à cause de la base binaire*
  - *2^{10} = 1024*

Taille des types des données

- **int**
  - 16bits → 2^{16} possibilités
  - de **-32768** à **+32767**
- **short int**
  - 8bits → 2^8 possibilités
  - de **-128** à **+127**
- **long int**
  - 32bits → 2^{32} possibilités
  - de **-2147483648** à **+2147483647**

unsigned
(sans signal)
- unsigned short
  - de 0 à 255

Taille des types des données

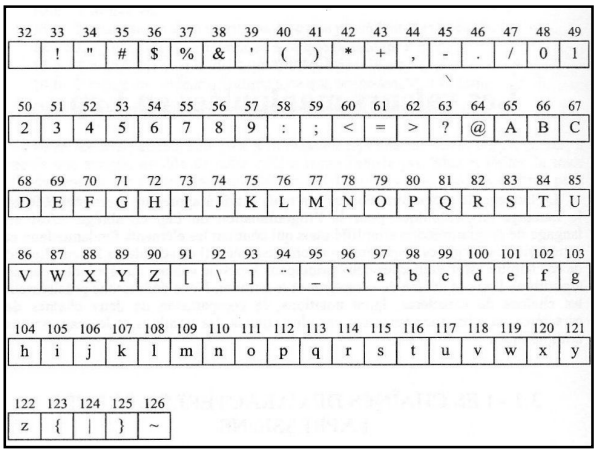
- **float**
  - 32 bits
  - Jusqu'à 3.4e+38
- **double**
  - 64 bits
- **char**
  - 8 bits
  - **256 caractères**

Représentation du type char

- La représentation des caractères par une séquence de bits comme pour les numéros entiers
  - C'est l'indication du type des données qui permet de faire la différence entre eux
- Il faut **coder** les caractères (chiffres, lettres et autres symboles) sous un **format** qui peut être reconnu par tous les ordinateurs
  - **ASCII**
  - UNICODE

ASCII

- Norme internationale pour la représentation des caractères
- 256 caractères y sont représentés
  - Chiffres 0 à 9
  - Lettres de l'alphabet en majuscule et minuscule
  - Caractères spéciaux
    - Space, *, /, \, <, >, !, ?, etc.
  - Caractères de contrôle
    - Nouvelle ligne, bip, tabulation, etc.



32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49
	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/	0	1
50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67
2	3	4	5	6	7	8	9	:	;	<	=	>	?	@	A	B	C
68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103
V	W	X	Y	Z	[	\	]	¨	_	`	a	b	c	d	e	f	g
104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121
h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y
122	123	124	125	126													
z	{	|	}	~													

Type char en langage C

- Représentation selon le code **ASCII**
- char **c = 'A'**;
  correspond à (int) c == 65
- char **c = '0'**;
  correspond à (int) c == 48

Rappel !
Pour mentionner un char, on utilise ' ' :
'a', 'b', '0', '1', '\n', '\t'…